ARTIFICIAL INTELLIGENCE

UNINFORMED SEARCING

UNINFORMED SEARCH ALGORITHMS

Uninformed search algorithms, also known as blind search algorithms, systematically explore the search space without any domain-specific knowledge to find solutions.

COMMON TYPES OF UNINFORMED SEARCH ALGORITHMS

01

BFS

This algorithm explores all nodes at the present depth before moving on to nodes at the next depth level.

02

DFS

This algorithm explores as far down a branch s possible before backtracking.

03

UNIFORM COT EARCH

Uniform Cost Search: This algorithm expands the least costly node first, ensuring that the path found is the least expensive.

BREADTH FIRST SEARCH ALGORITHM

Breadth First Search (BFS) is a graph traversal technique that explores vertices level by level. Starting from a source node, it visits all its immediate neighbors before moving to the next level of neighbors. This ensures that nodes are visited in order of their distance from the source.

```
from collections import deque
def bfs(graph, start):
    visited = set() # Track visited nodes
    queue = deque([start]) # Initialize queue with start node
    visited.add(start)
    while queue:
        vertex = queue.popleft() # Dequeue a vertex
        print(vertex, end=" ") # Process the vertex
        for neighbor in graph[vertex]:
            if neighbor not in visited:
                visited.add(neighbor)
                queue.append(neighbor)
```

```
# EXAMPLE USAGE
    GRAPH = {
        0: [1, 2],
        1: [0, 2],
    2: [0, 1, 3, 4],
        3: [2],
        4: [2]
            }
BFS(GRAPH, 0)
```

BREADTH FIRST SEARCH (BFS)

BFS is simple, robust, and guarantees shortest path discovery in unweighted graphs, making it a fundamental algorithm in graph theory and real-world applications.

COMPLEXITY:

- Time: O(V + E) — Each vertex and edge is processed once.
- Space: O(V) — For the visited set and queue storage.

APPLICATIONS:

- Shortest Path in unweighted graphs.
- Connected Components detection.
  - Cycle Detection in undirected graphs.
  - Web Crawlers and Network Broadcasting.s.

HOW IT WORKS:

1. Initialization – Mark the start node as visited and enqueue it.
2. Processing – Dequeue a node, process it, and enqueue all its unvisited neighbors.
3. Repeat – Continue until the queue is empty.
4. Disconnected Graphs – If the graph is disconnected, run BFS from each unvisited node to cover all components.