Contents

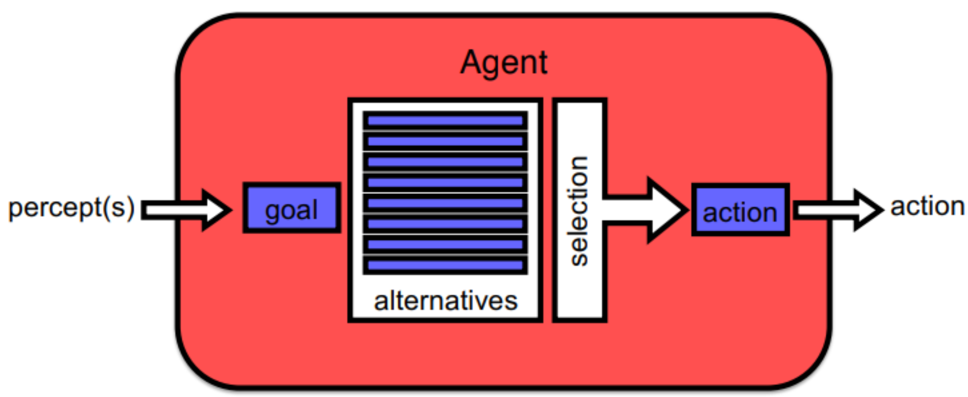
1 Limitations of reactive architectures

- there are many things agents with reactive architectures cant do:
  - they cant **represent or reason** about hypothetical objects and times;
  - they dont do well in domains where plausible actions **cant be ignored or undone** if they prove to be unwise;
  - it is difficult for purely reactive agents to **organise their own activities** over time or to coordinate their behaviour with that of other agents in a non-trivial way
- its difficult to get intelligent behaviour from a purely reactive agent

2 Deliberation

- deliberation is the explicit consideration of **alternative courses of action**
- deliberation involves **generating** alternatives and **choosing** an alternative
- an agent can deliberate about:
  - **means**: how to achieve a goal (this lecture)
  - **ends**: whether to achieve (intend) a goal
- **Examples**: deliberating about means
  - problem solving (search)
  - planning
  - scheduling
  - theorem proving
  - constraint satisfaction

3 Deliberation architecture



3.1 Means

- in a deliberative architecture, percepts (or communication) **give rise to goals** (representations of a state to be achieved) and representations of the **current state of the world**
- the agent deliberates about how to achieve the goal
  - deliberation involves (usually systematic) **exploration of alternative courses of action**
  - a deliberative architecture typically includes automatic generation and comparison of alternatives
- result of deliberation is a **representation of the action(s) to be performed**

3.2 The role of representations

- deliberation involves the manipulation **of a model** of the world and possible courses of action, rather than the world itself
- requires the ability to represent actions and derive the consequences of actions **without actually performing them**, e.g.:
  - by remembering their effects in previous, similar situations
  - by reference to a causal model of the world

3.3 Counterfactual representations

- to represent desired states and the consequences of actions:
  - some states of the agent must be counterfactual in the sense of **referring to hypothetical future states** (goals) or as yet unexecuted actions (plans)
  - some of the basic operations of the architecture should generate such counterfactual states
  - such states must be influential in the choice of actions
- to represent hypothetical situations, a deliberative agent requires representations with *compositional semantics*

3.4 Advantages of deliberation

- useful when the **penalty** for incorrect actions is high, e.g., when the environment is hazardous
- allows us to code a general procedure for finding a solution to a class of problems
  - may be better than reactive systems at **coping with novel problems**
  - we may be able to get a correct or even an optimal answer, e.g., decision theoretic approaches

4 TSP algorithms

- many TSP algorithms use iterative refinement, i.e., they require the representation of at least two alternative tours:
  - the *current best tour* (best known route)
  - the *candidate tour* (often a modification of the current best tour)
- iterative algorithms typically stop when no modification of the current best tour **has lower cost** or when there has been **no improvement** for n iterations
- agent then executes the steps in the current best tour

4.1 Shakey

Look at lecture notes here

5 Planning

- the agents knowledge of the *initial state* of the world is often **incomplete or mistaken**
- the world is continually changing and continues to change **while the agent is planning**
- actions dont always have the intended effectactions can fail or just have **unpredictable outcomes**
- the agent may **make errors executing the plan**
- the time available to find a solution may be **limited**

5.1 Simplifying assumptions

- the agent has **perfect knowledge of the world**, including the location and properties of all the objects in the world
- the world is **static**i.e., it doesnt change unless the agent changes it
- the world is **deterministic**: we know in advance **the effect** of performing an action in the world and each action has a single outcome
- plan execution is error-free and it doesnt matter how long it takes the agent to find a plan

5.2 Classical planning

- if we make these assumptions we get classical planning (production of a complete, totally ordered set of actions, which, when executed in a given initial situation, will achieve a goal)
- resulting sequence of actions will typically only work if the **simplifying assumptions hold**
- implies some way of coping with inaccurate world models, nondeterministic actions, plan execution errors, etc

5.3 STRIPS

- states are represented as conjunctions of (function-free) ground literals
- goals are represented by conjunctions of literals, possibly containing existentially quantified variables
- actions are represented by operators which specify
  - the name of the action
  - the precondition: a conjunction of positive literals specifying what must be true for the action to be applicable
  - the postcondition: a conjunction of literals specifying how the situation changes when the operator is applied
- for example, an operator which stacks one block on top of another in the blocks world could be specified as
  - $[Clear(x), Clear(y)]STACK(x, y)[On(x, y), Clear(y)]$
- the precondition implicitly refers to the situation, s, immediately before the action, and the postcondition implicitly refers to the situation, s, which results from performing the action
- in s , all the positive literals in the postcondition hold, as do all the literals that held at s, except for those that are negative literals in the postcondition

5.4 Regression planning

- one way to solve STRIPS problems is to **search backwards** from the goal in world (situation) space
- operator preconditions become *subgoals* - we stop when the operator preconditions are satisfied in the current state
- searching backwards often **reduces** the branching factor
- in typical problems the goal state has a small number of conjuncts, each of which is made true by a small number of operators, while there are many operators that can be applied in the initial state
- resulting plan is a series of instantiated operators which, if applied in the initial state, result in the goal state

Reference section

compositional semantics

The meaning of a phrase is determined by combining the meanings of its subphrases, using rules which are driven by the syntactic structure.