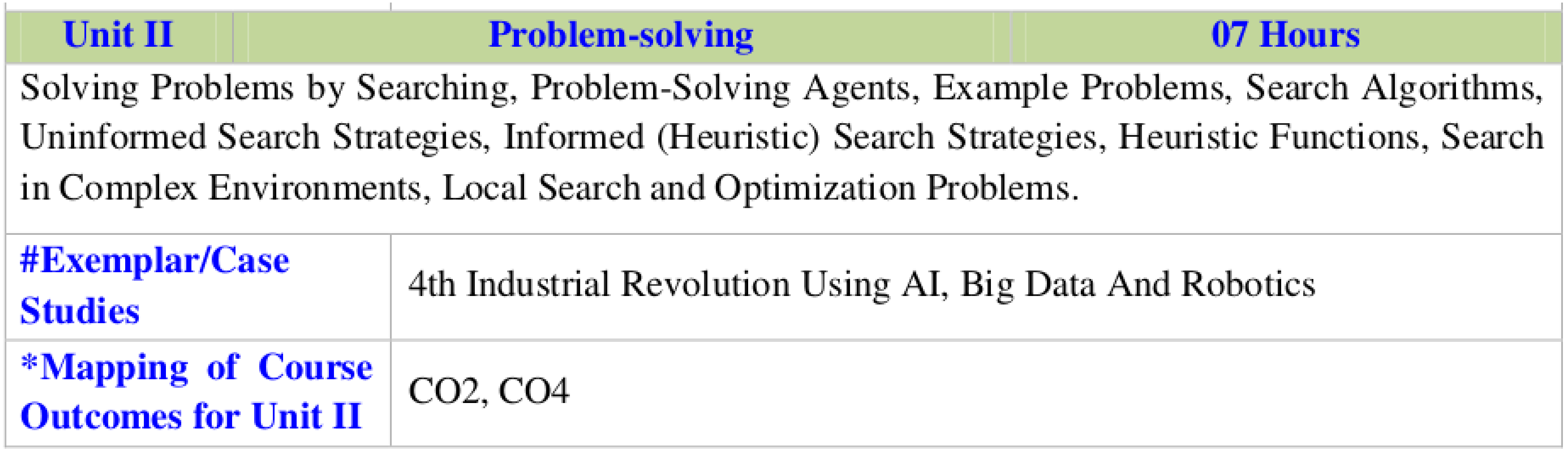
UNIT II

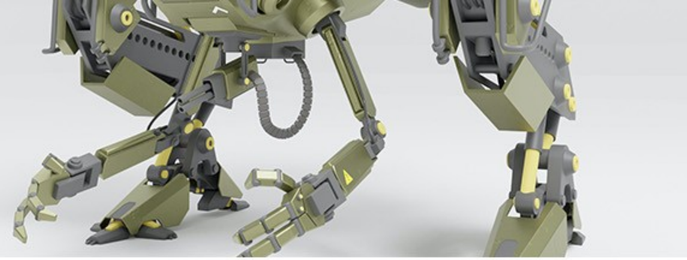
Problem Solving

Contents



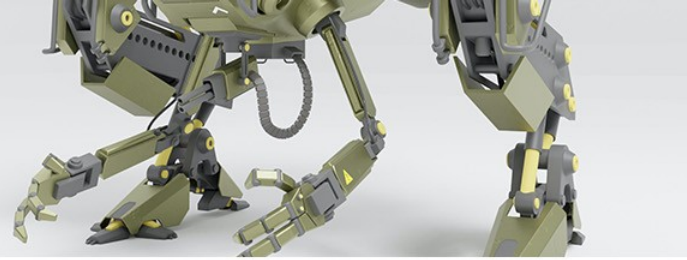
Unit II	Problem-solving	07 Hours
Solving Problems by Searching, Problem-Solving Agents, Example Problems, Search Algorithms, Uninformed Search Strategies, Informed (Heuristic) Search Strategies, Heuristic Functions, Search in Complex Environments, Local Search and Optimization Problems.		
#Exemplar/Case Studies	4th Industrial Revolution Using AI, Big Data And Robotics	
***Mapping of Course Outcomes for Unit II**	CO2, CO4	

Local Search Algorithm



- A local search algorithm in artificial intelligence is a type of optimization algorithm used to find the best solution to a problem by repeatedly making minor adjustments to an initial solution.
- Every time the algorithm iterates, the current solution is assessed, and a small modification to the current solution creates a new solution.
- The current solution is then compared to the new one, and if the new one is superior, it replaces the old one.
- This process keeps going until a satisfactory answer is discovered or a predetermined stopping criterion is satisfied.
- Hill climbing, simulated annealing, tabu search, and genetic algorithms are a few examples of different kinds of local search algorithms.

