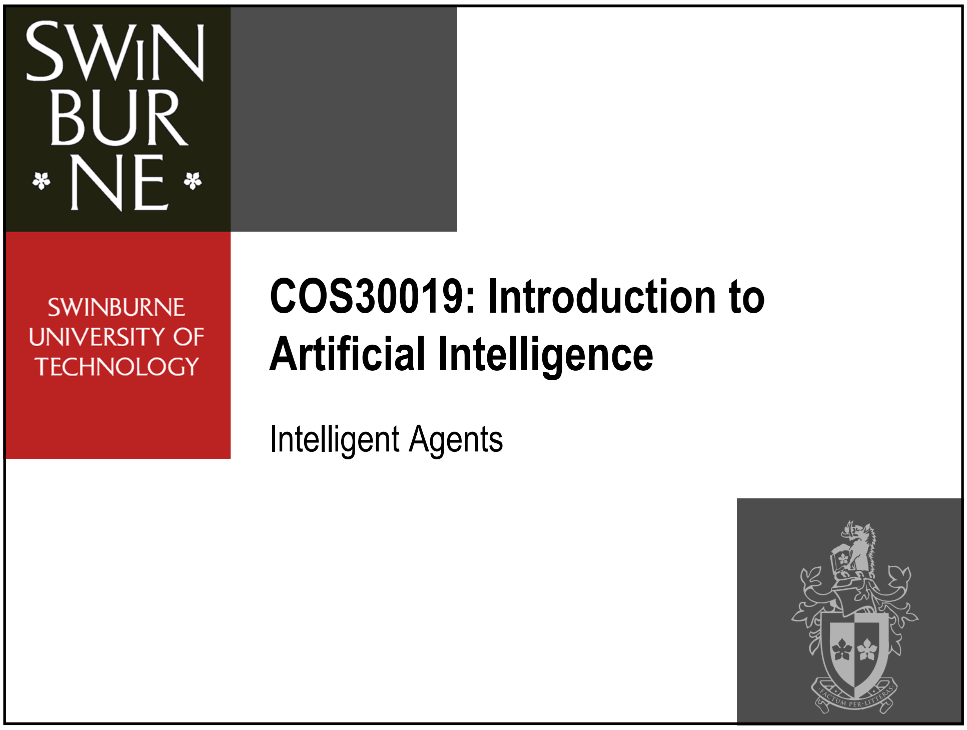
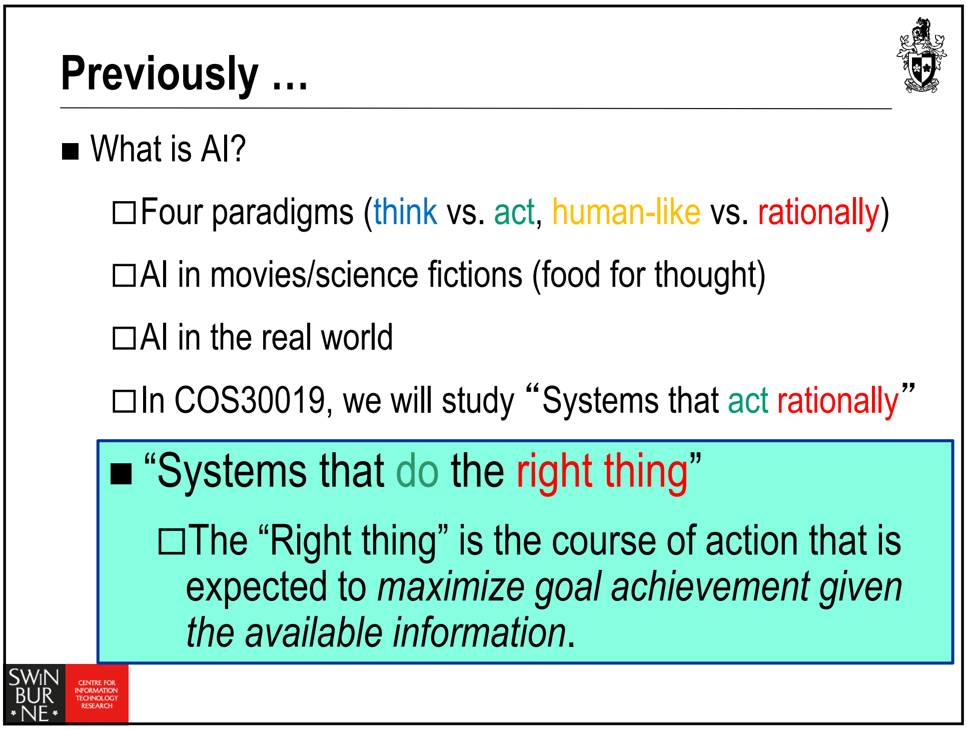


SWINBURNE UNIVERSITY OF TECHNOLOGY

COS30019: Introduction to Artificial Intelligence

Intelligent Agents

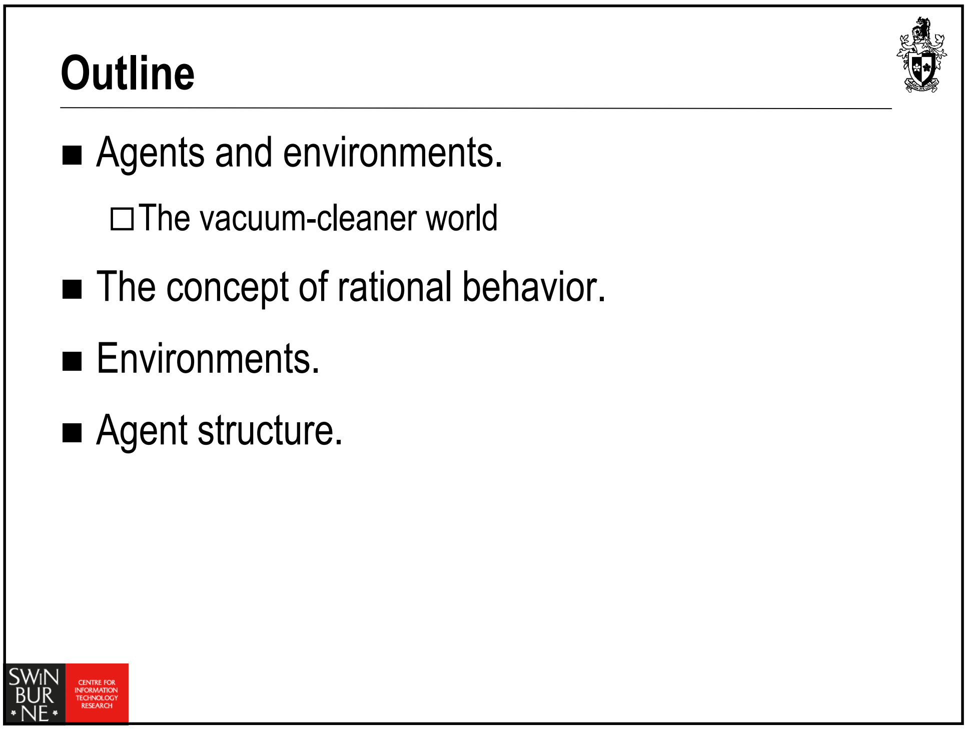
1



Previously …

- What is AI?
  - □Four paradigms (think vs. act, human-like vs. rationally)
  - □AI in movies/science fictions (food for thought)
  - □AI in the real world
  - □In COS30019, we will study "Systems that act rationally"
- "Systems that do the right thing"
  - □The "Right thing" is the course of action that is expected to *maximize goal achievement given the available information*.

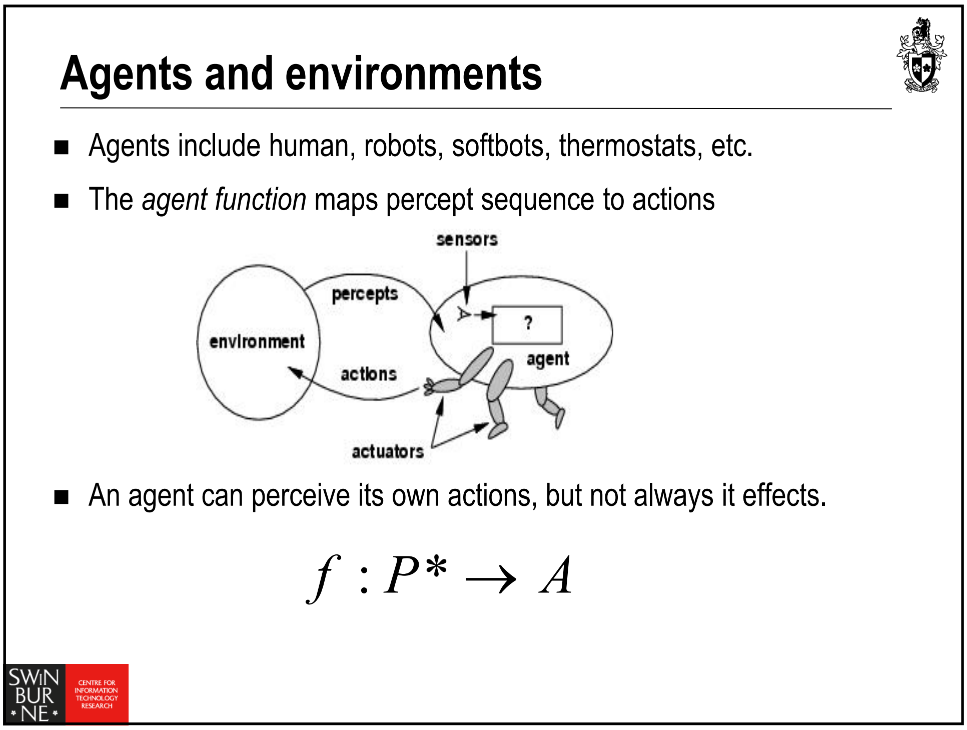
2



Outline

- Agents and environments.
  - □The vacuum-cleaner world
- The concept of rational behavior.
- Environments.
- Agent structure.

