Artificial Intelligence and Evolutionary Computing

Lecture 1

IT/T/414A
Winter Semester 2014

Instructor: Kartick Ch. MONDAL

Course Content

Part I:

Introduction to Production systems.
Search, Heuristic search, A* Algorithm, AND/OR Graph, AO* algorithm.
Knowledge representation using Predicate Calculus; Resolution and Theorem Proving.

Part II:

Introduction to Logic Programming Language.
Forward and Backward Search.

Part III:

Genetic algorithms (Gas), evolution strategies (Ess), evolutionary programming (EP), genetic programming (GP); selection, crossover, mutation; schema analysis; analysis of selection algorithms; convergence;

Part IV:

Markov and other stochastic models; classifier systems; constraint handling; multi-objective and multimodal optimization.

Part V:

Feed forward and feedback (recurrent) networks and hybrid learning algorithms; multi layer perceptron and back propagation learning algorithm.

Artificial Intelligent:
•Concerned with <u>intelligent behavior</u> in artifacts

Intelligent behavior:
•It involves <u>perception</u>, <u>reasoning</u>, learning, communicating and acting in complex environment

Aims:
•Developing machines having intelligent behavior as well as human can

<u>**What can makes a machine to think more like human?**</u>

•Brain can think parallely whereas conventional computers do it serially.
•So, building powerful parallel computers can make progress

•Conventional computing machinery is based on binary logic (true or false)
•We have to use some sort of <u>fuzzy logic</u>

•Basic building blocks of computers are switches but animal neurons are much more complex
•We have to build realistic artificial neurons

Approaches to AI

1)Symbol processing approaches

•e.g.: logical operation used in declarative knowledge bases
•Also called knowledge based approach
•Represent knowledge of a problem domain by declarative sentences.
•Declarative sentences are equivalent to first order logic
•Then logical reasoning methods are used to deduce consequences of this knowledge
•This kind of systems needs substantial knowledge of the domain

•Also known as Top-down approach
•Top level is knowledge level
•Then Symbol level where knowledge are represented as symbols and operations are specified
•Next, the lower level where the symbol processing operations are performed

<u>Approaches to AI</u>

1)Symbol processing approaches

2)Subsymbolic approaches

- Also known as bottom-up approach
- At the lowest levels, the concept of symbol is not as appropriate as signal.
- Here the aim is
- To duplicate the ability of signal processing and
- To duplicate the ability of control systems of simpler animal
- Then proceed in step up the evolutionary ladder

- Example of such approach is <u>neural networks</u>

1)Hybrid approach

- Intermediate between top down and bottom up approach
- Based on <u>situated automata</u>

Grid Space World

- A three dimensional space demarcated by two dimensional grid of cells
- This grid is called floor
- Each cell may contain objects having various properties
- There may be wall like boundaries between sets of cells
- The agents are confined to the floor and may move from cell to cell
- Other 'word' used in AI literature is Blocks world, Tile world, Wumpus world and Ant world

