Multi Agent Systems

Reactive Agents

and

Hybrid Agents

B. Tenbergen (btenberg@uos.de)

1. Overview

- 1. Overview
- **2. Model 1: Reactive Agents**
  - **2.1 The subsumption architecture**
  - **2.3 Alternatives**
- 3. Model 2: Hybrid Agents
  - 3.1 Structures
  - 3.2 InteRRaP

Reactive Agents

A simple and fast solution

2.0 Reactive Agents

Introduction

- There are many unsolved problems in symbolic AI
- Some claim that minor changes to the symbolic approach are not sufficient for real world environments
- A new approach was necessary
- Rodney Brooks, one of the most influential critics developed in 1991one of the best known reactive agent architectures

2.0 Reactive Agents

Brooks main theses

- Intelligent behaviour can be generated without explicit representation of the kind that symbolic AI proposes
- Intelligent behaviour can be generated without explicit abstract reasoning of the kind that symbolic AI proposes
- Intelligence is an emergent property of certain complex systems

2.0 Reactive Agents

Brooks key ideas

- Situatedness & embodiement: real intelligence is situated in the world, not in disembodied systems such as theorem provers or expert systems
- Intelligence and emergence: Intelligent behaviour arises as a result of an agents interaction with ist environment. Also intelligence is in the eye of the beholder, it is not an innate, isolated property

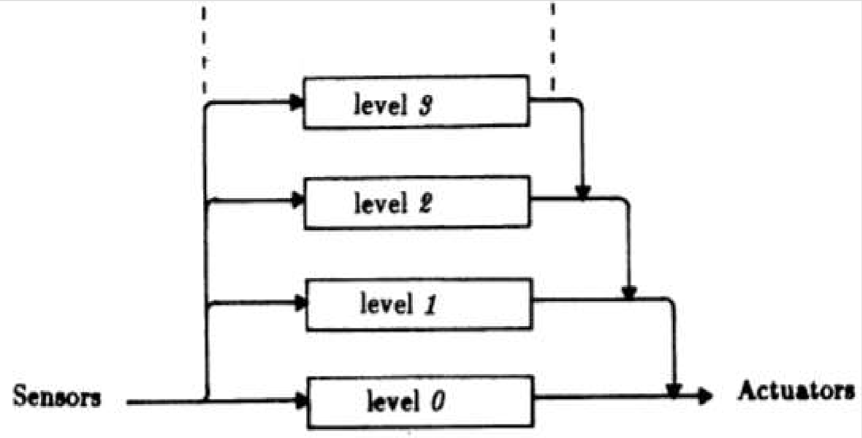
The subsumption architecture

- A subsumption architecture is a hierarchy of task accomplishing behaviours
- Each behaviour is a individual action function
- Each of this behaviour modules is intended to solve some particular task
- The modules are finite state machines
- Several behaviours can fire simultaneously and compete over the control of the agent

- No complex symbolic representations involved
- No symbolic reasoning done

Subsumption architecture: Decision making

- A subsumption hierarchy chooses which action to peform
- Behaviours are arranged in layers
- The lower the layer the higher the priority
- Lower layers can inhibit higher layers