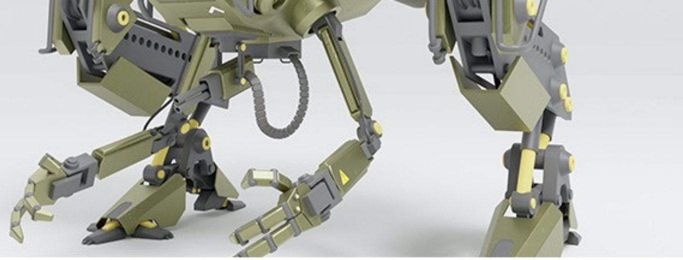
Local Search Algorithm



- Applications for local search algorithms include scheduling, routing, and resource allocation. They are particularly helpful for issues where the search space is very large and can be used to solve both discrete and continuous optimization problems.
- In contrast to multiple paths, a local search algorithm completes its task by traversing a single current node and generally following that node's neighbors.
- One of the key benefits of local search algorithms is that they can be very efficient, particularly when compared to other optimization techniques such as exhaustive search or dynamic programming.
- This is because local search algorithms only need to explore a relatively small portion of the entire search space, which can save a significant amount of time and computational resources.

Local Search Algorithm

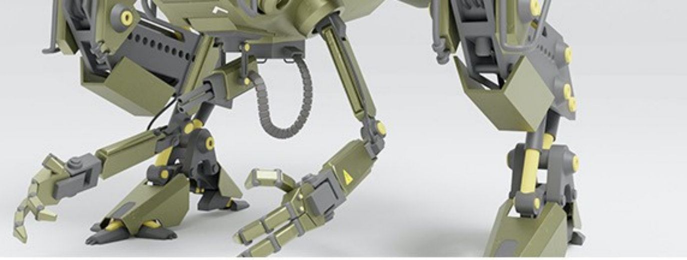
- one of the main limitations of local search algorithms is that they can become trapped in local optima, which are solutions that are better than all of their neighbors but are not the best possible solution overall.
- To overcome this limitation, many local search algorithms use various techniques such as randomization, memory, or multiple starting points to help them escape from local optima and find better solutions.



Working on a Local Search Algorithm

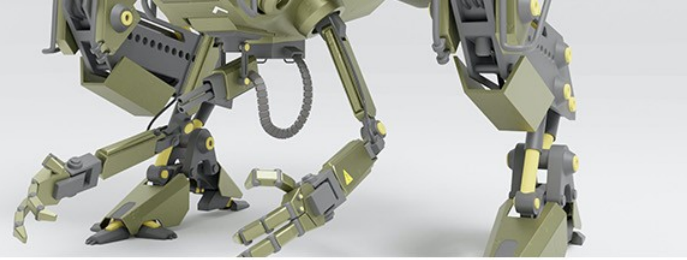
- Local search algorithms are a type of optimization algorithm that iteratively improves the solution to a problem by making small, local changes to it. Here are the general steps of a local search algorithm:
- **Initialization:**
- The algorithm starts with an initial solution to the problem. This solution can be generated randomly or using a heuristic.
- **Evaluation:**
- The quality of the initial solution is evaluated using an objective function. The objective function measures how good the solution is, based on the problem constraints and requirements.
- **Neighborhood search:**
- The algorithm generates neighboring solutions by making small modifications to the current solution. These modifications can be random or guided by heuristics.

Working on a Local Search Algorithm



- **Selection:**
- The neighboring solutions are evaluated using the objective function, and the best solution is selected as the new current solution.
- **Termination:**
- The algorithm terminates when a stopping criterion is met. This criterion can be a maximum number of iterations, a threshold value for the objective function, or a time limit.
- **Solution:**
- The final solution is the best solution found during the search process.