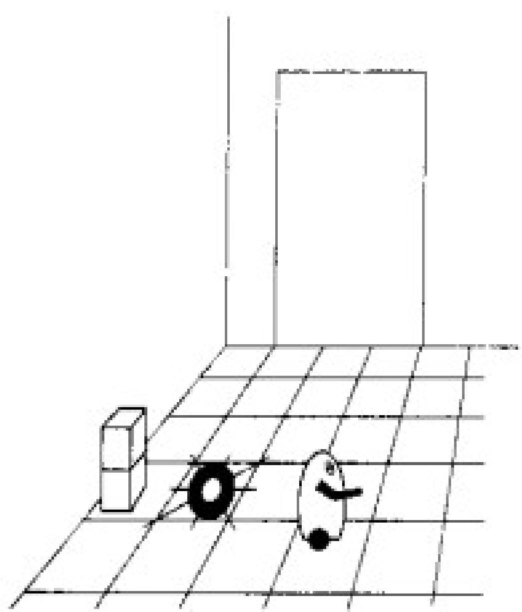


Figure 1.2

Grid-Space World

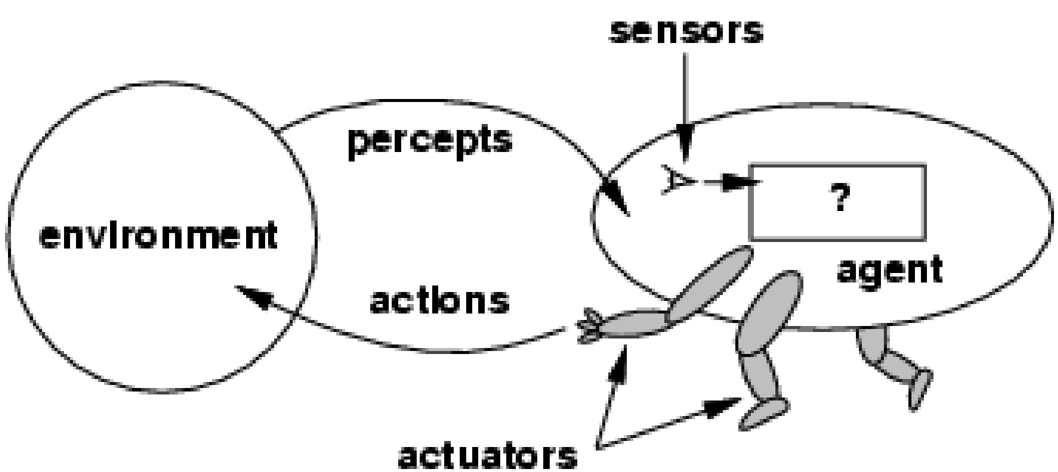
An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators

Human agent:

eyes, ears, and other organs for sensors;
hands, legs, mouth, and other body parts for actuators

Robotic agent:

cameras and infrared range finders for sensors;
various motors for actuators



•Agents can sense the world and acting in them
•Can manipulate objects present on the floor
•Have the ability to remember different properties
•Store internal <u>models of the world</u>

Different types of Agent based on generated responses:

•<u>Stimulus-response Agent:</u>
•Have no internal states
•Reacts to immediate <u>stimuli</u> in their environment

•<u>Reactive Agents:</u>
•Actions taken by these agents are functions of the current and past states of their world
•It can have complex perceptual and motor processes

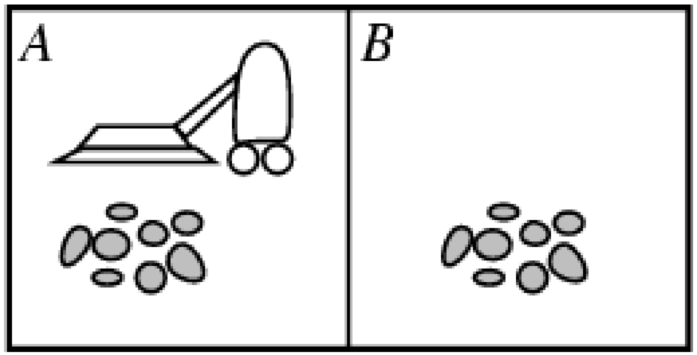
• The agent function maps from percept histories to actions:
[*f* : P* ->A]

• The agent program runs on the physical architecture to produce *f*

agent = architecture + program

• Percepts: location and state of the environment, e.g., [A,Dirty], [B,Dirty]

