Write short note on Inform search.

* Informed searching techniques build upon basic uninformed search methods by incorporating additional information about the problem domain.
* The key idea is to enhance the intelligence and efficiency of the search process by leveraging knowledge of the state space.
* An evaluation function is created for each state, gauging the desirability of expanding that state to progress toward the goal.
* "Informed Search" strategies utilize this evaluation function to choose the next state for exploration.
* This approach proves more efficient in terms of time and space requirements compared to uninformed search techniques.
* Essentially, informed searching techniques employ domain-specific knowledge to guide the search process intelligently and achieve better performance.

What is heuristic function and what r its qualities?

* A heuristic function is an evaluation function, to which the search state is given as input and it generates the tangible representation of the state as output.
* The function looks at each point and estimates how likely it is to be on the right track to the solution.
* A well-designed heuristic helps the search process find a good solution more quickly.
* Heuristic function can be of two types depending on the problem domain. It can be a **Maximization Function** or **Minimization function** of the path cost.
* But designing it is a balancing act – it shouldn't be too complex to figure out or it might slow down the search.
* The heuristic function that we are considering in this syllabus, for a node n is, h(n) = estimated cost of the cheapest path from the state at node n to a goal state.

Qualities:

* It should quickly finds the solution, reducing time and computational resources
* Easy to understand and apply, suitable for real world complexities
* It should reduce the search time; specifically for hard problems like travelling salesman problem where the time
* required is exponential
* Provides accurate results without guarantying optimality
* Allows learning based on feedback for continuous improvement
* Versatile and adaptable to various problem domains

**World block problem:**

In this case, the goal state is a configuration where all blocks are in their designated positions. The heuristic function provides an estimate of how far the current state is from the goal state by counting the blocks that are not yet in their correct places.

For instance, let's say you have a block world with five blocks (A, B, C, D, E), and the goal state is for them to be arranged in alphabetical order from left to right (A, B, C, D, E). If the current state is ABCDE, the heuristic value would be 0 because all blocks are in their correct positions. If the current state is ACBDE, the heuristic value would be 1 because block B is not in its final position.

This heuristic is admissible because it never overestimates the cost to reach the goal state – moving a block to its correct position can only reduce the count. However, it might not be very informative for all scenarios, and more sophisticated heuristics could be designed based on the specifics of the problem or the characteristics of the blocks involved.

Block World Configuration:

Let's consider a block world with five blocks labeled A, B, C, D, and E. The goal is to arrange these blocks in alphabetical order from left to right (A, B, C, D, E).

Example State:

Suppose the current state of the block world is: ACBED.

Heuristic Function:

The heuristic function counts the number of blocks that are not in their final positions. In this case, we evaluate how many blocks are not in alphabetical order.

Calculation:

A is in the correct position.

C is not in the correct position (it should be after B).

B is not in the correct position (it should be after A).

E is not in the correct position (it should be after D).

D is not in the correct position (it should be after C).

So, three blocks (C, B, E) are not in their final positions.

Heuristic Value (h):

The heuristic value for this state is 3, as there are three blocks not in their correct positions.

Interpretation:

The heuristic value of 3 suggests that, based on this heuristic, the current state is relatively far from the goal state. The higher the heuristic value, the more blocks are out of order, indicating a greater distance from the goal.

Admissibility:

This heuristic is admissible because it never overestimates the cost to reach the goal state. The actual cost to move a block to its correct position cannot be greater than the number of blocks out of place.

Keep in mind that while this heuristic is simple, it provides a basic estimate of how far the current state is from the goal. More complex heuristics could take into account the specific relationships between blocks or consider other factors for a more accurate estimation.

**Write short note on admissibility of A\*. (May 14, 5 Marks)**

* A search algorithm is admissible**,** if for any graph, it always terminates in an optimal path from initial state to goal state, if path exists. A heuristic is admissible if it never overestimates the actual cost from current state to goal state.
* Alternatively, we can say that A\* always terminates with the optimal path in case h(n) is an admissible heuristic function.
* A heuristic h(n) is admissible if for every node n, if h(n) ≤ h\*(n), where h\*(n) is the true cost to reach the goal state from n.
* An admissible heuristic never overestimates the cost to reach the goal. Admissible heuristics are by nature optimistic because they think the cost of solving the problem is less than it actually is.
* An obvious example of an admissible heuristic is the straight line distance. Straight line distance is admissible because the shortest path between any two points is a straight line, so the straight line cannot overestimate the actual road distance.

o **Theorem :** If h(n) is admissible, tree search using A\* is optimal.

o **Proof :** Optimality of A\*with admissible heuristic.

* Suppose some suboptimal goal G2 has been generated and is in the fringe. Let n be an unexpanded node in the fringe such that n is on a shortest path to an optimal goal G.

