DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

SEARCHING WITH PARTIAL INFORMATION

- Different types of incompleteness lead to three distinct problem types:

Sensorless problems (conformant): If the agent has no sensors at all

- **Contingency problem**: if the environment if partially observable or if action are uncertain (adversarial)
- **Exploration problems**: When the states and actions of the environment ar no e unknown.
- No sensor
- Initial State(1,2,3,4,5,6,7,8)
- After action [Right] the state (2,4,6,8)
- After action [Suck] the state (4, 8)
- After action [Left] the state (3,7)
- After action [Suck] the state (8)
- Answer : [Right, Suck, Left, Suck] coerce the world into state 7 without any sensor

Belief State: Such state that agent belief to be there

Partial knowledge of states and actions:

- *sensorless or conformant problem*
  - Agent may have no idea where it is; solution (if any) is a sequence.
- *contingency problem*
  - Percepts provide *new* information about current state; solution is a tree or policy; often interleave search and execution.
  - If uncertainty is caused by actions of another agent: *adversarial problem*
- *exploration problem*
  - When states and actions of the environment are unknown.

Contingency, start in {1,3}.

Murphy's law, Suck *can* dirty a clean carpet. Local sensing: dirt, location only. − Percept = [L,Dirty] ={1,3}

– [Suck] = {5,7}

– [Right] ={6,8}

− [Suck] in {6}={8} (Success)

− BUT [Suck] in {8} = failure Solution??

− Belief-state: no fixed action sequence guarantees solution

Relax requirement:

− [*Suck*, *Right*, if *[R,dirty]* then *Suck*]

− Select actions based on contingencies arising during execution.

Time and space complexity are always considered with respect to some measure of the problem difficulty. In theoretical computer science, the typical measure is the size of the state space.

In AI, where the graph is represented implicitly by the initial state and successor function, the complexity is expressed in terms of three quantities:

b, the **branching factor** or maximum number of successors of any node; **d**, the **depth** of the **shallowest goal node**; and **m**, the **maximum length** of any path in the state space.

Search-cost - typically depends upon the time complexity but can also include the term for memory usage.

Total–cost – It combines the search-cost and the path cost of the solution found. **2.15 INFORMED SEARCH AND EXPLORATION**

Informed (Heuristic) Search Strategies

Informed search strategy is one that uses problem-specific knowledge beyond the definition of the problem itself. It can find solutions more efficiently than uninformed strategy.

Best-first search

Best-first search is an instance of general TREE-SEARCH or GRAPH-SEARCH algorithm in which a node is selected for expansion based on an **evaluation function** f(n). The node with lowest evaluation is selected for expansion, because the evaluation measures the distance to the goal.

This can be implemented using a priority-queue, a data structure that will maintain the fringe in ascending order of f-values.

2.16 HEURISTIC FUNCTIONS

A **heuristic function** or simply a **heuristic** is a function that ranks alternatives in various search algorithms at each branching step basing on an available information in order to make a decision which branch is to be followed during a search.

The key component of Best-first search algorithm is a **heuristic function**, denoted by h(n): h(n) = estimated cost of the **cheapest path** from node n to a **goal node**.

For example, in Romania, one might estimate the cost of the cheapest path from Arad to Bucharest via a **straight-line distance** from Arad to Bucharest (Figure 2.19).

Heuristic function are the most common form in which additional knowledge is imparted to the search algorithm.

Greedy Best-first search

Greedy best-first search tries to expand the node that is closest to the goal, on the grounds that this is likely to a solution quickly.

It evaluates the nodes by using the heuristic function f(n) = h(n).

Taking the example of **Route-finding problems** in Romania, the goal is to reach Bucharest starting from the city Arad. We need to know the straight-line distances to Bucharest from various cities as shown in Figure. For example, the initialstate is In(Arad),and the straight line distance heuristic hSLD (In(Arad)) is found to be 366.

Using the **straight-line distance** heuristic **hSLD,** the goal state can be reached faster.

Figure shows the progress of greedy best-first search using hSLD to find a path from Arad to Bucharest. The first node to be expanded from Arad will be Sibiu, because it is closer to Bucharest than either Zerind or Timisoara. The next node to be expanded will be Fagaras, because it is closest. Fagaras in turn generates Bucharest, which is the goal.

Properties of greedy search

- **Complete:** No–can get stuck in loops, e.g., Iasi !Neamt !Iasi !Neamt !

  Complete in finite space with repeated-state checking

- **Time:** O(bm), but a good heuristic can give dramatic improvement
- **Space:** O(bm) - keeps all nodes in memory
- Optimal: No

Greedy best-first search is not optimal, and it is incomplete.

The worst-case time and space complexity is $O(b^m)$,where m is the maximum depth of the search space.

A* SEARCH

A* Search is the most widely used form of best-first search. The evaluation function f(n) is obtained by combining

(1) **g(n) =** the cost to reach the node, and

(2) **h(n) =** the cost to get from the node to the **goal** :

$$f(n) = g(n) + h(n).$$

A^* Search is both optimal and complete. $A^*$is optimal if h(n) is an admissible heuristic. The obvious example of admissible heuristic is the straight-line distance hSLD. It cannot be an overestimate.

A* Search is optimal if h(n) is an admissible heuristic – that is, provided that h(n) never overestimates the cost to reach the goal.

An obvious example of an admissible heuristic is the straight-line distance hSLD that we used in getting to Bucharest. The progress of an A*tree search for Bucharest is shown in Figure

The values of ‘g ‘ are computed from the step costsshown in the Romania map(figure). Also the values of hSLD are given in Figure