• Actions: *Left*, *Right*, *Suck*, *NoOp*

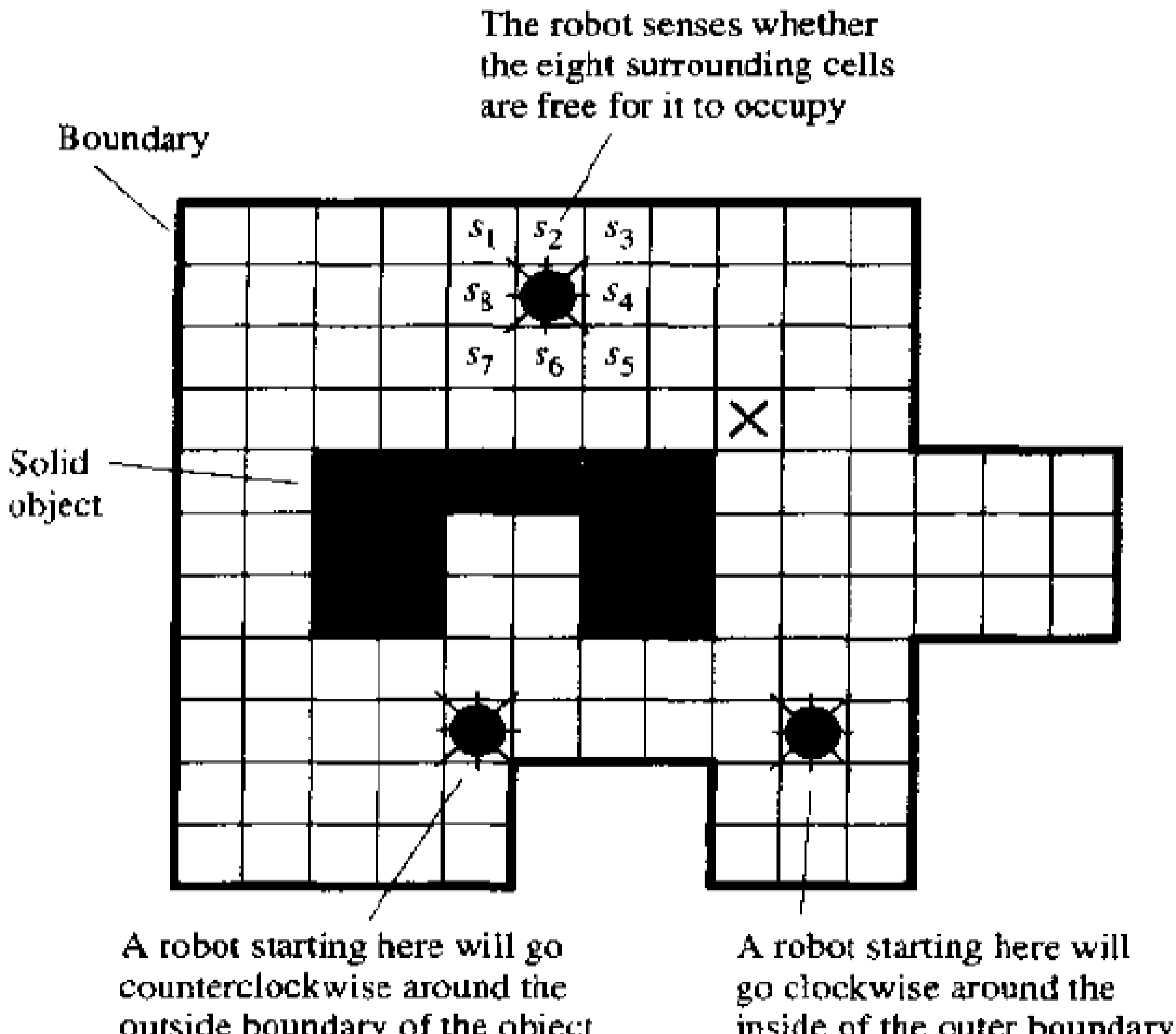


Models or representations of the world:

- Iconic Model
- Involves data structure and computations
- Simulate aspects of agent's environment
- Also simulate the effects of agent actions upon the environment
- Representing the state of play in chess by 8x8 arrays of cells is an example of this representation

- Feature based model
- Uses declarative descriptions of environments
- This kind of representation is incomplete

Robots in two dimensional grid world



The aim:

- Go to a cell adjacent to a boundary or object
- Then follow that boundary along its perimeter forever

The task:

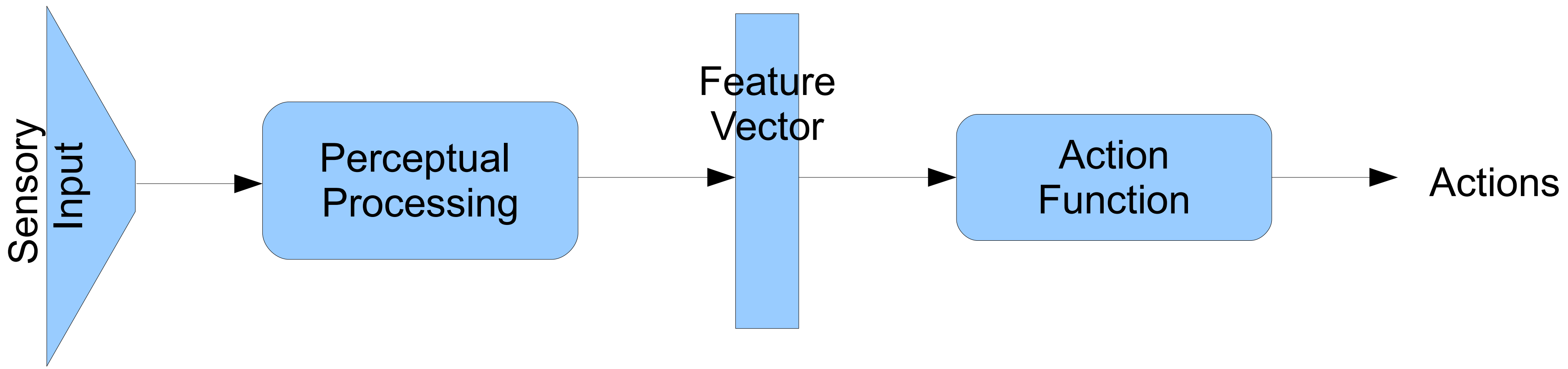
- Sense whether or not certain cells are free to occupy
- Perform certain primitive actions

