BEST-FIRST SEARCH

Authored by

mohammad looti

November 8, 2025

BEST-FIRST SEARCH

Primary Disciplinary Field(s): Artificial Intelligence, Computer Science, Operations Research

1. Core Definition

The **Best-First Search** (BFS) algorithm represents a foundational methodology within the field of Artificial Intelligence, specifically designated for graph traversal and problem-solving where the objective is to find the most efficient path from an initial state to a goal state. Unlike uninformed search strategies--such as Breadth-First Search (BFS) or Depth-First Search (DFS)--which explore the search space systematically without foresight, Best-First Search is an **informed search** technique. This means it incorporates domain-specific knowledge, usually in the form of a **heuristic function**, to guide its exploration process. The fundamental principle is to prioritize the expansion of the node that appears most promising or "best" according to this evaluation function.

In operational terms, the algorithm perpetually assesses all currently available successor paths, calculating for each one the estimated cost or distance remaining to reach the solution. This calculation provides a quantitative measure of proximity to the target. The search then selectively expands the node (or state) that possesses the most favorable evaluation score, under the assumption that this path is statistically or empirically most likely to lead to a successful and prompt resolution. This focus on probability and proximity makes Best-First Search an inherently goal-directed strategy, contrasting sharply with exhaustive search methods that might waste computational resources exploring irrelevant branches of the search tree. As noted in the foundational understanding of the method: "In terms of problem-solving, the **best-first search** is one of the fastest, most practical ways to end a search and find the solution," particularly when compared to less sophisticated search protocols.

The core efficiency benefit of Best-First Search stems directly from its ability to prune large sections of the search space that are deemed unlikely to contain the solution. By employing a priority queue to maintain and rank all generated nodes based on their heuristic scores, the algorithm ensures that computational effort is constantly directed toward the frontier closest to the ultimate goal. The general framework encompasses several significant variants, most notably the **Greedy Best-First Search** and the highly influential A* Search algorithm, both of which utilize the core principle of informed node selection but differ critically in the specific formulation of their evaluation functions.

2. Etymology and Historical Development

The development of Best-First Search algorithms is intrinsically linked to the early history of Artificial Intelligence, specifically the pursuit of creating systems capable of solving complex

problems efficiently, such as game playing or theorem proving. Prior to the formalization of informed search, many AI problems required exhaustive methods that quickly became computationally intractable as the search space grew--a phenomenon known as the **combinatorial explosion**. Researchers in the 1960s recognized the necessity of incorporating knowledge into the search process to manage this complexity effectively, moving away from brute-force exploration.

The foundational work formalizing the concept of a general best-first strategy is often attributed to Nils J. Nilsson, particularly through his influential textbooks and research in the late 1960s and early 1970s. Nilsson defined the general Best-First Search approach as a method where an evaluation function $f(n)$ is applied to a node n, and the node with the minimum f value is chosen for expansion next. This general template allowed for the creation of specific, powerful search techniques. The most significant and mathematically rigorous refinement of the Best-First approach emerged with the introduction of the A* search algorithm by Peter Hart, Nils Nilsson, and Bertram Raphael in 1968, building upon earlier work like the A algorithm.

A* cemented the legacy of informed search by providing a Best-First strategy that was proven to be both complete (guaranteed to find a solution if one exists) and optimal (guaranteed to find the lowest-cost solution) under the condition that the heuristic function used is admissible. This achievement marked a critical turning point, moving AI search methodology beyond simple trial-and-error toward sophisticated, knowledge-driven exploration. Consequently, the term "Best-First Search" today often serves as an umbrella category encompassing any search algorithm that uses an evaluation function to prioritize nodes, with A* being the most widely adopted and studied implementation due to its robustness and proven guarantees regarding solution quality.

3. Key Characteristics and Components

The effectiveness of any Best-First Search implementation relies on the careful integration of several core components that manage the state space and direct the search process. These components ensure that the algorithm remains focused on the most promising paths while preventing redundant effort. The operational structure typically involves managing two primary lists that organize the frontier of the search: the OPEN list and the CLOSED list, both crucial for efficient graph traversal.

The **OPEN list** functions as the priority queue, containing all generated nodes that have yet to be examined. Crucially, the nodes in the OPEN list are continuously ranked based on their calculated heuristic score $f(n)$. The node with the best (lowest or highest, depending on whether the function calculates cost or utility) evaluation is always chosen for expansion. This choice of the "best" node ensures that the search is always progressing along the path estimated to be most successful. Conversely, the **CLOSED list** stores all nodes that have already been expanded and

evaluated. This prevents the search from returning to previously evaluated states, which is essential for ensuring termination and avoiding redundant cycles in complex, high-dimensional problem spaces.

