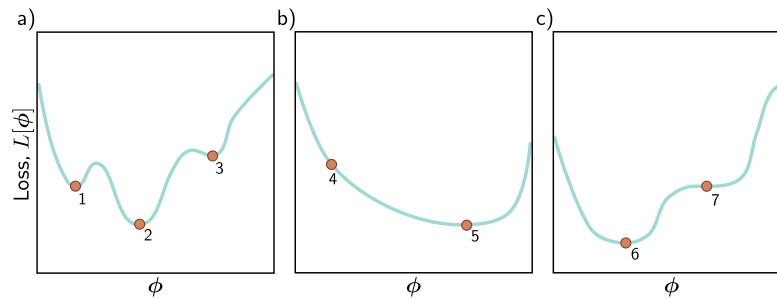


Local Search Review

Artificial Intelligence



1. Which of the functions above is/are convex?

Solution: Convex: b

2. Which of the points above is/are a local minimum?

Solution: Local minima: 1, 3, 7

3. Which of the points above is/are a global minimum?

Solution: Global minima: 2, 5, 6

4. Write the basic hill-climbing algorithm.

function HILL-CLIMBING(*problem*) **returns** a state that is a local maximum
 $current \leftarrow problem.INITIAL$
 while true do
 $neighbor \leftarrow$ a highest-valued successor state of $current$
 if VALUE($neighbor$) \leq VALUE($current$) **then return** $current$
Solution: $current \leftarrow neighbor$

5. What is the main weakness of hill-climbing algorithms?

Solution: They get stuck in local minima.

6. How can the basic hill-climbing algorithm be modified to overcome its weaknesses?

Solution:

- Allow "sideways" moves
- Stochastic hill climbing chooses randomly from uphill moves. Stochastic beam search does this with k states in parallel.
- Random restart hill climbing restarts from multiple initial states.

7. How does simulated annealing avoid getting stuck in local minima?

Solution: Instead of picking the best move, simulated annealing picks a random move. If the move is better than current state, it is always accepted. Otherwise, the algorithm accepts the move with some probability less than 1. The probability decreases exponentially with the "badness" of the move—the amount by which the evaluation is worsened. The probability also decreases as the "temperature" T goes down: "bad" moves are more likely to be allowed at the start when T is high, and they become more unlikely as T decreases. If the schedule lowers T to 0 slowly enough, then a property of the Boltzmann distribution, $e^{\Delta E/T}$, is that all the probability is concentrated on the global maxima, which the algorithm will find with probability approaching 1.

8. What is (stochastic) beam search?

Solution: Local beam search keeps track of k states rather than just one. It begins with k randomly generated states. At each step, all the successors of all k states are generated. If any one is a goal, the algorithm halts. Otherwise, it selects the k best successors from the complete list and repeats. In this way, information is passed among the parallel search threads.

Local beam search can suffer from lack of diversity – all k states clustered in a small region of the state space. Stochastic beam search fixes this problem by not simply choosing the top k successors, but successors with probability proportional to the successor's value, thus increasing diversity.

9. What does the mixing number parameter, ρ , in the basic algorithm control?

Solution: The number of "parents" used to generate new candidate solutions.

10. What do you have when you set $\rho = 1$ in the basic genetic algorithm?

Solution: When $\rho = 1$, the basic genetic algorithm is equivalent to stochastic beam search.

11. In gradient descent algorithms, what happens if you set the step size/learning rate parameter too high?

Solution: The algorithm might "skip over" the minimum.

12. Define belief state.

Solution: An agent's belief state is a set of states that the agent believes is possible. For example, if $Results(1, Suck) = \{5, 7\}$ and the agent executes *Suck* in State 1, then the agent's belief state is $\{5, 7\}$.

13. What form does the solution (sequence of actions that leads to a goal state) to an environment with nondeterministic actions take?

Solution: A conditional/contingency plan.

14. Is it possible to find a solution to a problem in a sensorless environment?

Solution: Yes. Even if the environment is nondeterministic, as long as the environment is known, e.g., agent has a map, then the environment can be coerced into a goal state by taking actions that successively eliminate non-goal states from the agent's belief states.

15. Describe the three-step state estimation procedure used by agents in partially observable environments.

Solution:

- The **prediction** stage computes the belief state resulting from the action, $Result(b, a)$. Because this is a prediction, we use the notation $\hat{b} = Result(b, a)$, where the hat over the b means "estimated," and we also use $Predict(b, a)$ as a synonym for $Result(b, a)$. Remember, a belief state is a **set** of states.

$$\hat{b} = Result(b, a) = Predict(b, a)$$

- The **possible percepts** stage computes the set of percepts that could be observed in the predicted belief state, that is, the set of percepts that could produce the predicted belief state (using the letter o for observation):

$$PossiblePercepts(\hat{b}) = \{o : o = Percept(s) \text{ and } s \in \hat{b}\}$$

- The **update** stage computes, for each possible percept, the belief state that would result from the percept. The updated belief state b_o is the set of states in b that could have produced the percept:

$$b_o = Update(\hat{b}, o) = \{s : o = Percept(s) \text{ and } s \in \hat{b}\}$$