Best-First Search: Following the Most Promising Leads

What it is:

- A search technique that uses a heuristic (a rule of thumb) to guide exploration.
- Like having a map that highlights the most promising paths towards your goal.

Key Idea:

- Heuristic Function: Estimates the “cost” or distance from a node to the goal.
- Priority: Always explores the most promising node first (lowest estimated cost).

How it Works:

1. Start: Begin with the initial state.
2. Evaluate: Apply the heuristic function to estimate the cost of reaching the goal from each successor node.
3. Prioritize: Add the successors to the list of open nodes (a priority queue).
4. Explore: Expand the most promising node (lowest cost).
5. Repeat: Continue until a solution is found or there are no more nodes to explore.

Above tree diagram illustrates a best-first search procedure.

1. Initial State: The search starts at node A.
2. Expansion: Node A is expanded, generating its successors: nodes B, C, and D. Each node has a heuristic value in parentheses representing the estimated cost to reach the goal from that node.
3. Prioritization: Node D has the lowest heuristic value (1), indicating it's the most promising path. So, the search expands node D next, generating nodes E and F.
4. Switching Paths: After evaluating nodes E and F, the path through node B now appears more promising. The search switches to node B, expanding it to generate nodes G and H.
5. Continuing the Search: The process continues, with the search switching back to the

path through D and expanding node E to generate I and J. The figure indicates that node J would be the next to be expanded since it has the lowest heuristic value.

Analogy:

Imagine searching for treasure in a vast forest. You'd use clues and a map to prioritize the most likely areas, even if they require detours.

Advantages:

- Efficiency: Avoids exploring less promising paths.
- Flexibility: Can switch between paths if a more promising one emerges.

Disadvantages:

- Can be computationally expensive.
- Relies on a good heuristic.

Applications:

- Game playing (finding the best move)
- Route planning (shortest path)
- Machine learning (optimizing parameters)

pseudocode for the Best-First Search algorithm

1. Start with OPEN containing just the initial state.
2. Until a goal is found or there are no nodes left on OPEN do:

(a) Pick the best node on OPEN.
(b) Generate its successors.
(c) For each successor do:
(i) If it has not been generated before, evaluate it, add it to OPEN, and record its parent.
(ii) If it has been generated before, change the parent if this new path is better than the previous one. In that case, update the cost of getting to this node and to any successors that this node may already have.

References:

- Russell, S., and Norvig, P. Artificial Intelligence: A Modern Approach, 4th Edition, 2020, Pearson.
- Rich, E., Knight, K., & Nair, S. B. Artificial Intelligence. McGraw-Hill International.
- Nilsson, N. J. Artificial Intelligence: A New Synthesis. Morgan Kaufmann.

Note: This content was generated with the assistance of Google’s Gemini AI.

Related posts:

1. Artificial Intelligence Intelligence Tutorial for Beginners
2. Difference between Supervised vs Unsupervised vs Reinforcement learning
3. What is training data in Machine learning
4. What other technologies do I need to master AI?
5. How Artificial Intelligence (AI) Impacts Your Daily Life ?
6. Like machine learning, what are other approaches in AI ?
7. Heuristic Search Algorithm
8. Hill Climbing in AI

9. A* and AO* Search Algorithm
10. Knowledge Representation in AI
11. Propositional Logic and Predicate Logic
12. Resolution and refutation in AI
13. Deduction, theorem proving and inferencing in AI
14. Monotonic and non-monotonic reasoning in AI
15. Probabilistic reasoning in AI
16. Bayes’ Theorem
17. Artificial Intelligence Short exam Notes
18. Transformer Architecture in LLM
19. Input Embedding in Transformers
20. Positional Encoding in Transformers
21. Multi-Head Attention in Transformers