The most defining characteristic is the **Heuristic Function**, $h(n)$. This function provides the estimate of the cost from the current node n to the goal state. A well-designed heuristic is crucial; a poor heuristic provides little guidance, causing the search to degrade toward uninformed methods like uniform cost search, whereas a powerful, accurate heuristic significantly speeds up convergence by correctly identifying fruitful pathways. Furthermore, the selection mechanism--always picking the node with the best score--means that Best-First Search is fundamentally a form of **greedy search** if only the heuristic cost $h(n)$ is considered, or a form of **optimal search** if the full evaluation function includes the actual cost incurred so far, $g(n)$, as is the rigorous formulation used in A* search.

4. Key Components and Operational Flow

Evaluation Function ($f(n)$): This is the formula used to determine the priority of a node. In general BFS, $f(n)$ is designed to estimate the overall desirability or proximity to the goal. In specific variants, such as A*, $f(n) = g(n) + h(n)$, combining known cost (g) and estimated cost (h).

Heuristic Function ($h(n)$): A domain-specific function that estimates the cost from the current node n to the goal. A strong heuristic minimizes the number of nodes explored, directly enhancing the algorithm's speed and practicality.

Priority Queue (OPEN List): Maintains the set of nodes available for expansion, ordered by their $f(n)$ score. The operational flow dictates that the node with the highest priority (best score) is always removed first for expansion.

Expansion Process: When a node is selected from the OPEN list, its neighbors (successors) are generated. These neighbors are evaluated using the $f(n)$ function, and if they have not been processed, they are added to the OPEN list. The expanded node itself is moved to the CLOSED list.

Admissibility and Consistency: For optimal results (as in A*), the heuristic must be **admissible** (never overestimating the true cost to the goal) and preferably **consistent** (the estimated cost between two nodes is never less than the actual cost).

5. Applications and Examples

Best-First Search algorithms, particularly A*, have enormous practical significance across

computer science and artificial intelligence, offering efficient solutions to problems that involve finding optimal paths or configurations within a large state space. The most immediate and pervasive application is in **pathfinding**, especially in robotics, video games, and geographical information systems (GIS). Game AI frequently uses A* to enable non-player characters (NPCs) to navigate complex terrain maps, find the shortest route to a target, and avoid obstacles dynamically, ensuring realistic and efficient movement within the virtual environment.

Beyond pathfinding, Best-First strategies are critical in complex combinatorial optimization problems where the goal is to find the best configuration among an exponential number of choices. For instance, in operational research, variants of BFS are used for scheduling tasks, resource allocation, and optimizing logistical routes (e.g., the Traveling Salesperson Problem, often addressed using specialized search techniques informed by BFS principles). In computational linguistics and bioinformatics, they are applied in tasks like finding the most probable parse tree for a sentence or aligning genetic sequences, where the heuristic guides the search toward statistically likely solutions.

Classic examples used to demonstrate the power of BFS include the **Eight Puzzle** and the **Missionaries and Cannibals problem**. In the Eight Puzzle, the state space (all possible tile configurations) is vast, but an effective heuristic (like the Manhattan distance or the number of misplaced tiles) allows a Best-First Search to solve the puzzle rapidly by prioritizing moves that bring the tiles closer to their correct goal positions. This rapid resolution validates the principle that informed search drastically reduces the time complexity compared to uninformed methods.

6. Debates and Criticisms

Despite the widespread success and theoretical elegance of Best-First Search algorithms, they are not without significant practical and theoretical limitations, primarily revolving around resource consumption and the reliability of the heuristic. The primary challenge lies in the **design of the heuristic function**. While A* requires an admissible heuristic for optimality, finding a heuristic that is both admissible and highly informative (i.e., closely estimates the true cost) can be extremely difficult, especially for novel or highly abstract problem domains. If the heuristic is weak, the search effort increases dramatically; if it is inconsistent or non-admissible, A* loses its guarantee of optimality and may return suboptimal solutions without warning.

A significant practical criticism, particularly concerning A*, is its high memory requirement, often termed the **space complexity problem**. Because A* needs to maintain the complete OPEN list of all generated nodes and their associated costs--as well as the CLOSED list to avoid revisits--the algorithm can rapidly exhaust system memory in problems with very deep or broad state spaces. This limitation often renders standard A* impractical for state spaces exceeding a few million nodes. Alternative, memory-constrained algorithms, such as Iterative Deepening A* (IDA*) or

Simplified Memory-Bounded A* (SMA*), were developed specifically to address this constraint by sacrificing some execution time or optimality guarantees to manage limited resources more effectively.

Finally, non-optimal variants like **Greedy Best-First Search** face the fundamental criticism of incompleteness and sub-optimality. While often faster than A* in execution time, their inherent short-sightedness means they may follow a highly attractive path that quickly leads to a dead end, resulting in the failure to find a solution even when one exists (incompleteness). Furthermore, they frequently return a solution that is significantly more costly than the true optimal path (sub-optimality). The critical decision in applying BFS thus becomes a trade-off between the guaranteed quality offered by A* and the computational speed favored by the purely greedy approach.

7. Further Reading

Wikipedia: Best-first search
Wikipedia: A* search algorithm
Wikipedia: Heuristic function
Artificial Intelligence: A Modern Approach (AIMA) Textbook Resources