- The eight sensory inputs are denoted by binary valued variables s1, s2, s3, s4, s5, s6, s7 and s8
- Can be 0 if the corresponding cell can be occupied by the robot
- 1 if the corresponding cell is not empty
- The value of the sensory input for the position x would be (0,0,0,0,0,0,1,0)

Robot can move to adjacent free cell along its column or row by using four actions in the cellular grid:

- North moves the robot one cell up
- East moves the robot one cell right
- South moves the robot one cell down
- West moves the robot one cell left

The designers' job is to specify a function of the sensory inputs that selects an appropriate action for the task.



- Feature Vector can be of
  - Numerical Feature
  - Binary Feature
  - Categorical Feature

Is Binary feature and categorical feature are same? If yes, then when? If no, then why?

The features are selected by the designer

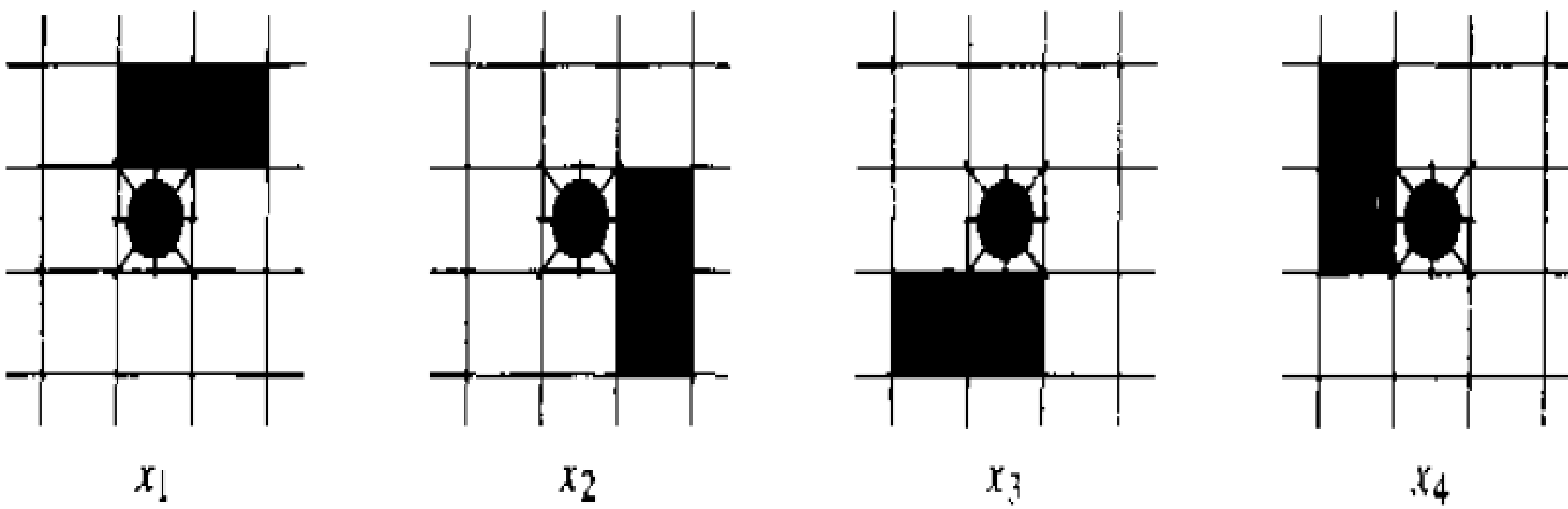
To

Correlate with those properties of the robot's environment

Those are

Relevant to which action should be performed in the state described by the features.