3



Agents and environments

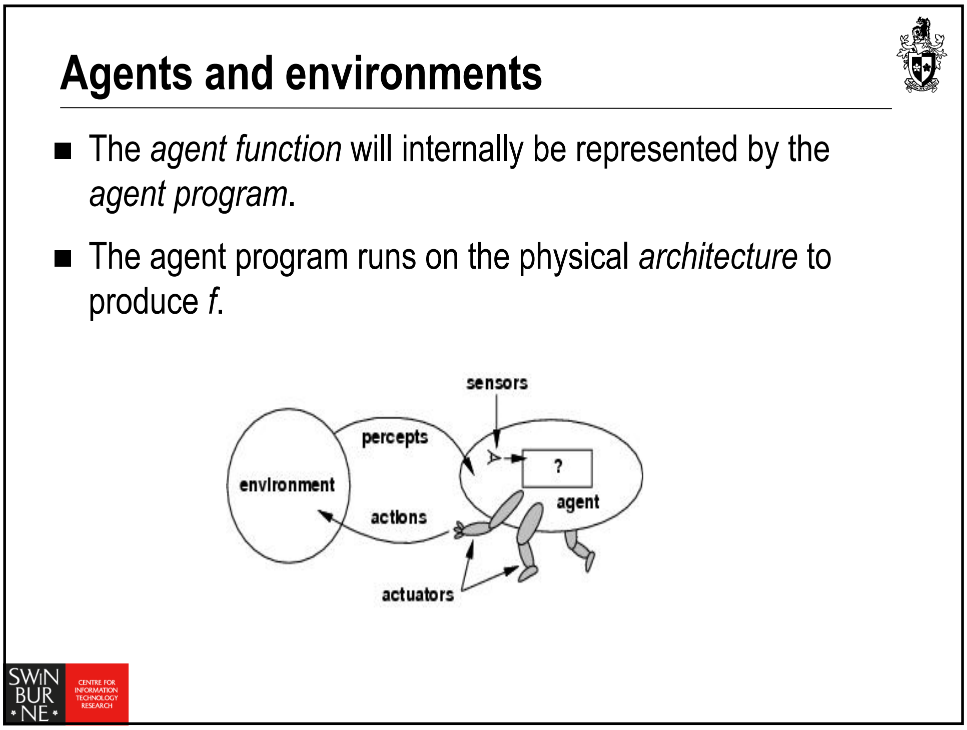
- Agents include human, robots, softbots, thermostats, etc.
- The *agent function* maps percept sequence to actions

sensors
percepts
environment
?
agent
actions
actuators

- An agent can perceive its own actions, but not always it effects.

$$f : P^* \rightarrow A$$

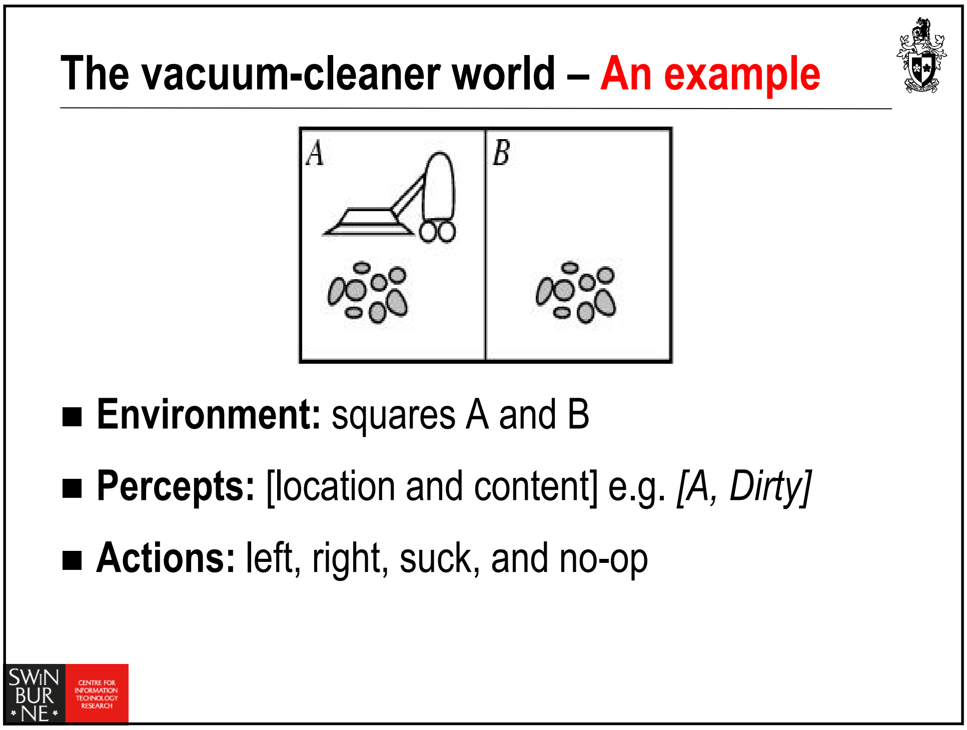
4



Agents and environments

- The *agent function* will internally be represented by the *agent program*.
- The agent program runs on the physical *architecture* to produce *f*.

5

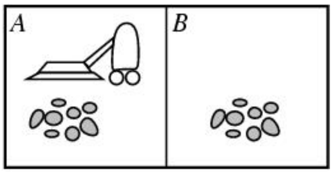


The vacuum-cleaner world – An example

- **Environment:** squares A and B
- **Percepts:** [location and content] e.g. *[A, Dirty]*
- **Actions:** left, right, suck, and no-op

6

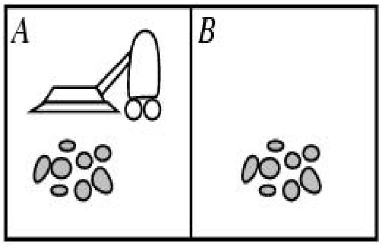
The vacuum-cleaner world – Agent function



Percept sequence	Action
[A,Clean]	Right
[A, Dirty]	Suck
[B, Clean]	Left
[B, Dirty]	Suck
[A, Clean],[A, Clean]	Right
[A, Clean], [A, Dirty]	Suck
…	…

7

The vacuum-cleaner world – An agent program



```
procedure REFLEX-VACUUM-AGENT ([location, status]) return an action
    if status == Dirty then return Suck
    else if location == A then return Right
    else if location == B then return Left
```

What is the right function? Can it be implemented in a small agent program?

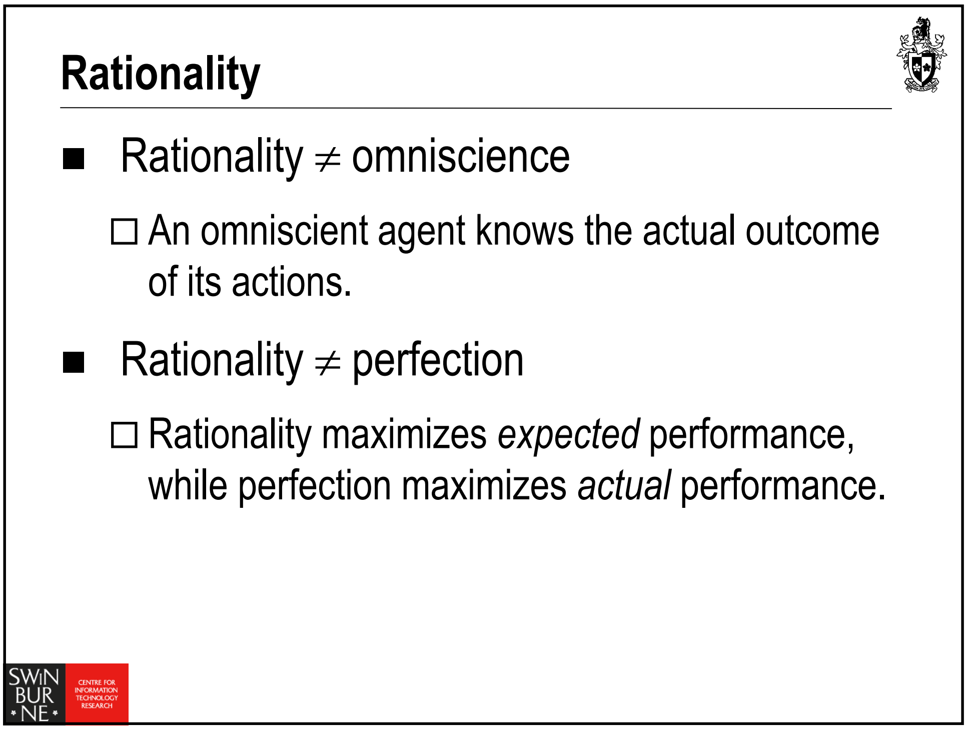
8

The concept of rationality

- A **rational agent** is one that does the right thing.
  - □ Every entry in the table is filled out correctly.
- What is the right thing?
  - □ Approximation: the most *successful* agent.
  - □ *Measure of success?*
- Performance measure should be objective
  - □ E.g. the amount of dirt cleaned within a certain time.
  - □ E.g. how clean the floor is.
  - □ …
- *Performance measure according to what is wanted in the environment instead of how the agents should behave.*