HILL CLIMBING

* Hill climbing technique is used widely in artificial intelligence, to solve computationally hard problems, which has multiple possible solutions.
* In the depth-first search, the test function will merely accept or reject a solution. But in hill climbing the test function is provided with a heuristic function which provides an estimate of how close a given state is to goal state.
* **each state** is provided with the **additional information needed to find the solution**, i.e. the **heuristic value.**
* The algorithm is **memory efficient** since it does not maintain the complete search tree.
* it looks only at the **current state and immediate level states.**
* For example, if you want to find a mall from your current location. There are n possible paths with different directions to reach to the mall. The heuristic function will just give you the distance of each path which is reaching to the mall, so that it becomes very simple and time efficient for you to reach to the mall.
* **iteratively improve the current state by means of an evaluation function**.
* **make changes that improve the current state**.
* hill climbing can **only advance if there is a higher point in the adjacent landscape.**
* **Hill climbing is a type of local search technique.**
* It is relatively simple to implement**.**
* **In many cases where state space is of moderate size, hill climbing works even better than many advanced techniques.**
* For example, hill climbing when applied to travelling salesman problem; initially it produces random combinations of solutions having all the cities visited. Then it selects the better rout by switching the order, which visits all the cities in minimum cost.

Problems:

Following are the problems that may arise in hill climbing strategy:

Sometimes the **algorithms may lead to a position, which is not a solution**, **no further state that is going closer to the solution**. This will happen if we have reached one of the following three states.

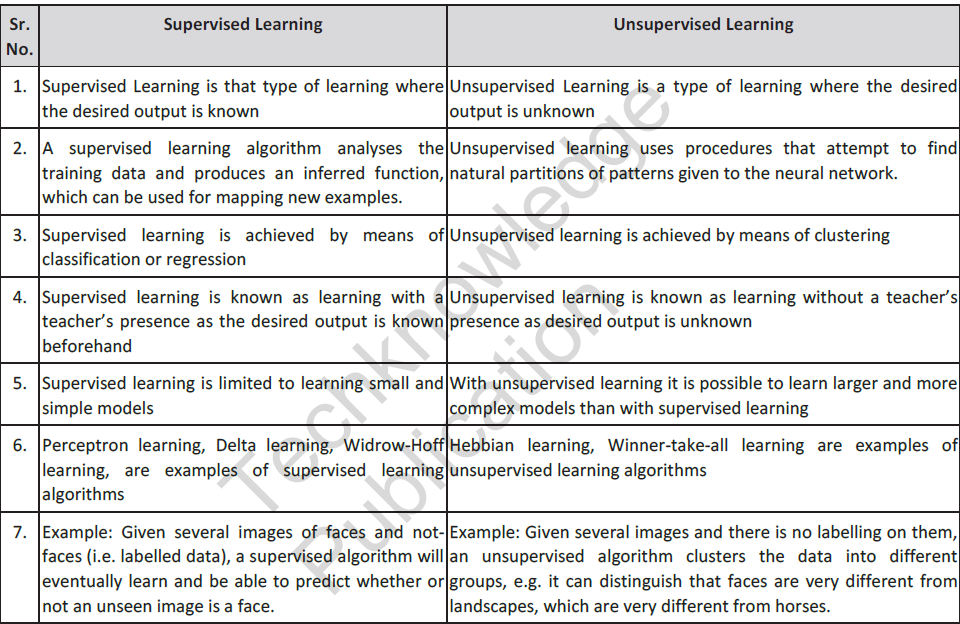
1. **Local Maximum :**

* A “local maximum” is a location in hill which is **at height from other parts of the hill but is not the actual hill top**.
* In the search tree, it is a **state better than all its neighbors**, but there **is not** **next better state** which can be chosen for further expansion.

1. **Plateau :**

* A “plateau” is a **flat area** **at some height** in hilly region.
* There is **a large area of same height** in plateau.
* In the search space, plateau situation occurs **when all the neighbouring states** **have the same value.**
* On a plateau, it is not possible to determine the best direction in which to move by making local comparisons.

1. **Ridge :** A "ridge" is an area in the hill such that, it is higher than the surrounding areas, but there is no further uphill path from ridge. In the search tree it is the situation, where all successors are either of same value or lesser, it’s a ridge condition. The suitable successor cannot be searched in a simple move.



Inductive learning, also known as discovery learning, is a process where the learner discovers rules by observing examples. This is different from deductive learning, where students are given rules that they then need to apply.



We can often work out rules for ourselves by observing examples to see if there is a pattern; to see if things regularly happen in the same way. We then try applying the rule in different situations to see if it works.

With inductive language learning, tasks are designed specifically to help guide the learner and assist them in discovering a rule.