- Perception
  - There are four sensory input of the boundary following robot s1, s2, s3, s4, s5, s6, s7 and s8
  - $2^8 = 256$ combinations are possible for feature vectors
  - The four binary valued features of the sensory values that are useful for computing an appropriate action are x1, x2, x3 and x4
  - E.g., x1 = 1, iff, s2=1 or s3=1
- Features for boundary Following Task



In each diagram, the indicated feature has value 1 if and only if at least one of the shaded cells is *not* free.

- Disadvantages of perception based processing system:
  - In many real world problem, perceptual processing might occasionally give <u>erroneous</u>, <u>ambiguous</u> or incomplete information.
  - This kind of perception might evoke inappropriate action.

Action

- After having generated the feature vectors, we have to specify a function that selects an appropriate action for the boundary following task.
- This function is called Action Function
- If none of the features (x1, x2, x3 & x4) has value 1 (all of its surrounding cells are free), the robot can move in any direction until it encounters a boundary.
- In such case, let move the robot to north till one of the feature has value 1.
- Now the boundary following behavior will be achieved by the following rules of action
  - If x1=1 and x2=0, move east
  - If x2=1 and x3=0, move south
  - If x3=1 and x4=0, move west
  - If x4=1 and x1=0, move north

Action Function

- If there are R possible actions (4 in case of the example), then we must find an appropriate R-valued function of the feature vector to compute an action.
- There are various ways of representations and implementations are available for action functions.
- One such convenient representational form for an action function is a production system.

Production System

- A production system comprises an ordered list of rules called production rules or productions.
- Each rule has a conditional part and an action part.
- The production rules are written in the form of C -> A where C is the conditional part and A is the action part
- Condition part of a rule can be any binary-valued (0, 1) function of the features resulting from perceptual processing of the sensory input
  - e.g. If x1=1 and x2=0 then move east.
- The conditional part can be a monomial or a conjunction of Boolean Literals

Working Flow

- To select an action in the production system by using action function, the rules are processed as follows:
  1. Start with the first rule in the ordering
  2. Search for the first rule where the conditional part evaluates to 1 and select the action part of that rule.
  3. The action function can be of several types:
     1. A primitive action
     2. A call to another production system
     3. A set of actions to be executed simultaneously
  4. Usually, the last rule in the ordering has 1 as the conditional part
  5. If no rule above the last rule has a condition part equal to 1, then the action associated with the last rule is executed by default.

- As actions are executed, sensory inputs and the values of features based on them change.
- We assume for the example that,
  - The conditions are continuously being checked so that the action being executed at any time corresponds to the first rule whose condition has value 1.
- For the boundary following task, the production system representation is:
  - $$\begin{aligned} x_4\overline{x_1} &\longrightarrow \text{north} \\ x_3\overline{x_4} &\longrightarrow \text{west} \\ x_2\overline{x_3} &\longrightarrow \text{south} \\ x_1\overline{x_2} &\longrightarrow \text{east} \\ 1 &\longrightarrow \text{north} \end{aligned}$$