Rationality

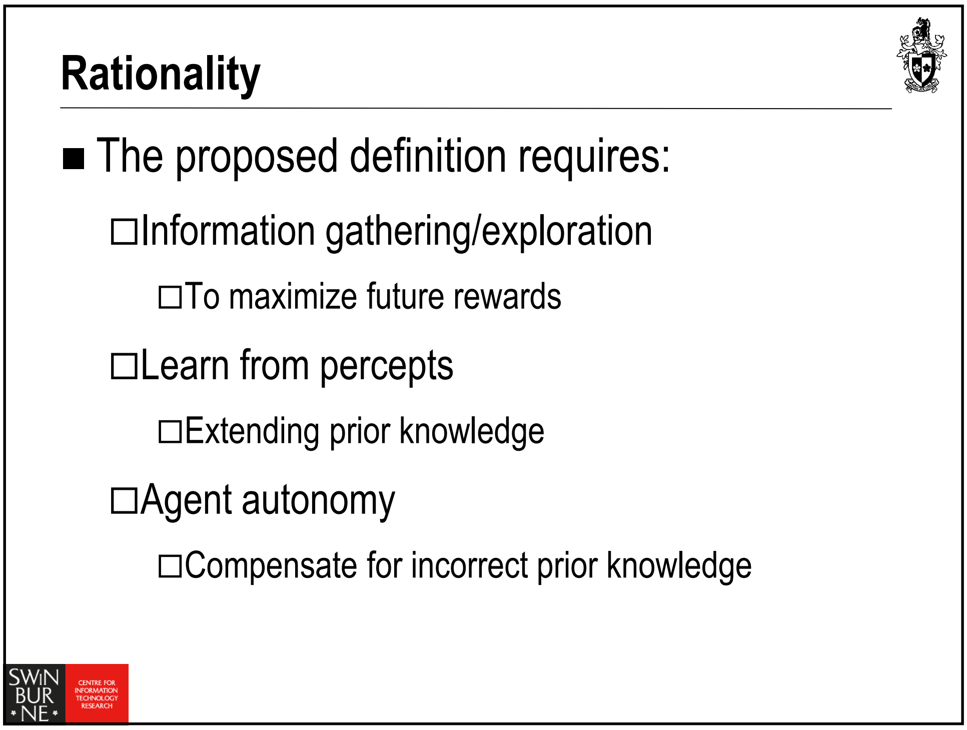
- What is rational at a given time depends on four things:
  - □ Performance measure,
  - □ Prior environment knowledge,
  - □ Actions,
  - □ Percept sequence to date (sensors).
- DEF: *A rational agent chooses whichever action that maximizes the expected value of the performance measure given the percept sequence to date and prior environment knowledge.*



Rationality

- Rationality ≠ omniscience
  - □ An omniscient agent knows the actual outcome of its actions.
- Rationality ≠ perfection
  - □ Rationality maximizes *expected* performance, while perfection maximizes *actual* performance.

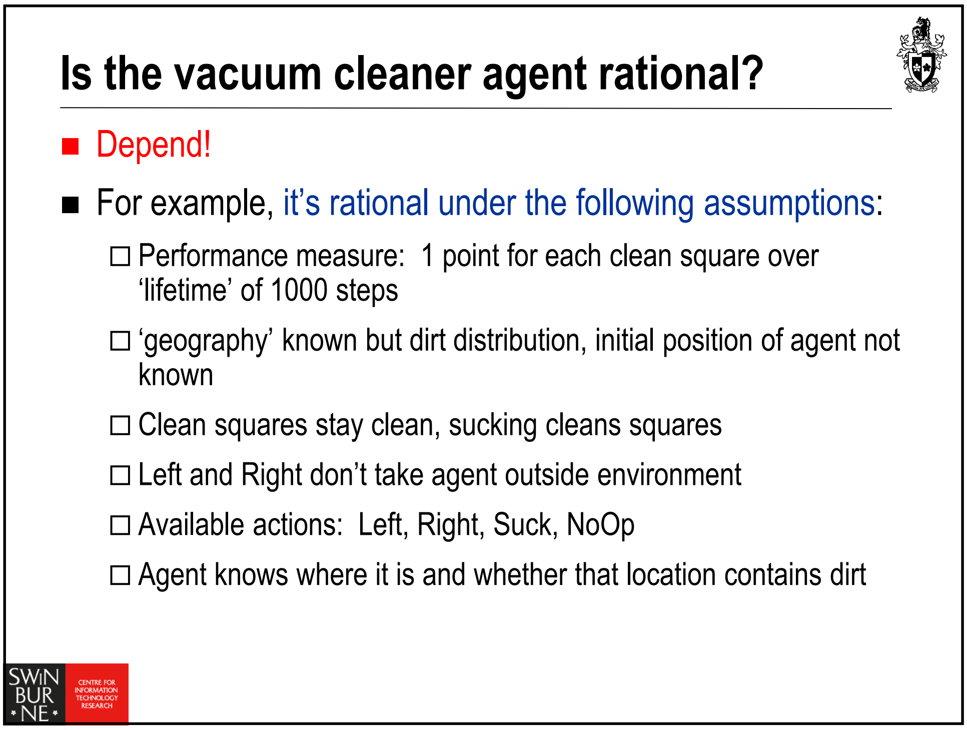
11



Rationality

- The proposed definition requires:
  - □Information gathering/exploration
    - □To maximize future rewards
  - □Learn from percepts
    - □Extending prior knowledge
  - □Agent autonomy
    - □Compensate for incorrect prior knowledge

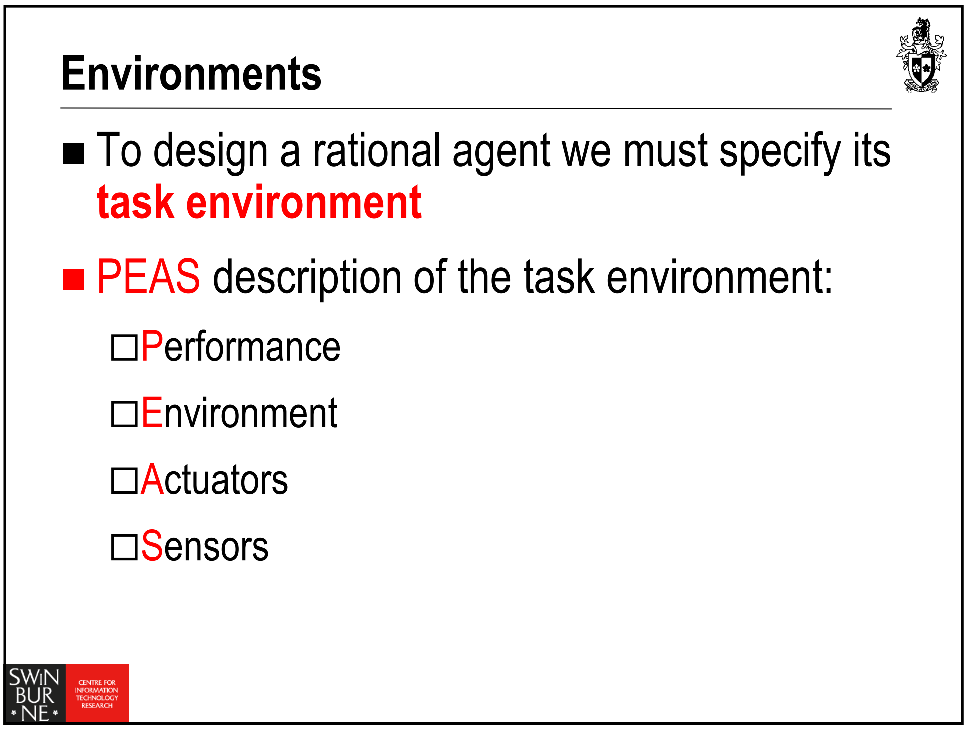
12



Is the vacuum cleaner agent rational?

