USECASES

Breath-first-search:

1. When the solution depth is small.

2. When finding the shortest solution in terms of moves is

Depth_first-search:

1. When memory is limited.

2. Exploring deep paths quickly without concern for optimality.

Backtracking:

important.

1. Suitable for constraint satisfaction problems where constraints need to be satisfied.

2.Can be adapted for heuristic-based searches.

DFID:

path cost.

- 1. When the solution depth is unknown and optimality is required.
- 2. When memory constraints prevent BFS.

Best-First Search (Out of Place Heuristic):

1. When solving the 8-puzzle heuristically without considering

2. When faster exploration is desired but optimality is not critical.

Best-First Search (Manhattan Distance Heuristic):

1. When a more accurate heuristic is needed.

2. When faster solving is needed with a focus on reducing the number of misplaced tiles.

A* Algorithm (Manhattan Distance Heuristic):

1. When both optimality and efficiency are required.

2. Widely used in real-world problems for finding the shortest path with an efficient heuristic.



PROS AND CONS

Breath-first-search:

Pros: Guarantees optimal solution, complete. **Cons**: High memory usage, slow for deep solutions.

Depth_first-search:

Pros: Less memory usage, explores deep solutions quickly. **Cons**: Not optimal, can get stuck in loops.

Backtracking:

Pros: Efficient search, uses less memory. **Cons**: Can be slow without a good heuristic, no optimality guarantee.

DFID:

Pros: Optimal like BFS, low memory usage. **Cons**: Repeats states, slower than BFS, uninformed.

Best-First Search (Out of Place Heuristic):

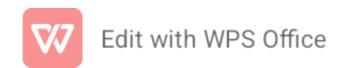
Pros: Faster than BFS, more memory-efficient. **Cons**: Not guaranteed optimal, can get stuck in local optima.

Best-First Search (Manhattan Distance Heuristic):

Pros: More accurate and faster than other heuristics. **Cons**: Not guaranteed optimal, high memory usage.

A* Algorithm (Manhattan Distance Heuristic):

Pros: Guarantees optimal solution, efficient with good heuristics. **Cons**: Memory-intensive, can be slow with poor heuristics.



COMPARISON ACCORDING TO THE VALUES I

NOTED

In Terms of Visited Nodes:

DFS (124259) > BFS (107388) > DFID (41273) > *A (13582)** > Backtracking (3029) > BFS with Out-of-Place Heuristic (178) > BFS with Manhattan Heuristic (153)

In Terms of Number of Moves Suggested:

DFS (56997) > Backtracking (3300) > BFS with Out-of-Place Heuristic (69) > BFS with Manhattan Heuristic (49) > DFID (35) > BFS (25) = A*(25)

In Terms of Time Consumed:

DFS (7.7s) > DFID (6.5s) > BFS (1.99s) > Backtracking (1.129s) > BFS with Out-of-Place Heuristic (0.7s) = BFS with Manhattan Heuristic (0.7s) > A*(0.48s)

In Terms of Memory Consumed(MB):

DFS (71.32) > BFS (55.39) > A*(12.89) > DFID (2.38) > Backtracking (1) > BFS with Out-of-Place Heuristic (0.01) > BFS with Manhattan Heuristic (0.00)

UNINFORMED VS INFORMED

Uninformed algorithms (like BFS, DFS, DFID) explore a lot of nodes, using more time and memory. **DFS** is the worst, while **BFS** is better but still explores too many nodes.

Informed algorithms (like A* and heuristic BFS) use heuristics to guide the search, making them faster and more memory-efficient. **A*** is the best overall, balancing speed and memory.



IMPACTS OF DIFFERENT HEURISTICS

Heuristics significantly improve efficiency. For example, BFS with a heuristic (like Out-of-Place or Manhattan) explores far fewer nodes and uses much less memory compared to standard BFS.

Manhattan Heuristic performs the best in terms of both **speed and efficiency** (with fewer nodes explored and fewer moves compared to the Out-of-Place heuristic).

A* is the best overall because it finds the optimal solution (shortest path) in the least amount of time (0.48s) with a reasonable amount of memory (12.89MB).

CONCLUSION

DFS is the worst performer because it uses too many moves, takes up a lot of memory, and takes too much time.

BFS is an improvement but still ends up exploring a lot of nodes.

A* is the best overall because it balances speed, efficiency, and memory usage.

BFS with the Manhattan Heuristic is the most memory-efficient, but A* finds the shortest solution faster.

