Solving the Farmer, Wolf, Goat, and Cabbage Puzzle: A Java Implementation

Problem Statement

The Farmer, Wolf, Goat, and Cabbage puzzle is a classic AI problem where a farmer must transport all four items across a river. The constraints are:

- The farmer can only take one item at a time.
- The wolf cannot be left alone with the goat.
- The goat cannot be left alone with the cabbage.

This puzzle can be solved using various search algorithms, and this report focuses on a breadth-first search implementation in Java.

Java Implementation

Main Class:

- Initializes the initial state of the puzzle.
- Creates a `BreadthFirstSolver` object.
- Calls the `solve` method of the solver, passing the initial state.
- Prints the solution path.

FWGCState Class:

- Represents a state of the puzzle.
- Attributes include the positions of the farmer, wolf, goat, and cabbage (either on the west or east side).
- Provides methods to:
    - Check if the current state is the goal state (all items on the west side).

    - Generate all possible next states from the current state, ensuring they don't violate the constraints.
    - Check if two states are equal.
    - Print the current state.

BreadthFirstSolver Class:

- Implements the breadth-first search algorithm.
- Uses two lists: `queue` for states to be explored and `closed` for states already explored.
- The `solve` method:
    - Initializes the `queue` and `closed` lists.
    - Adds the initial state to the `queue`.
    - Iteratively removes the first state from the `queue` and checks if it's the goal state.
    - If not, generates all possible next states and adds them to the `queue` if they haven't been explored yet.
- The `findPath` method is used to trace the path from the goal state back to the initial state, reconstructing the solution.

Algorithm and Data Structures

The breadth-first search algorithm is a graph search algorithm that explores all neighbors at the present depth prior to moving on to the neighbors at the next depth level. In this implementation:

- A `queue` data structure is used to store states to be explored.
- A `closed` list is used to keep track of states that have already been visited.
- The `FWGCState` class represents a node in the search graph.

Advantages of Breadth-First Search for this Problem

- **Guaranteed to find the shortest solution:** If a solution exists, breadth-first search will find the solution with the fewest moves.
- **Simple to implement:** The algorithm is relatively straightforward to implement.

Comparison with Prolog

While Prolog is well-suited for logic-based problems and offers a declarative programming style, Java provides more flexibility and control over the implementation. For large-scale problems, Java's iterative approach and explicit memory management can be more efficient than Prolog's backtracking mechanism, which can lead to stack overflows for large search spaces.

In conclusion, this Java implementation effectively solves the Farmer, Wolf, Goat, and Cabbage puzzle using a breadth-first search algorithm. The code is well-structured and provides a clear demonstration of how to apply AI search techniques to solve a classic problem.