- Depend!
- For example, it's rational under the following assumptions:
  - □ Performance measure: 1 point for each clean square over 'lifetime' of 1000 steps
  - □ 'geography' known but dirt distribution, initial position of agent not known
  - □ Clean squares stay clean, sucking cleans squares
  - □ Left and Right don't take agent outside environment
  - □ Available actions: Left, Right, Suck, NoOp
  - □ Agent knows where it is and whether that location contains dirt

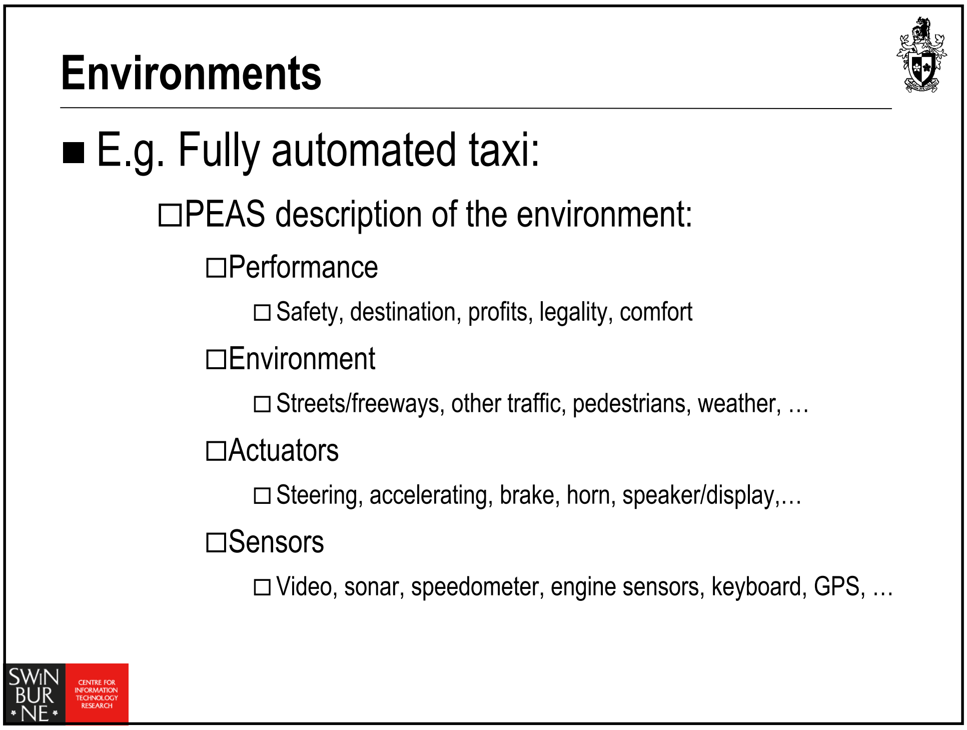
13



Environments

- To design a rational agent we must specify its **task environment**
- PEAS description of the task environment:
  - □ Performance
  - □ Environment
  - □ Actuators
  - □ Sensors

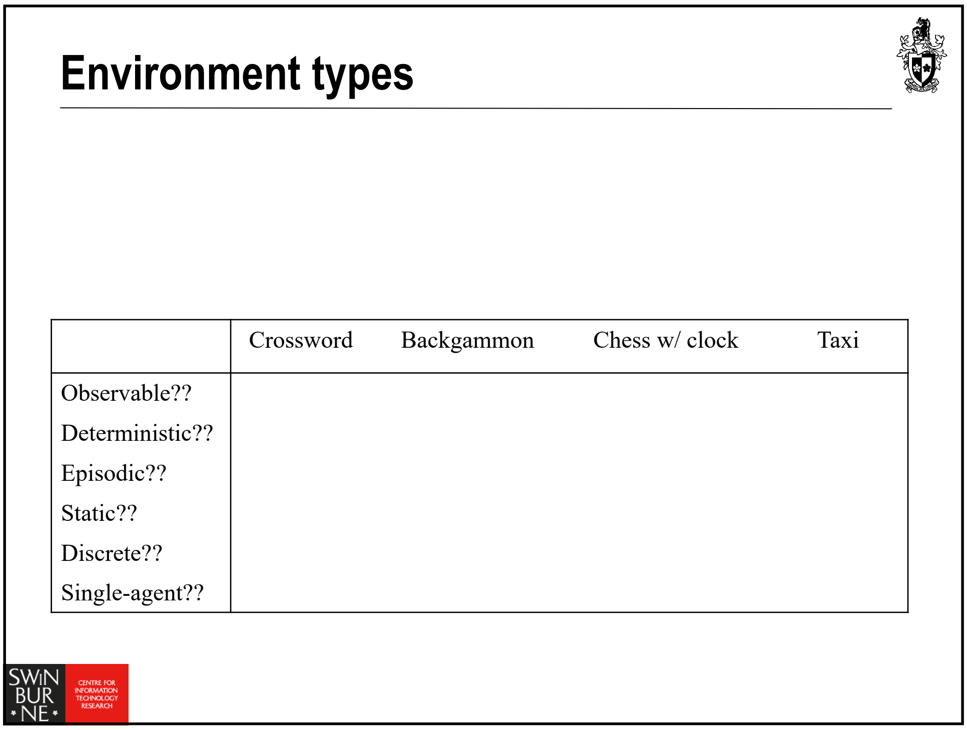
14



Environments

- E.g. Fully automated taxi:
  - □PEAS description of the environment:
    - □Performance
      - □ Safety, destination, profits, legality, comfort
    - □Environment
      - □ Streets/freeways, other traffic, pedestrians, weather, …
    - □Actuators
      - □ Steering, accelerating, brake, horn, speaker/display,…
    - □Sensors
      - □ Video, sonar, speedometer, engine sensors, keyboard, GPS, …

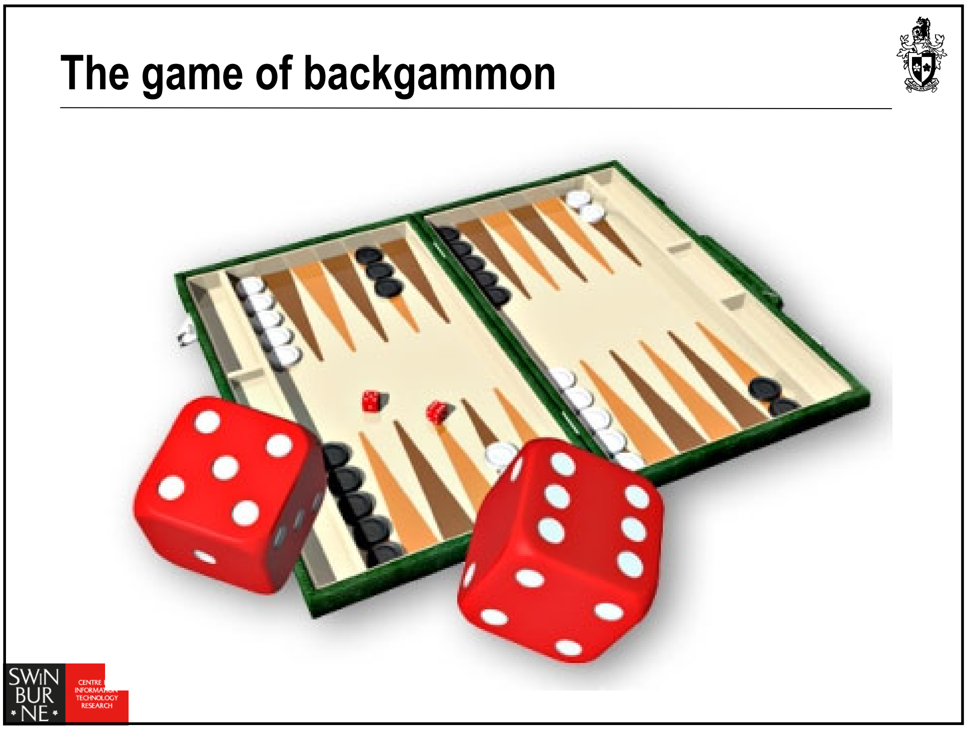
15



Environment types

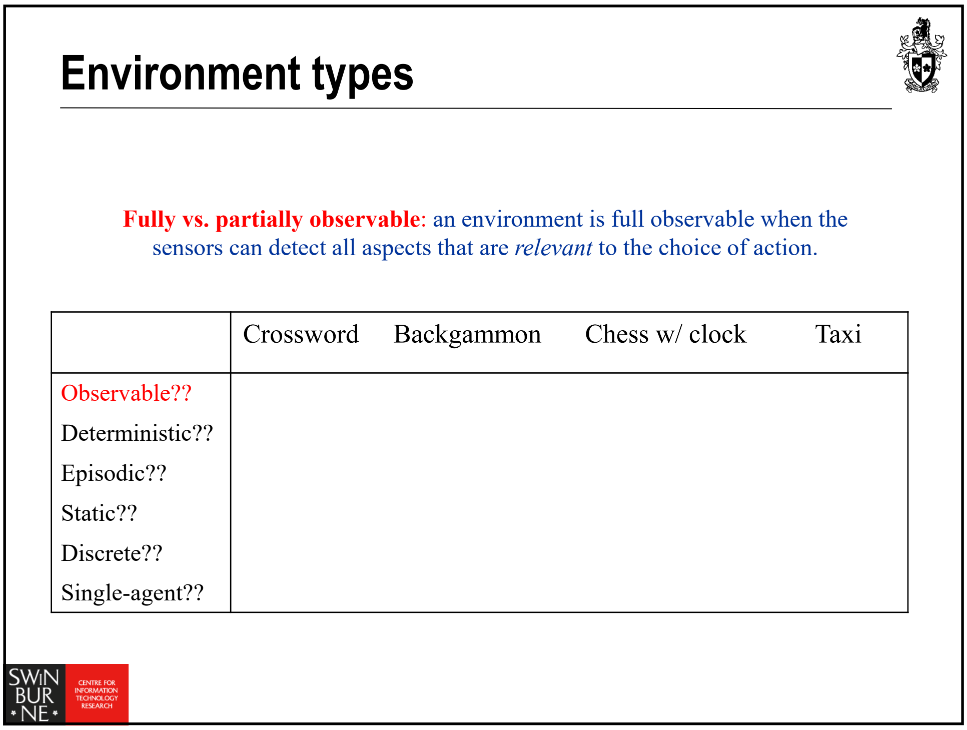
	Crossword	Backgammon	Chess w/ clock	Taxi
Observable??				
Deterministic??				
Episodic??				
Static??				
Discrete??				
Single-agent??				