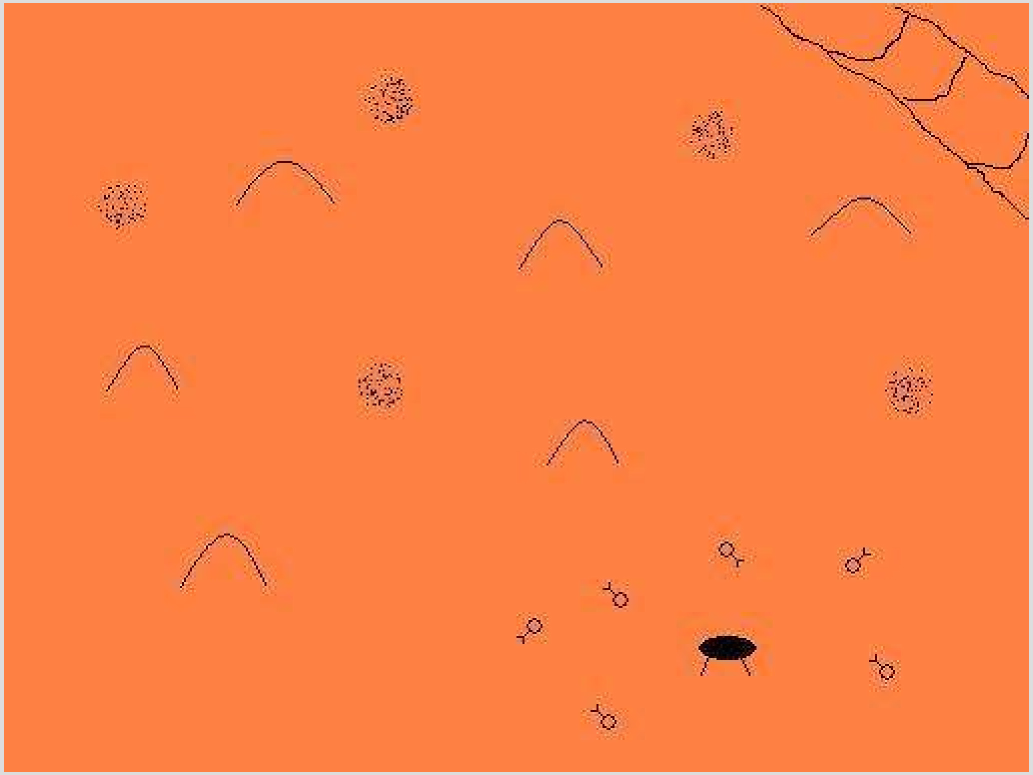
2.2 Reactive Agents

Example: Mars explorer

- Task:
  - Robots on mars should collect precious rocks
  - Location of rocks is not known, but rocks are clustered together
  - No direct communication between agents possible
  - Position of mothership is known through a radio signal
- The claim is that logic based agents are „entirely unrealistic“ for this problem

2.2 Reactive Agents

Mars Explorer



Solution

- The first trick is the radio signal. As a radio signals strength decreases with the distance the mothership can be easily found by travelling up the gradient
- The second trick are radioactive crumbs the agents can deploy, detect and pick up

The subsumption hierarchy

- If true → move randomly
- If detect sample → pick up sample
- If carrying samples & not at base → travel up gradient
- If carrying samples & at base → drop samples
- If detect obstacle → change direction

Multi agent solution

If true → move randomly

If sense crumbs → pick one up
→ Travel up gradient

If detect sample → pick up sample

If carrying samples & not at base → drop 2 crumbs
→ travel up gradient

If carrying samples & at base → drop samples

If detect obstacle → change direction

– 1985, Rosenschein & Kaelbling „situated automata“
  - Just because an agent is conceptualized in logical terms, it need not be implemented as a theorem prover
  - In their situated automata paradigm an agent is specified in declarative, rule like terms. This rules are then compiled down to a digital machine, which satisfies the declarative specification
  - The digital machine can operate in a provable time bound, reasoning is done at compile time, not at run time

- A specification of the semantics of the agent's inputs (‘whenever bit 1 is on, it is raining’); a set of static facts (‘whenever it is raining, the ground is wet’); and a specification of the state transitions of the world (‘if the ground is wet, it stays wet until the sun comes out’).
  The programmer then specifies the desired semantics for the output (‘if this bit is on, the ground is wet’), and the compiler ... [synthesizes] a circuit whose output will have the correct semantics. ... All that declarative ‘knowledge’ has been reduced to a very simple circuit.” [Kaelbling, 1991]

2.3 Reactive Agents

Alternatives

- The theoretical limitations of the approach are not well understood
- Compilation with propositional specifications is equiva-lent to an NP-complete problem
- The more expressive the agents specification is, the harder is it to compile
- There are some deep theoretical results which say that after a certain expressiveness, the compilation can't be done anymore

2.3 Reactive Agents

Alternatives

- 1987, Agre and Chapman proposed an architecture based on running arguments. The idea is that most activity is routine, once learned, little variation is nee-ded
- Routine decisions are encoded in digital circuits which only need periodic updating
- Used in a famous computer game named PENGI

Alternatives

- Agent network architecture developed by Pattie Maes in 1991
- Agent is defined as set of competence modules
- Each module has preconditions, postconditions and an activation level.
- The activation level indicates the relevance of that module in the specific situation
- Modules are linked