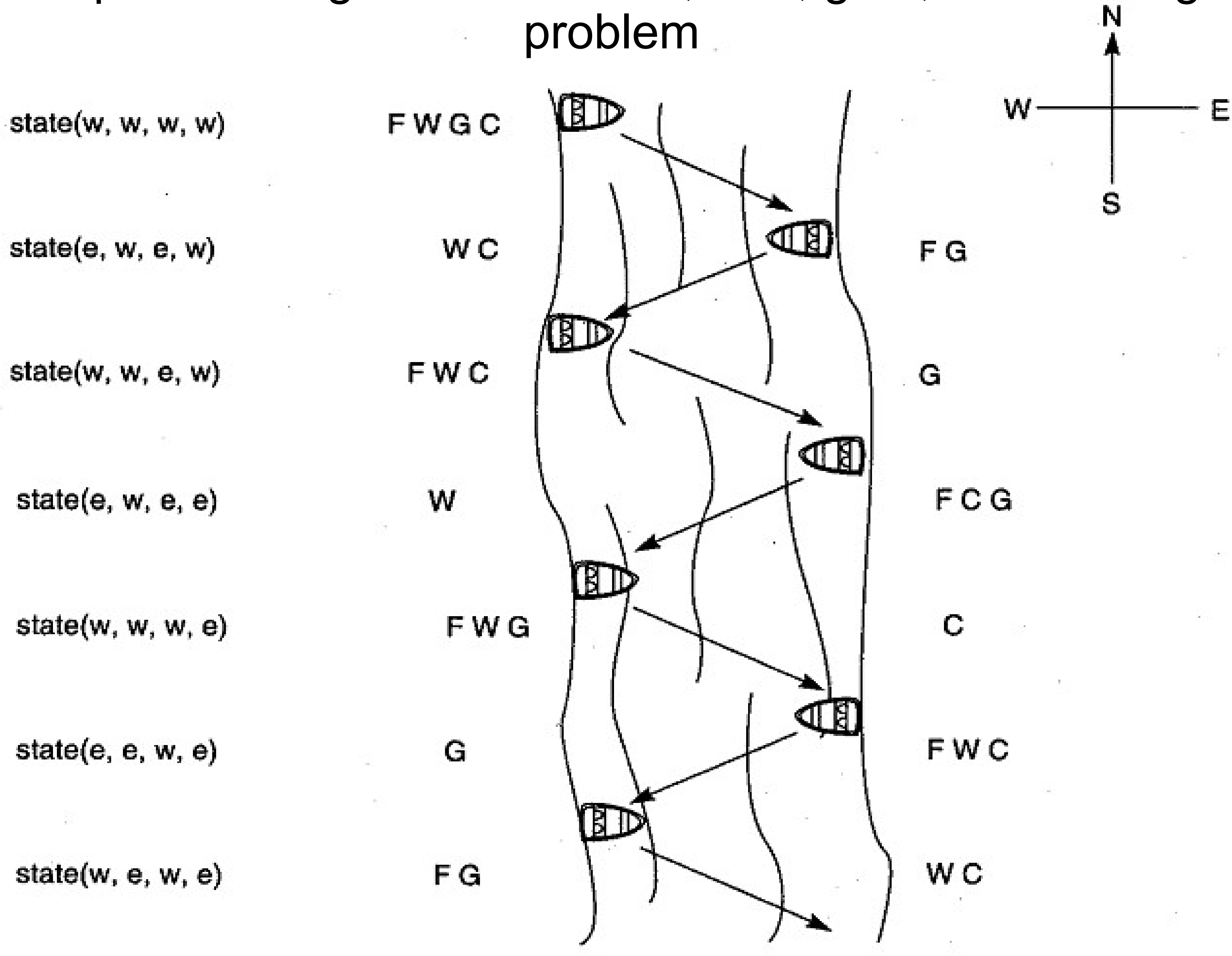
- This boundary following behavior is an example of a <u>durative procedure</u>
- One example of <u>goal driven procedure</u> is instead of following boundary task, we might want it to go to a corner (concave) and stay there.
  - In this case, a corner detecting feature should be present where the condition part will be 1 if the robot is in a corner.

  - Nil is the null or does nothing option and b-f is the boundary following procedure.
    - C -> Nil
    - 1 -> B-F (Another production system)