16



The game of backgammon

17

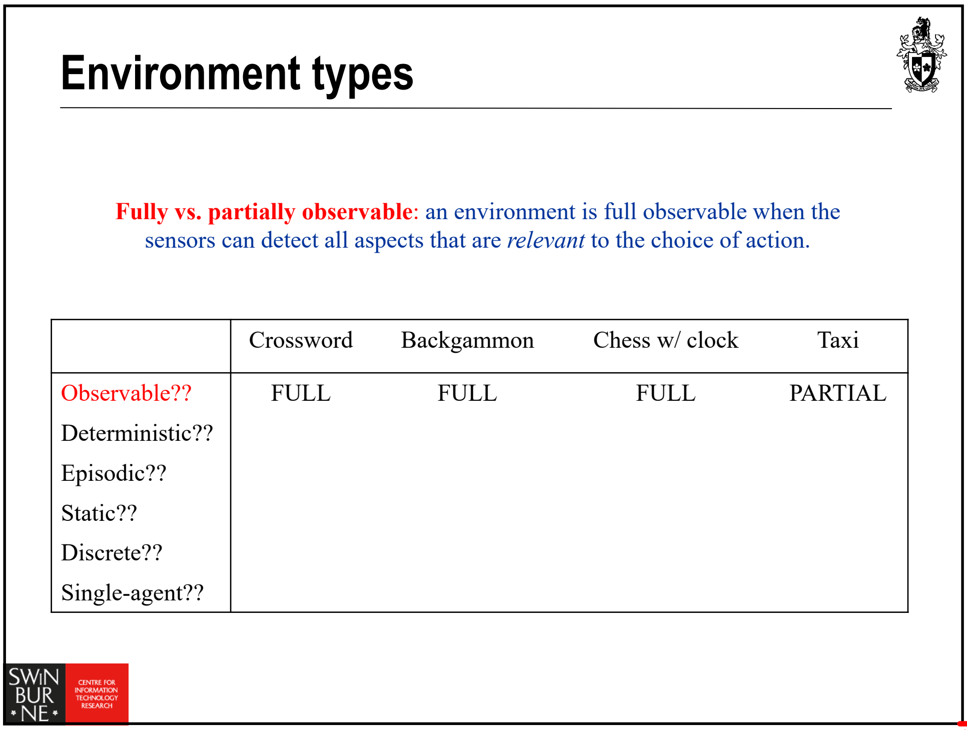


Environment types

Fully vs. partially observable: an environment is full observable when the sensors can detect all aspects that are *relevant* to the choice of action.

	Crossword	Backgammon	Chess w/ clock	Taxi
Observable??				
Deterministic??				
Episodic??				
Static??				
Discrete??				
Single-agent??				

18



Environment types

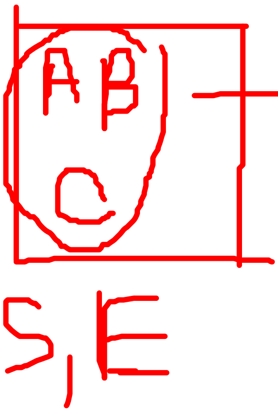
Fully vs. partially observable: an environment is full observable when the sensors can detect all aspects that are *relevant* to the choice of action.

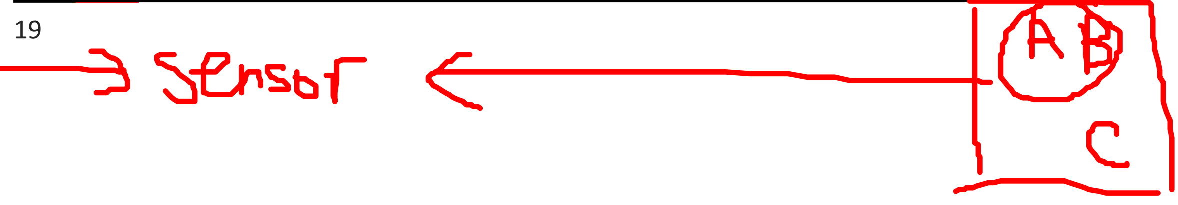
	Crossword	Backgammon	Chess w/ clock	Taxi
Observable??	FULL	FULL	FULL	PARTIAL
Deterministic??				
Episodic??				
Static??				
Discrete??				
Single-agent??				

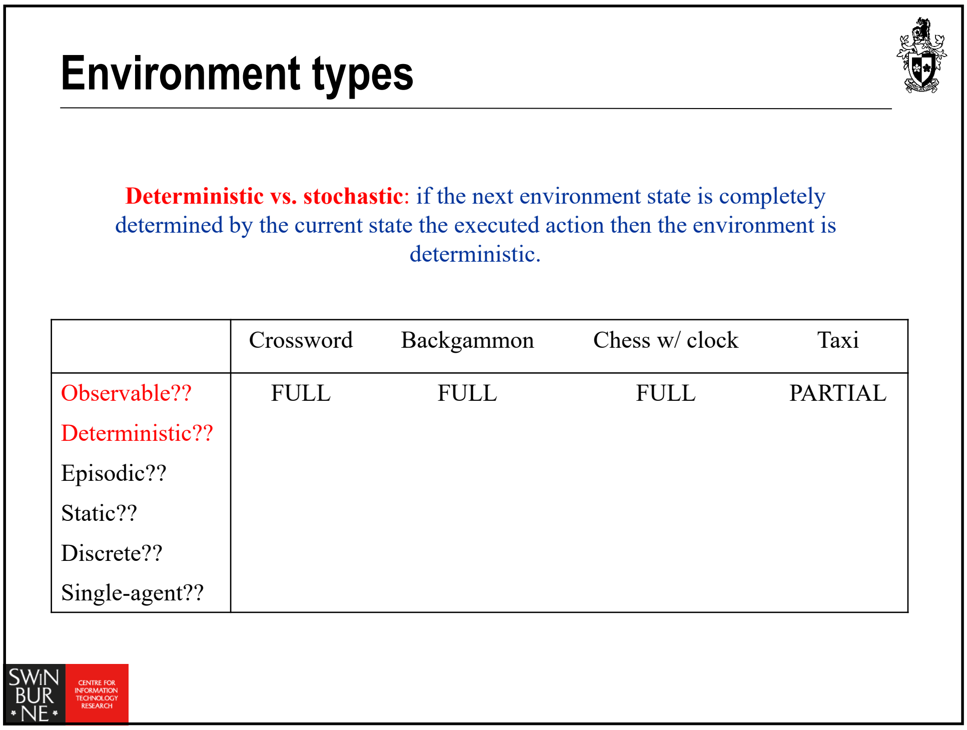
19

fully observable

partially observable







Environment types

Deterministic vs. stochastic: if the next environment state is completely determined by the current state the executed action then the environment is deterministic.

	Crossword	Backgammon	Chess w/ clock	Taxi
Observable??	FULL	FULL	FULL	PARTIAL
Deterministic??				
Episodic??				
Static??				
Discrete??				
Single-agent??				

20

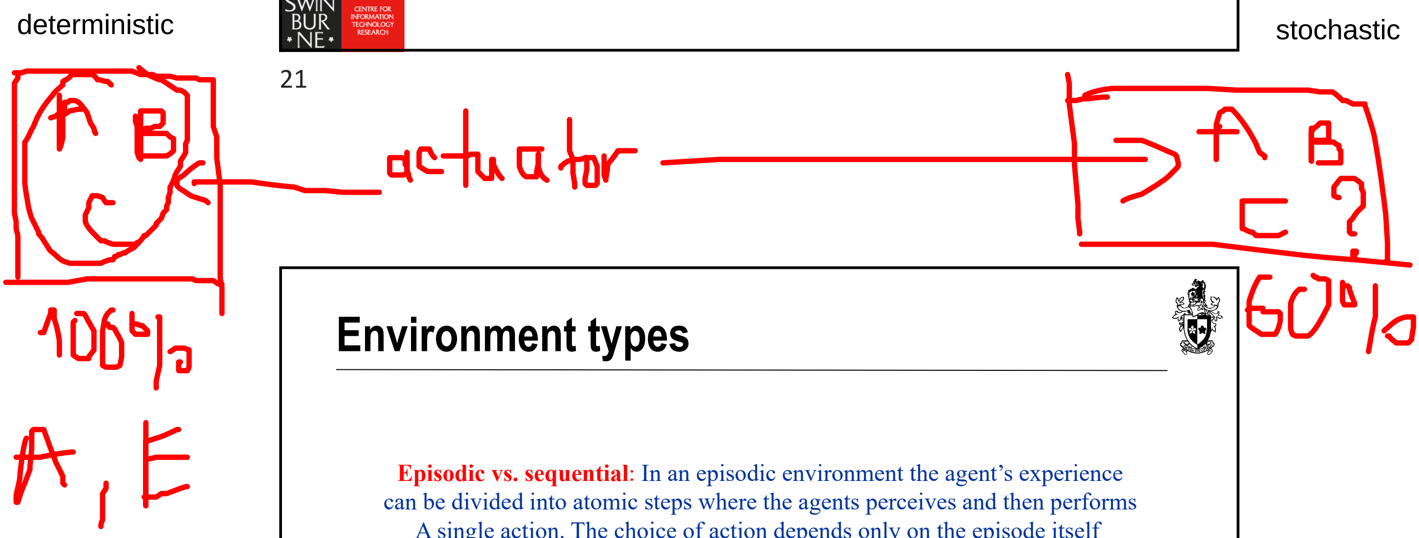
Environment types

Deterministic vs. stochastic: if the next environment state is completely determined by the current state the executed action then the environment is deterministic.