2.4 Reactive Agents

Advantages & Disadvantages

- One of the main concerns against logic based decision making was ist theoretical complexity and the resulting long computing time.
- The overall complexity of the subsumption action function is no worse than $O(n^2)$ where n is the larger of the number of percepts or behaviours
- In practice the decision making logic can be encoded into hardware, giving constant decision making time. This means an agent can be guaranteed to select an action within microseconds

2.4 Reactive Agents

Advantages **& Disadvantages**

- If agents do not have models of their environment, they must have sufficient information available in their local environment to determine an acceptable action
- Difficult to take non local information into account
- Difficult to design agents that learn
- Complex to engineer agents
- Complexity rises with the number of layers

Overview

- 1. Overview

- 2. Model 1: Reactive Agents
  - 2.1 The subsumption architecture
  - 2.3 Alternatives

- **3. Model 2: Hybrid Agents**
  - **3.1 Structures**
  - **3.2 InteRRaP**

Hybrid Agents

Combining the capability of reactive and proactive behaviour

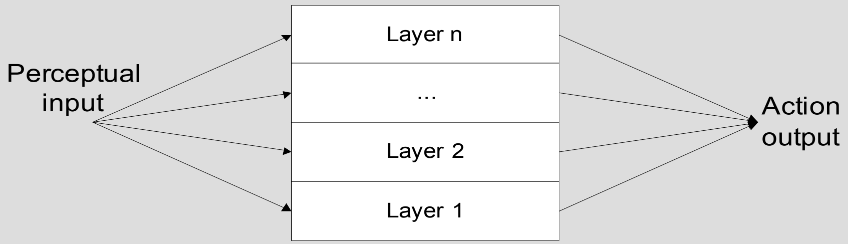
3.0 Hybrid Agents

How to combine reactive and proactive capabilities?

- Our architecture will need at least two systems, a reactive and a proactive one
- We simply put these subsystems into a hierarchally ordered laye-red architecture
- We can build up two types of layered architectures as far as the control flow is concerned

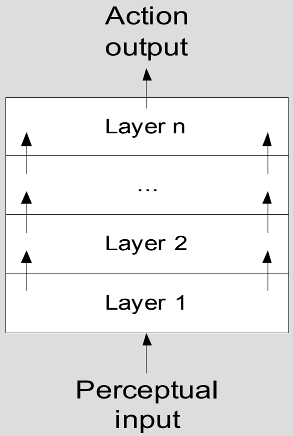
Horizontal and vertical layering

Horizontal layering:

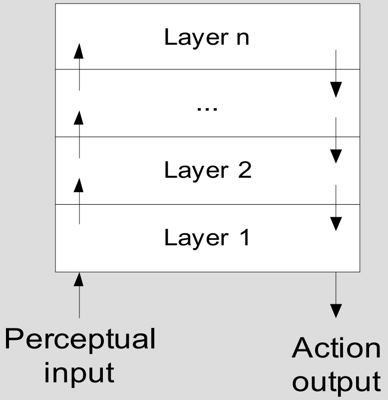


Vertical layering:

One pass control



Two pass control



3.1 Hybrid Agents

Properties

Horizontally layered:

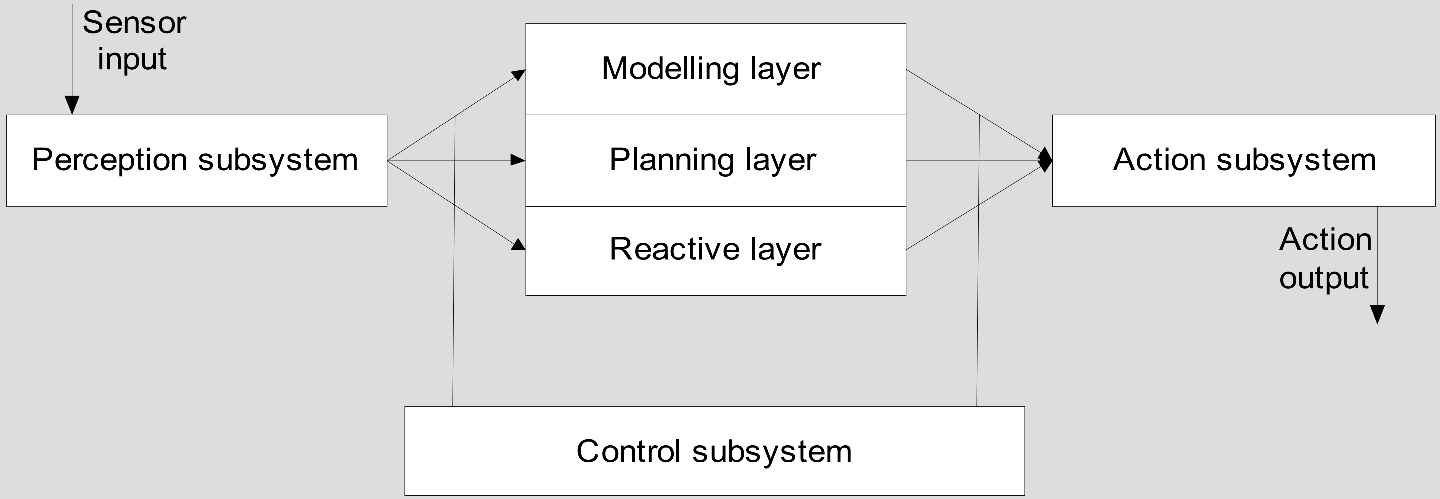
- very simple concept (one layer for each type of bahaviour
- often need a mediator fonction that ensures consistency, i.e. that decides which layer has control over the agent
- problems arise with growing number of layers *n* and and their possible actions *m* (the designer has to deal with *m* to the power of *n* interactions between the layers)
- the introduction of a mediator function as a central control also introduces a bottleneck to the agent‘s decision making

Vertically layered:

- Two types (one-pass and two-pass)
- Control flows sequentially through each layer
- Less complex as there are only *n*-1 interfaces between the layers, i.e. at most $m^2(n-1)$ interactions to be considered
- But thereby also less flexible as failures in any layer may have serious consequences for the agent‘s performance
- Analogy between two-pass architecture and organizations

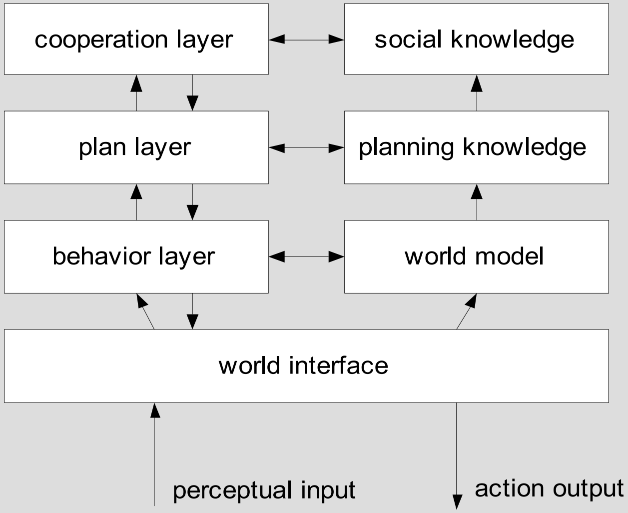
3.1 Hybrid Agents

Touring Machines



- An example for a horizontally layered agent architecture
- Each layer produces action suggestions
- Reactive layer – immediate response
- Planning layer – ‚day-to-day‘-running (schemas („plan-library“))
- Modelling layer – conflict resolving (represents all entities in the world)

InteRRaP



- An example for a vertically two-pass agent architecture
- Layers correspond to the respective layers in TouringMachines
- Main difference to TouringMachines are the knowledge bases
- Social knowledge – plans and actions of other agents in the environment
- Planning knowledge – plans and actions of the agent itself
- World model – ‚raw‘ information about the environment

3.2 Hybrid Agents

InteRRaP

Further differences to TouringMachines:

– No need for a control subsystem as the layers interact with each other

– If a lower layer is not competent to deal with a situation it passes control to a higher layer by *bottom-up activation*; if a higher layer makes use of the fascilities of a lower layer in order to achieve ist goals this is referred to as top-down execution

InteRRaP

Each layer implements two general functions:

- *Situation recognition and goal activation:* creates new set of goals by matching current goals and respective knowledge base
- *Planning and scheduling:* selects plans to be executed with re-spect to the current plans, goals and knowledge base

Some Remarks

- Layered architectures are currently the most popular general class of agent architecture available
- But they lack the conceptual and semantic clarity of unlayered approaches