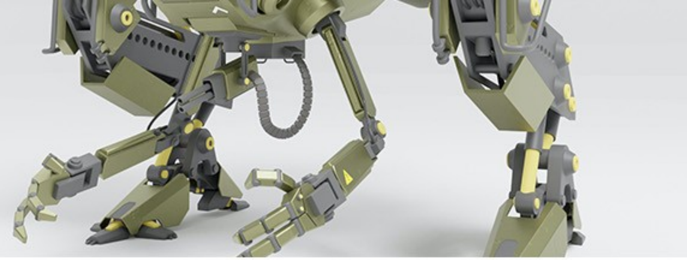
Local Search Algorithm: Hill Climbing



- Hill climbing algorithm is a local search algorithm which continuously moves in the direction of increasing elevation/value to find the peak of the mountain or best solution to the problem. It terminates when it reaches a peak value where no neighbor has a higher value.
- Hill climbing algorithm is a technique which is used for optimizing the mathematical problems. One of the widely discussed examples of Hill climbing algorithm is Traveling-salesman Problem in which we need to minimize the distance traveled by the salesman.
- It is also called greedy local search as it only looks to its good immediate neighbor state and not beyond that.
- A node of hill climbing algorithm has two components which are state and value. Hill Climbing is mostly used when a good heuristic is available.

- In this algorithm, we don't need to maintain and handle the search tree or graph as it only keeps a single current state.

Local Search Algorithm: Hill Climbing

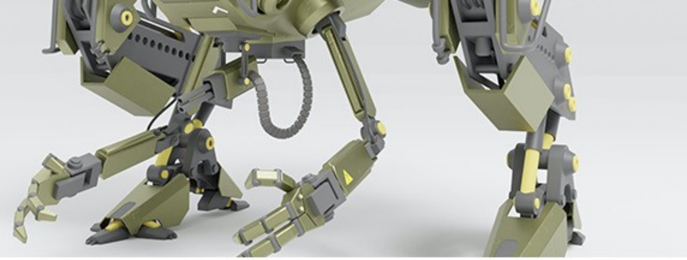


Features of Hill Climbing

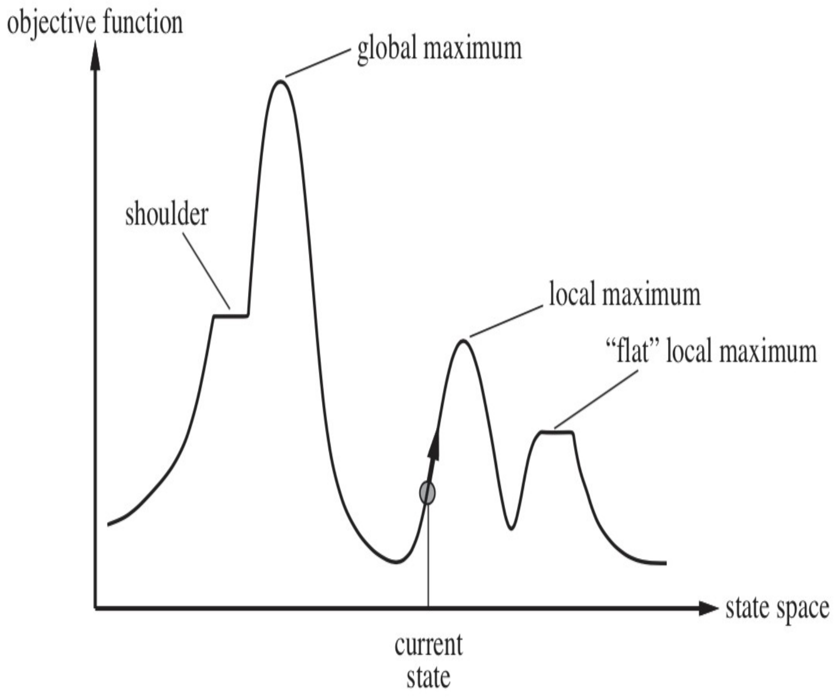
Following are some main features of Hill Climbing Algorithm:

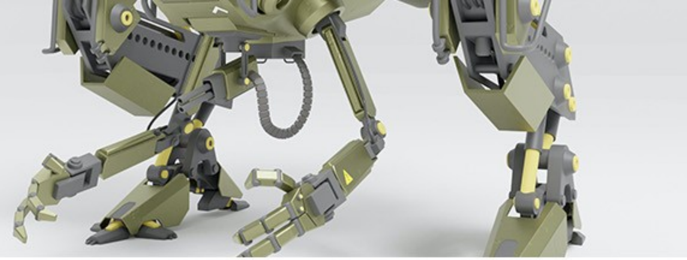
1. Generate and Test variant: Hill Climbing is the variant of Generate and Test method. The Generate and Test method produce feedback which helps to decide which direction to move in the search space.
2. Greedy approach: Hill-climbing algorithm search moves in the direction which optimizes the cost.
3. No backtracking: It does not backtrack the search space, as it does not remember the previous states.

State-space Diagram for Hill Climbing



Location (x-axis): State
Elevation (y-axis): Heuistic cost function or objective function
Global Minimum: If elevation corresponds to cost, then the aim is to find the lowest valley
Global Maximum: If elevation corresponds to an objective function, then the aim is to find the highest peak
Local Maximum: a peak that is higher than or equal to each of its neighboring states but lower than the global maximum
Plateau: a flat area of the state-space landscape (either a flat local maximum, from which no uphill exit exists, or shoulder, from which progress is possible)





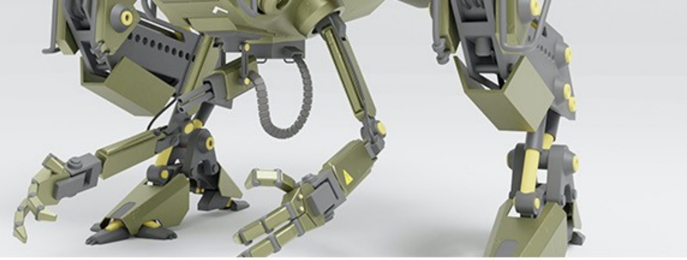
Hill Climbing

Advantages

- Hill climbing is very useful in routing-related problems like travelling salesmen problem, job scheduling, chip designing, and portfolio management.
- It is good in solving optimization problems while using only limited computation power.
- It is sometimes more efficient than other search algorithms.
- Even though it may not give the optimal solution, it gives decent solutions to computationally challenging problems.

Disadvantages

- Both the basic hill climbing and the steepest-ascent hill climbing may fail to produce a solution. Either the algorithm terminates without finding a goal state or getting into a state from which no better state can be generated. This will happen if the programme has reached either a local maximum, a ridge or a plateau

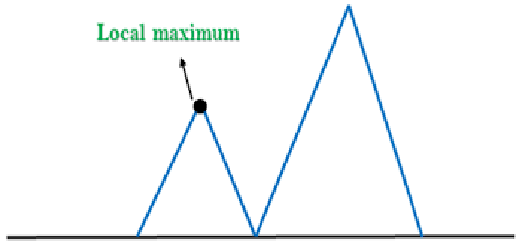


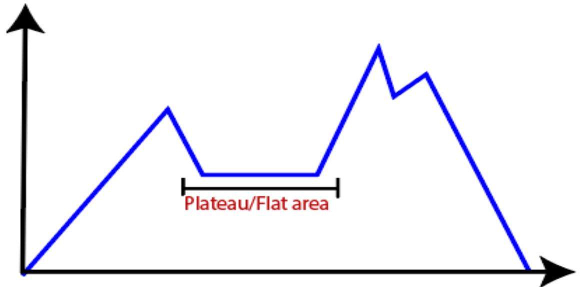
Hill Climbing

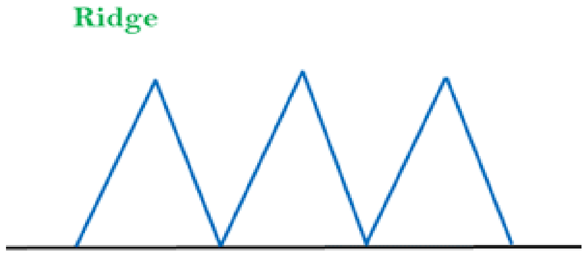
Limitations:

Hill climbing algorithms have limitations that can affect their performance in certain situations. Here are some of the main limitations of hill climbing algorithms:

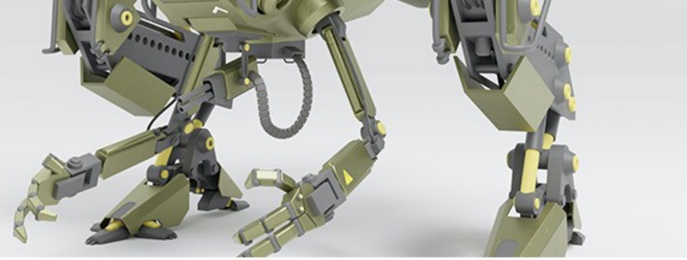
- Local Optima: Hill climbing algorithms can get stuck in local optima and need help finding the global optimum. This occurs when the algorithm converges to a solution that is locally optimal but not globally optimal.
- Plateaus: Plateaus occur when the objective function is flat or nearly flat in certain regions of the search space. In such cases, hill climbing algorithms may take a long to reach the global optimum or get stuck at a local optimum.
- Ridges: A ridge is a special form of the local maximum, which has a higher area than its surrounding but itself has a slope that cannot be reached in a single move.



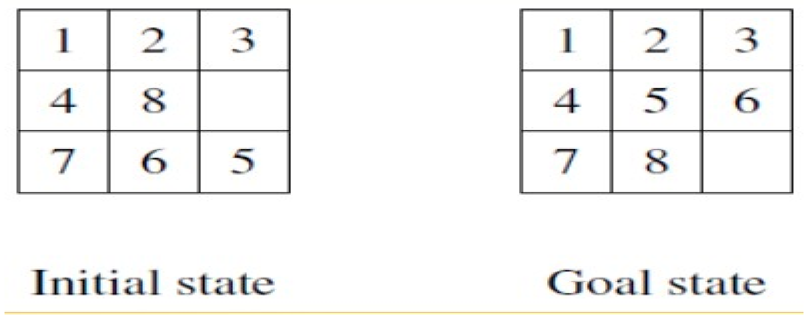




Hill Climbing



Example: Given the 8-puzzle shown in Figure, use the hill-climbing algorithm with the Manhattan distance heuristic to find a path to the goal state.



1	2	3
4	8	
7	6	5

Initial state

1	2	3
4	5	6
7	8	

Goal state

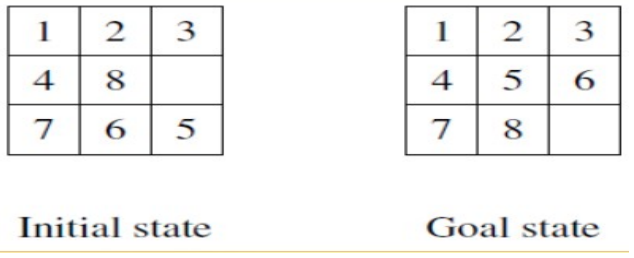
Solution

By definition, the Manhattan distance heuristic is the sum of the Manhattan distances of tiles from their goal positions. In Figure, only the tiles 5, 6 and 8 are misplaced and their distances from the goal positions are respectively 2, 2 and 1.

Therefore, h(Initial state) = 2 + 2 + 1 = 5

Hill Climbing

Example:

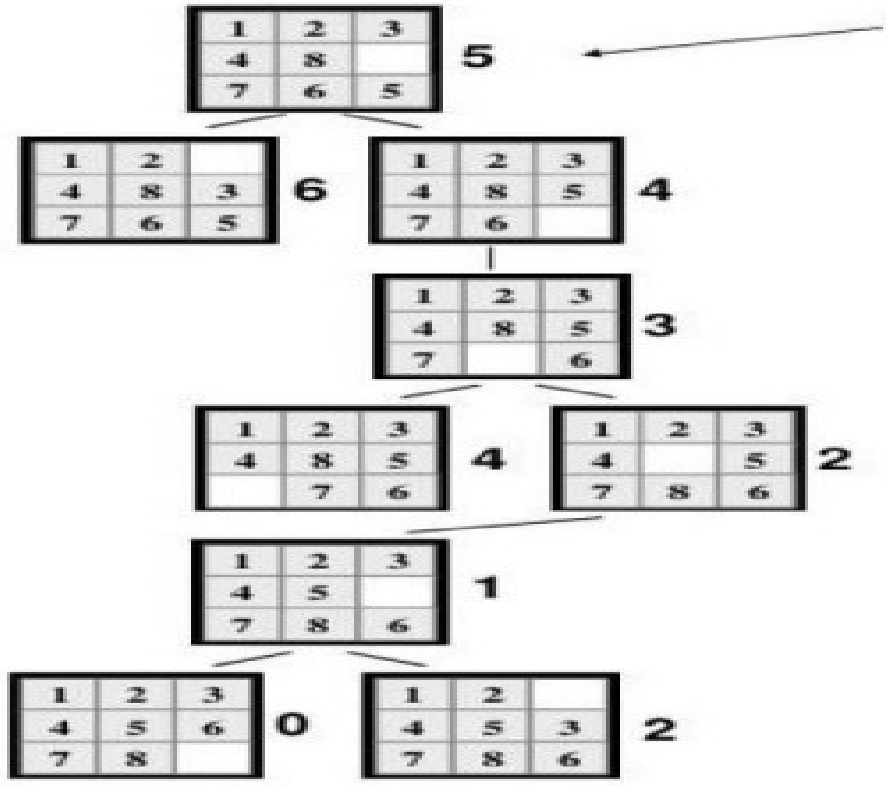


1	2	3
4	8	
7	6	5

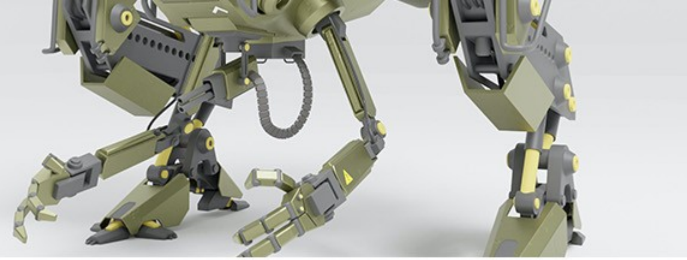
Initial state

1	2	3
4	5	6
7	8	

Goal state

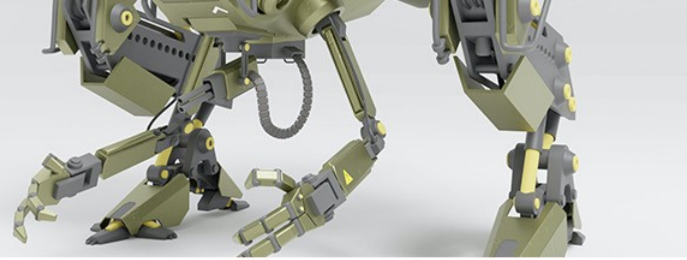


Local Search Algorithm: Local Beam Search



- Local beam search represents a parallelized adaptation of hill climbing, designed specifically to counteract the challenge of becoming ensnared in local optima. Instead of starting with a single initial solution, local beam search begins with multiple solutions, maintaining a fixed number (the "beam width") simultaneously. The algorithm explores the neighbors of all these solutions and selects the best solutions among them.
- Initialization: Start with multiple initial solutions.
- Evaluation: Evaluate the quality of each initial solution.
- Neighbor Generation: Generate neighboring solutions for all the current solutions.
- Selection: Choose the top solutions based on the improvement in the objective function.
- Termination: Continue iterating until a termination condition is met.
- Local beam search effectively avoids local optima because it maintains diversity in the solutions it explores. However, it requires more memory to store multiple solutions in memory simultaneously.

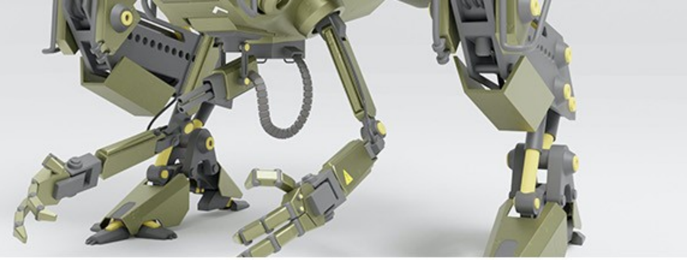
Local Search Algorithm: Simulated Annealing



- Simulated annealing is a probabilistic local search algorithm inspired by the annealing process in metallurgy. It allows the algorithm to accept worse solutions with a certain probability, which decreases over time. This randomness introduces exploration into the search process, helping the algorithm escape local optima and potentially find global optima.