	Crossword	Backgammon	Chess w/ clock	Taxi
Observable??	FULL	FULL	FULL	PARTIAL
Deterministic??	YES	NO	YES	NO
Episodic??				
Static??				
Discrete??				
Single-agent??				

SWIN BUR NE CENTRE FOR INFORMATION TECHNOLOGY RESEARCH

21



Environment types

Episodic vs. sequential: In an episodic environment the agent's experience can be divided into atomic steps where the agents perceives and then performs A single action. The choice of action depends only on the episode itself

	Crossword	Backgammon	Chess w/ clock	Taxi
Observable??	FULL	FULL	FULL	PARTIAL
Deterministic??	YES	NO	YES	NO
Episodic??				
Static??				
Discrete??				
Single-agent??				

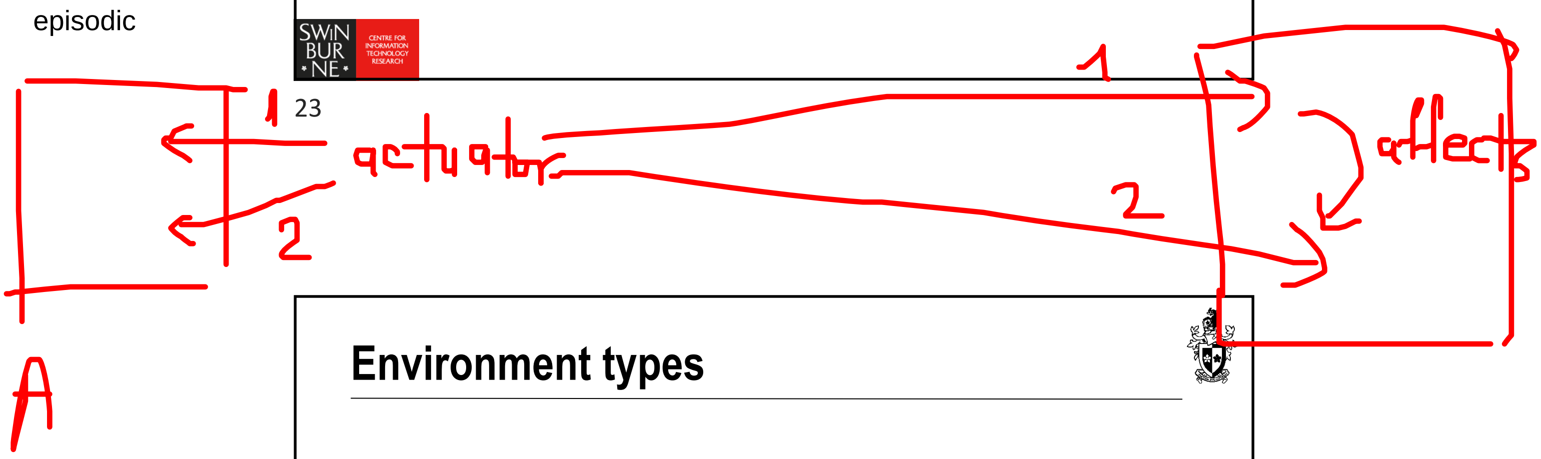
SWIN BUR NE CENTRE FOR INFORMATION TECHNOLOGY RESEARCH

22

Environment types

Episodic vs. sequential: In an episodic environment the agent's experience can be divided into atomic steps where the agents perceives and then performs A single action. The choice of action depends only on the episode itself

	Crossword	Backgammon	Chess w/ clock	Taxi
Observable??	FULL	FULL	FULL	PARTIAL
Deterministic??	YES	NO	YES	NO
Episodic??	NO	NO	NO	NO
Static??				
Discrete??				
Single-agent??				



Environment types

Static vs. dynamic: If the environment can change while the agent is choosing an action, the environment is dynamic. Semi-dynamic if the agent's performance changes even when the environment remains the same.

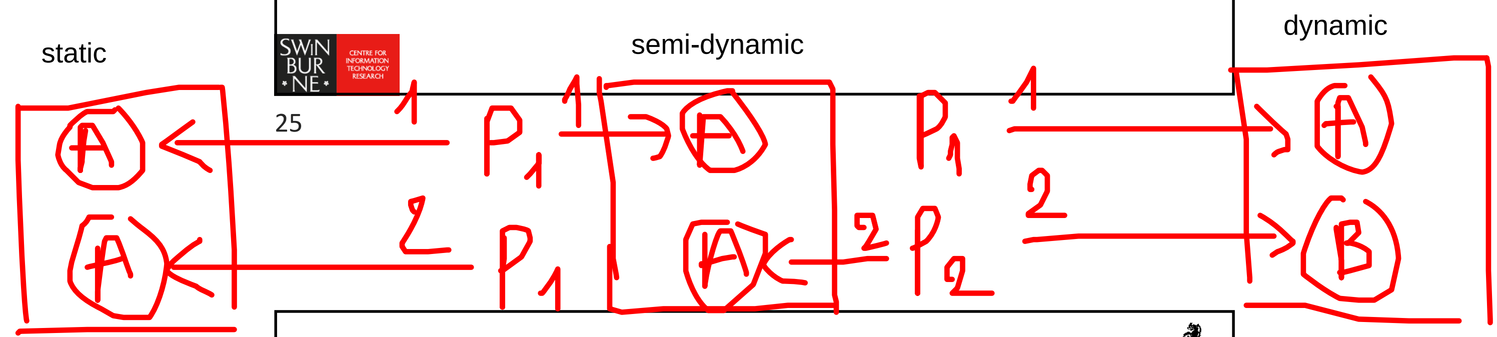
	Crossword	Backgammon	Chess w/ clock	Taxi
Observable??	FULL	FULL	FULL	PARTIAL
Deterministic??	YES	NO	YES	NO
Episodic??	NO	NO	NO	NO
Static??				
Discrete??				
Single-agent??				

Environment types

Static vs. dynamic: If the environment can change while the agent is choosing an action, the environment is dynamic. Semi-dynamic if the agent's performance changes even when the environment remains the same.

	Crossword	Backgammon	Chess w/ clock	Taxi
Observable??	FULL	FULL	FULL	PARTIAL
Deterministic??	YES	NO	YES	NO
Episodic??	NO	NO	NO	NO
Static??	YES	YES	SEMI	NO
Discrete??				
Single-agent??				

SWIN BUR NE CENTRE FOR INFORMATION TECHNOLOGY RESEARCH



A, E, P

Environment types

Discrete vs. continuous: This distinction can be applied to the state of the environment, the way time is handled and to the percepts/actions of the agent.

	Crossword	Backgammon	Chess w/ clock	Taxi
Observable??	FULL	FULL	FULL	PARTIAL
Deterministic??	YES	NO	YES	NO
Episodic??	NO	NO	NO	NO
Static??	YES	YES	SEMI	NO
Discrete??				
Single-agent??				

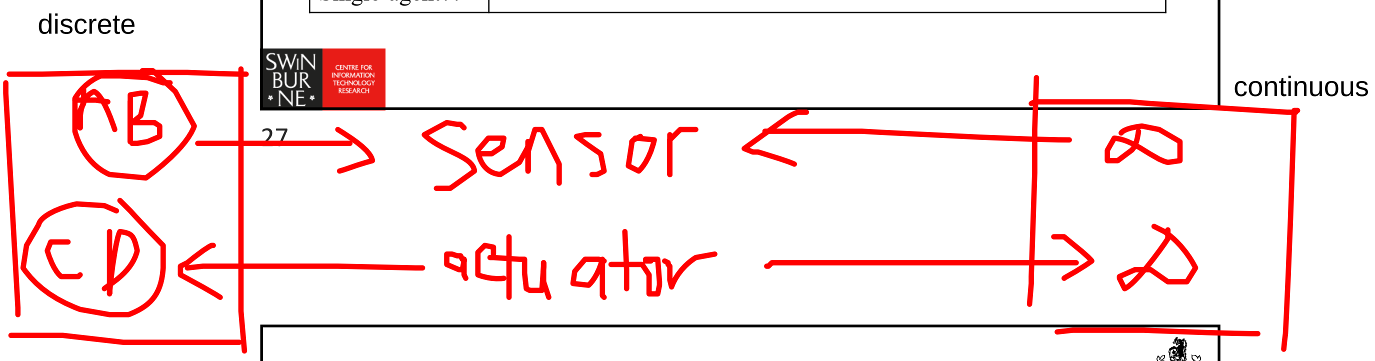
SWIN BUR NE CENTRE FOR INFORMATION TECHNOLOGY RESEARCH

Environment types

Discrete vs. continuous: This distinction can be applied to the state of the environment, the way time is handled and to the percepts/actions of the agent.