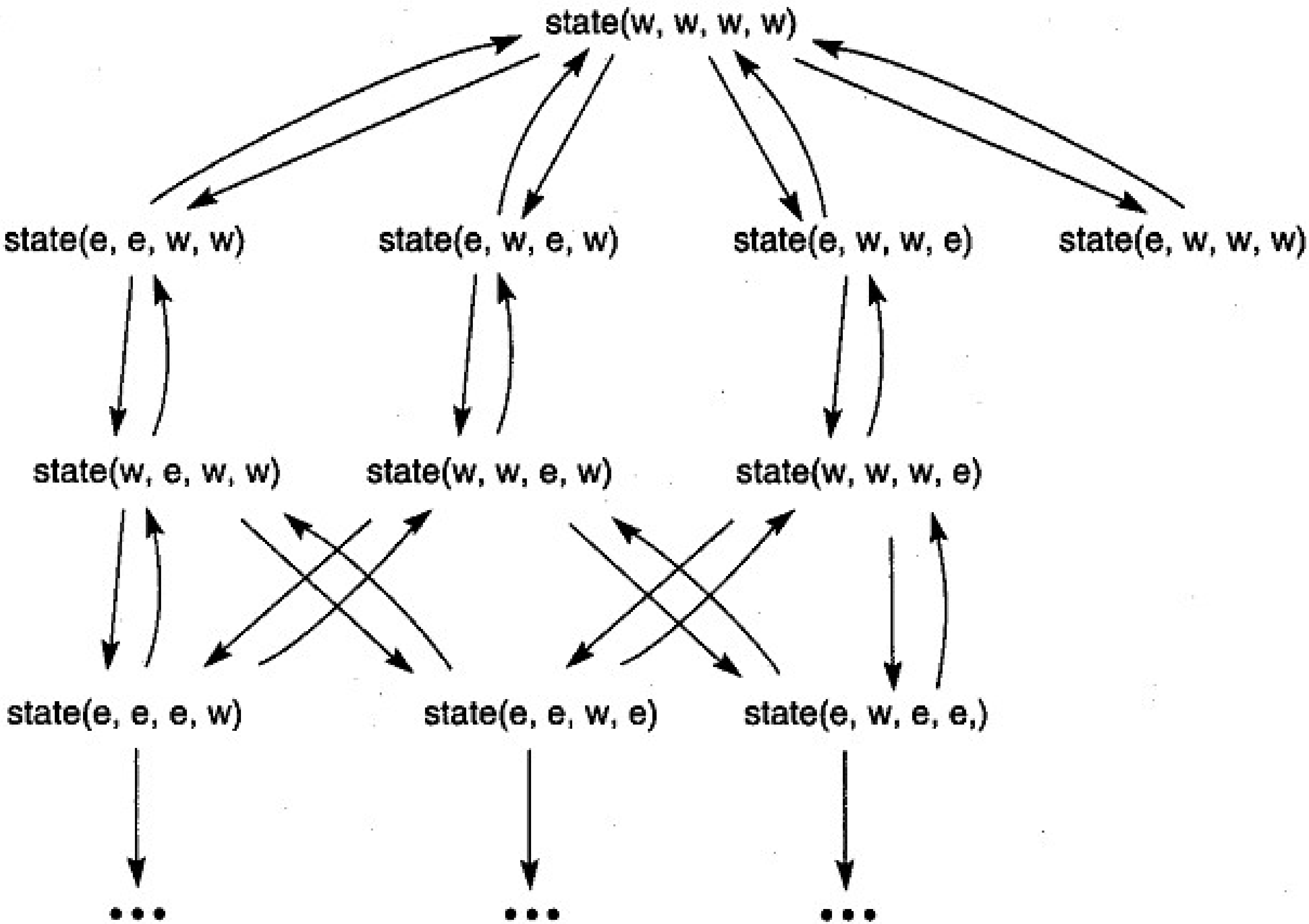
Example of a production system

- A farmer with his wolf, goat, and cabbage come to the edge of a river they wish to cross.
- There is a boat at the river's edge, but, of course, only the farmer can row.
- Also, the boat can carry only the two things at a time including the rower.
- If the wolf is ever left with the goat, the wolf will eat the goat.
- Similarly, if the goat is left alone with cabbage, the goat will eat the cabbage.
- Devise a sequence of crossing of the river so that all four characters arrive safely on the other side of the river.

Sample crossings for the farmer, wolf, goat, and cabbage problem



Portion of the state space graph of the farmer, wolf, goat, and cabbage problem (including unsafe states)



Exercise of a production system

- A discrete elevator can sense the following information about its world:
    - What floor the elevator is stopped at.
    - What floors passengers in the elevator want to go to.
    - What floors passengers outside of the elevator want rides from and whether they want to go up or down.
    - The status of the elevator door (open or closed).

The elevator is capable of performing the following actions:

- Go up exactly one floor (unless it is already at the top floor).
- Go down exactly one floor (unless it is already at the bottom floor).
- Open the elevator door
- Close the elevator door
- Wait delta seconds (a fixed time sufficient for all in the elevator to get off and for all outside the elevator to get in).

Design a production system to control the elevator in an efficient manner. (It is not efficient, for example, to reverse the elevator direction from going up to going down either if there is someone still inside the elevator who wants to go to a higher floor or if there is someone outside the elevator who wants to get on from a higher floor).