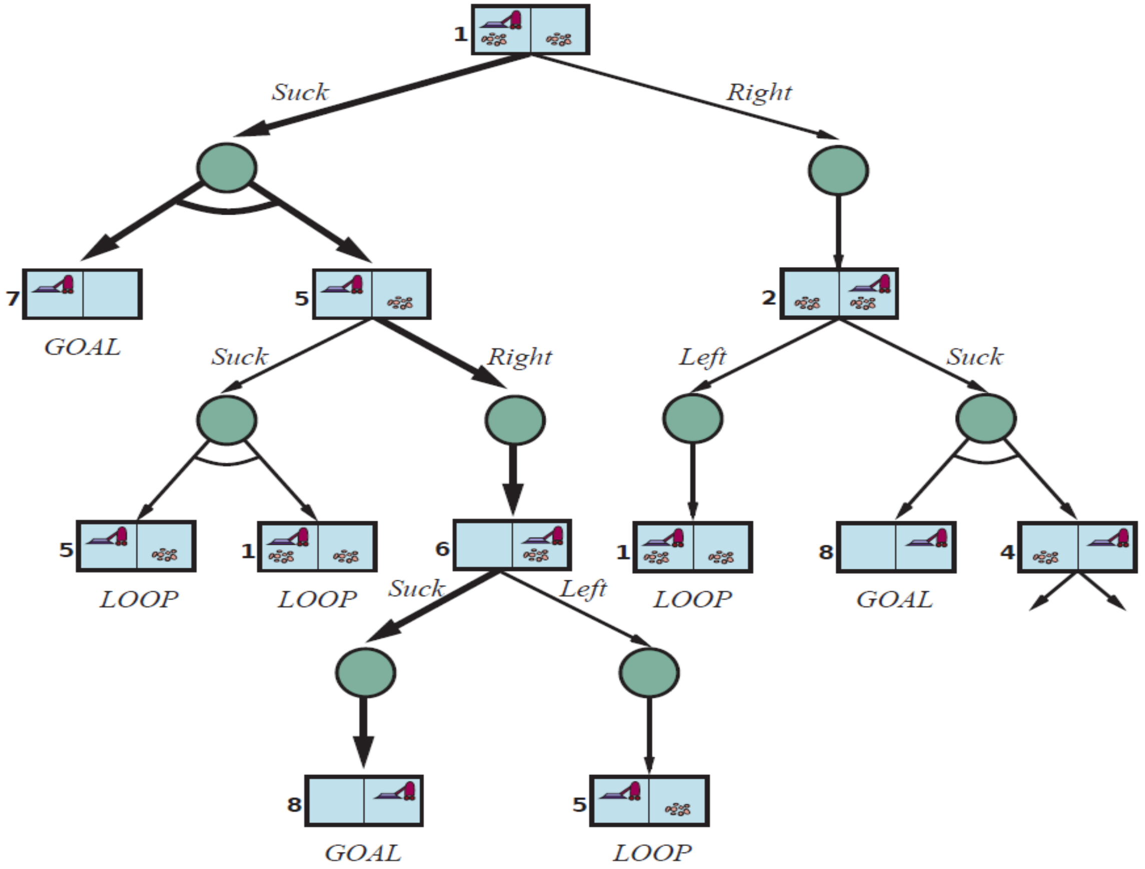
- Initialization: Start with an initial solution.
- Evaluation: Evaluate the quality of the initial solution.
- Neighbor Generation: Generate neighboring solutions.
- Selection: Choose a neighboring solution based on the improvement in the objective function and the probability of acceptance.
- Termination: Continue iterating until a termination condition is met.

Local Search Algorithm: Simulated Annealing



- The key to simulated annealing's success is the "temperature" parameter, which controls the likelihood of accepting worse solutions. Initially, the temperature is high, allowing for more exploration. As the algorithm progresses, the temperature decreases, reducing the acceptance probability and allowing the search to converge towards a better solution.

Search in Complex Enviroment


1
Suck
Right
7
GOAL
5
Suck
Right
2
Left
Suck
5
LOOP
1
LOOP
6
Suck
Left
1
LOOP
8
GOAL
4
8
GOAL
5
LOOP