	Crossword	Backgammon	Chess w/ clock	Taxi
Observable??	FULL	FULL	FULL	PARTIAL
Deterministic??	YES	NO	YES	NO
Episodic??	NO	NO	NO	NO
Static??	YES	YES	SEMI	NO
Discrete??	YES	YES	YES	NO
Single-agent??				

27



E, S, A

Environment types

Single vs. multi-agent: Does the environment contain other agents who are also maximizing some performance measure that depends on the current agent's actions?

	Crossword	Backgammon	Chess w/ clock	Taxi
Observable??	FULL	FULL	FULL	PARTIAL
Deterministic??	YES	NO	YES	NO
Episodic??	NO	NO	NO	NO
Static??	YES	YES	SEMI	NO
Discrete??	YES	YES	YES	NO
Single-agent??				

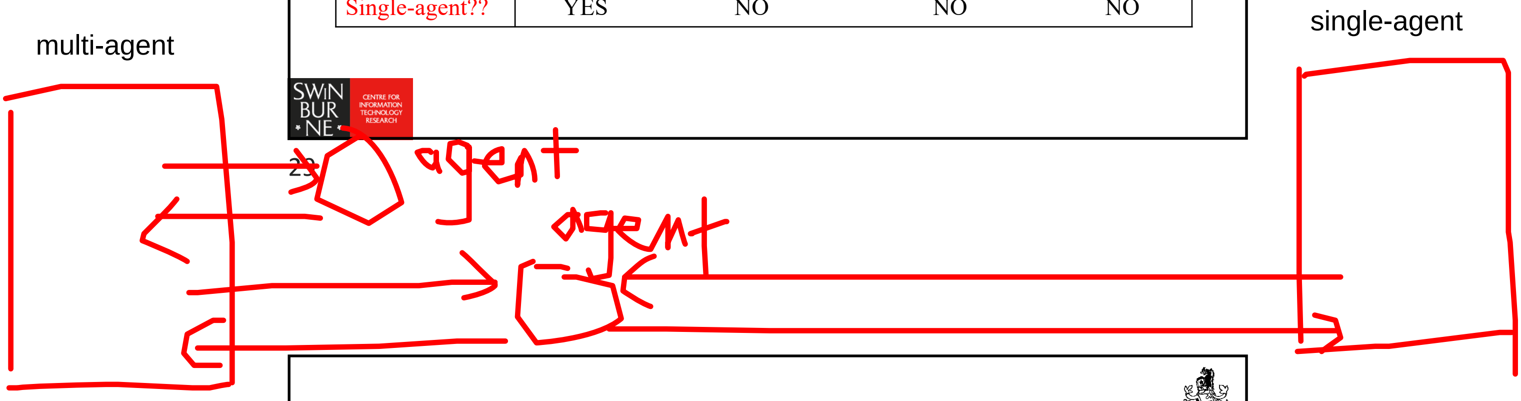
28

Environment types

Single vs. multi-agent: Does the environment contain other agents who are also maximizing some performance measure that depends on the current agent's actions?

	Crossword	Backgammon	Chess w/ clock	Taxi
Observable??	FULL	FULL	FULL	PARTIAL
Deterministic??	YES	NO	YES	NO
Episodic??	NO	NO	NO	NO
Static??	YES	YES	SEMI	NO
Discrete??	YES	YES	YES	NO
Single-agent??	YES	NO	NO	NO

23

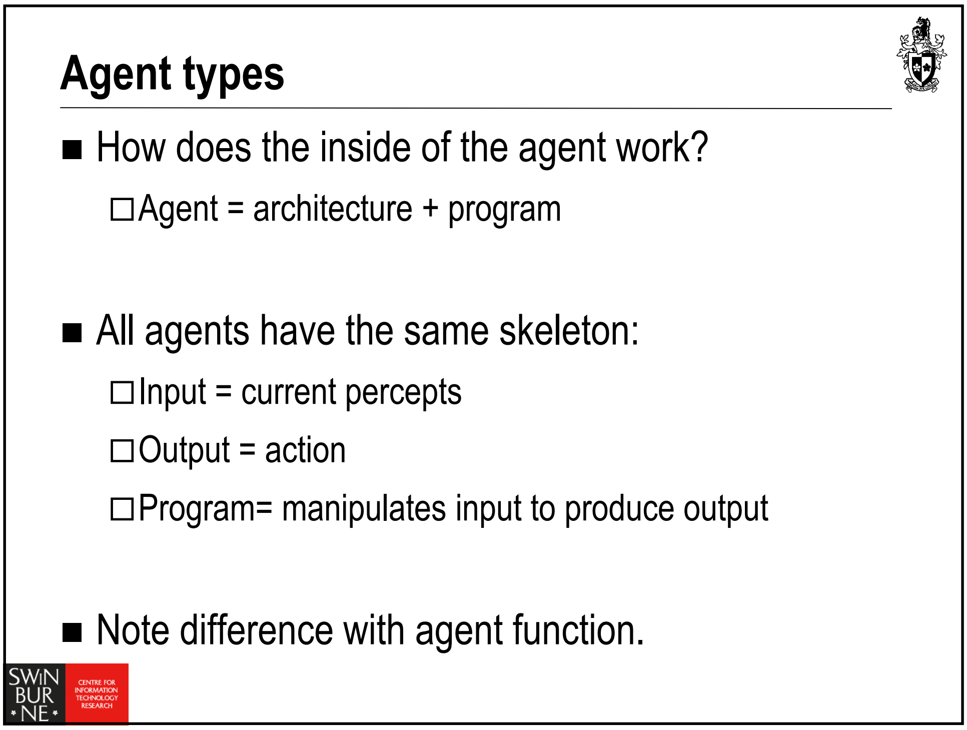


E

Environment types

- The simplest environment is
  - □Fully observable, deterministic, episodic, static, discrete and single-agent.
- Most real situations are:
  - □Partially observable, stochastic, sequential, dynamic, continuous and multi-agent.

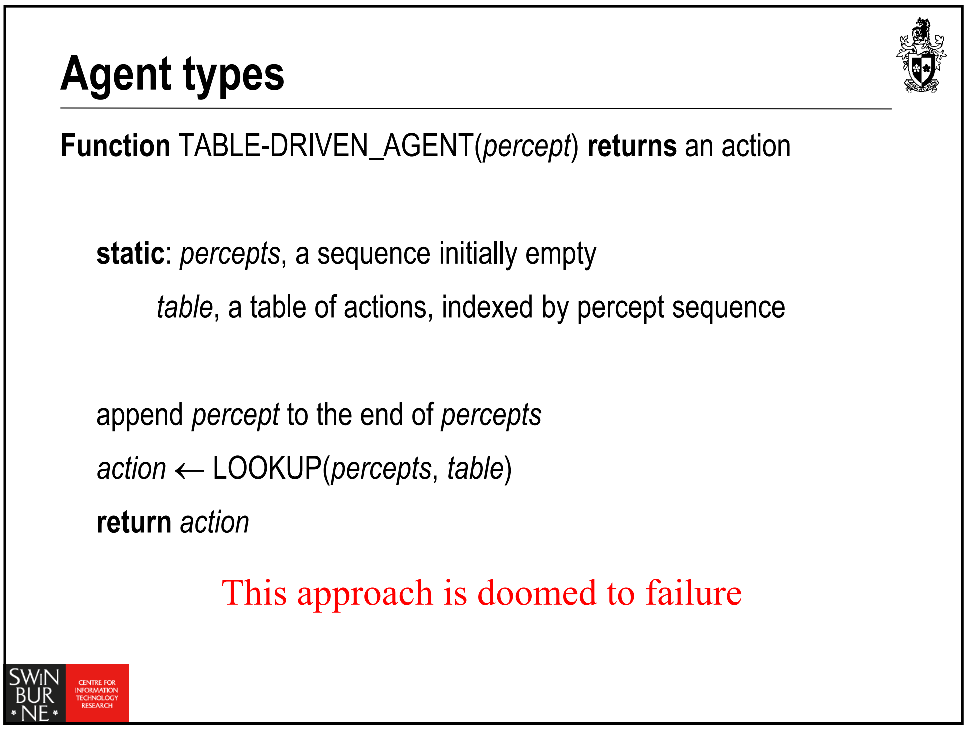
30



Agent types

- How does the inside of the agent work?
  - □Agent = architecture + program
- All agents have the same skeleton:
  - □Input = current percepts
  - □Output = action
  - □Program= manipulates input to produce output
- Note difference with agent function.

31



Agent types

```
Function TABLE-DRIVEN_AGENT(percept) returns an action

   static: percepts, a sequence initially empty
           table, a table of actions, indexed by percept sequence

   append percept to the end of percepts
   action ← LOOKUP(percepts, table)
   return action
```

This approach is doomed to failure

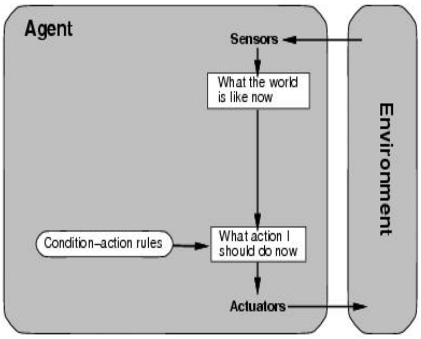
32

Agent types

- Four basic kind of agent programs will be discussed:
  - □Simple reflex agents
  - □Model-based reflex agents
  - □Goal-based agents
  - □Utility-based agents
- All these can be turned into learning agents.
  - □ And that gives you four additional advanced agent types

33

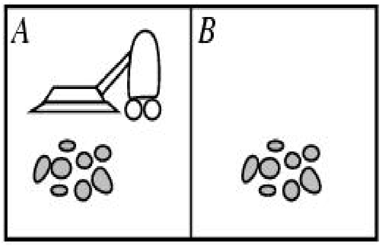
Agent types; simple reflex



- Select action on the basis of *only the current* percept.
  - □E.g. the vacuum-agent
- Large reduction in possible percept/action situations(next page).
- Implemented through *condition-action rules*
  - □If dirty then suck

34

The vacuum-cleaner world



```
function REFLEX-VACUUM-AGENT ([location, status]) return an action
    if status == Dirty then return Suck
    else if location == A then return Right
    else if location == B then return Left
```

Reduction from 4^T to 4 entries

Agent types; simple reflex

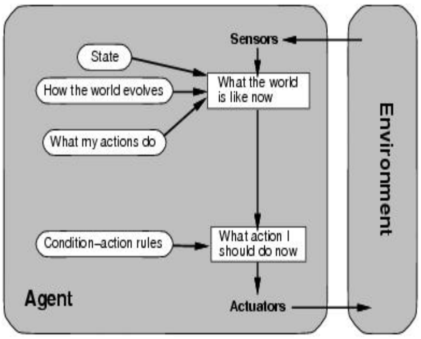
```
function SIMPLE-REFLEX-AGENT(percept) returns an action

    static: rules, a set of condition-action rules

    state ← INTERPRET-INPUT(percept)
    rule ← RULE-MATCH(state, rules)
    action ← RULE-ACTION[rule]
    return action
```

Will only work if the environment is *fully observable* otherwise infinite loops may occur.

Agent types; reflex and state



- To tackle *partially observable* environments.
  - □ Maintain internal state
- Over time update state using world knowledge
  - □ How does the world change.
  - □ How do actions affect world.
  - ⇒ *Model of World*

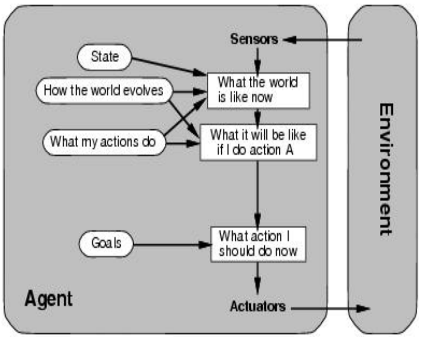
37

Agent types; reflex and state

```
function REFLEX-AGENT-WITH-STATE(percept) returns an
   action
   static: rules, a set of condition-action rules
         state, a description of the current world state
         actions, the most recent actions.
   state ← UPDATE-STATE(state, actions, percept)
   rule ← RULE-MATCH(state, rules)
   action ← RULE-ACTION[rule]
   return action
```

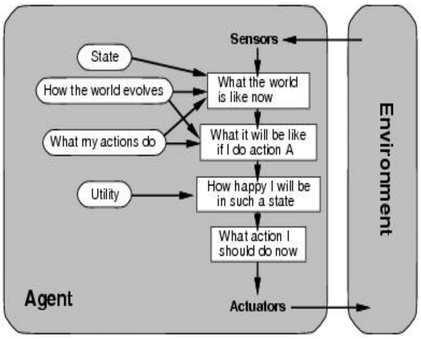
38

Agent types; goal-based


Sensors
State
How the world evolves
What my actions do
What the world is like now
What it will be like if I do action A
Goals
What action I should do now
Agent
Actuators
Environment

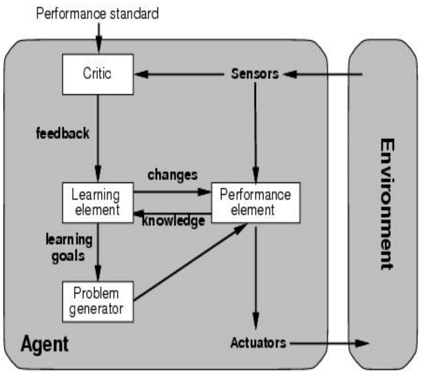
- The agent needs a goal to know which situations are *desirable*.
  - □ Things become difficult when long sequences of actions are required to find the goal.
- Typically investigated in **search** and **planning** research.
- Major difference: future is taken into account
- Is more flexible since knowledge is represented explicitly and can be manipulated.

Agent types; utility-based


Sensors
State
How the world evolves
What my actions do
What the world is like now
What it will be like if I do action A
Utility
How happy I will be in such a state
What action I should do now
Agent
Actuators
Environment

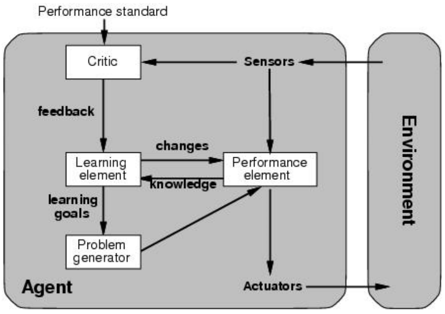
- Certain goals can be reached in different ways.
  - □ Some are better, have a higher utility.
- Utility function maps a (sequence of) state(s) onto a real number.
- Improves on goals:
  - □ Selecting between conflicting goals
  - □ Select appropriately between several goals based on likelihood of success.

Agent types; learning

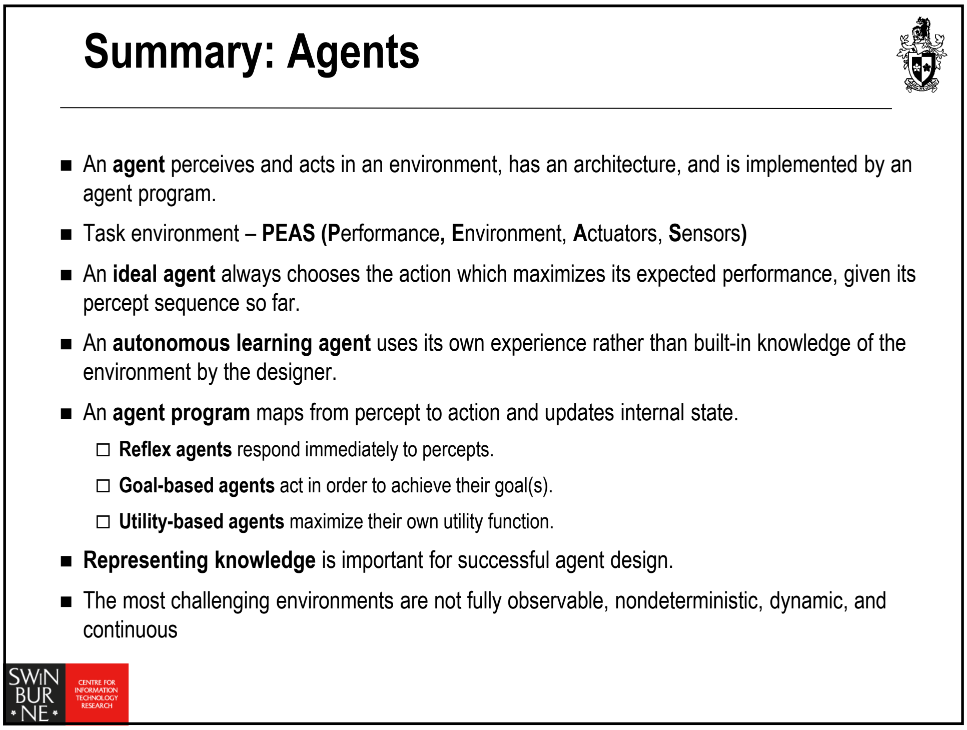


- All previous agent-programs describe methods for selecting *actions*.
  - □ Yet it does not explain the origin of these programs.
  - □ Learning mechanisms can be used to perform this task.
  - □ Teach them instead of instructing them.
  - □ Advantage is the robustness of the program toward initially unknown environments.

Agent types; learning



- *Learning element*: introduce improvements in performance element.
  - □ Critic provides feedback on agents performance based on fixed performance standard.
- *Performance element*: selecting actions based on percepts.
  - □ Corresponds to the previous agent programs
- *Problem generator*: suggests actions that will lead to new and informative experiences.
  - □ Exploration vs. exploitation



Summary: Agents

- An **agent** perceives and acts in an environment, has an architecture, and is implemented by an agent program.
- Task environment – **PEAS (P**erformance**, E**nvironment, **A**ctuators, **S**ensors**)**
- An **ideal agent** always chooses the action which maximizes its expected performance, given its percept sequence so far.
- An **autonomous learning agent** uses its own experience rather than built-in knowledge of the environment by the designer.
- An **agent program** maps from percept to action and updates internal state.
  - □ **Reflex agents** respond immediately to percepts.
  - □ **Goal-based agents** act in order to achieve their goal(s).
  - □ **Utility-based agents** maximize their own utility function.
- **Representing knowledge** is important for successful agent design.
- The most challenging environments are not fully observable, nondeterministic, dynamic, and continuous