Capstone Project Report

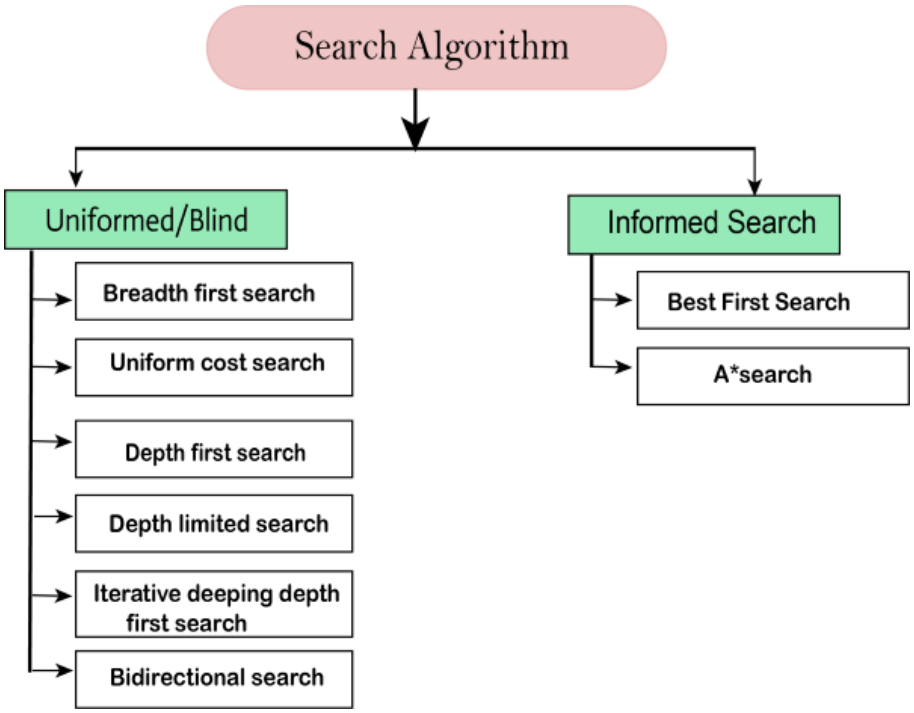
1. Presentation of the subject.

Problem solving by searching.

- Problem-solving agents.

  In Artificial Intelligence, Search techniques are universal problem-solving methods. Rational agents or Problem-solving agents in AI mostly used these search strategies or algorithms to solve a specific problem and provide the best result. Problem-solving agents are the goal-based agents and use atomic representation.

- Types of search algorithms.



(Picture from Search Algorithms in Artificial Intelligence, Java T point)

2. Description of the problem.

- Extended wolf-goat-cabbage problem.
  - The shepherd must bring a wolf, a goat, a cabbage, a wooden stick and a fire torch on the other side of a river.

- Without the shepherd, if the wolf and the goat are together, the wolf eats the goat.
- Without the shepherd, if the goat and the cabbage are together, the goat eats the cabbage.
- Without the shepherd, if the stick and the wolf are together, the stick beats the wolf.
- Without the shepherd, if the stick and the torch are together, the torch burns the stick.
- The boat can transport 2 objects or animals across the river at each step.

- Problem formulation.
  - **States:** boat location + animals and objects location. There are 6 things (shepherd, goat, cabbage, wolf, wooden stick, fire torch) and each one has 2 status (on each side of the river) → 6×2^6 states
  - **Initial state:** The three animals, two objects and the boat are on the left side of the river.
  - **Actions:** Bring up to 2 animals or objects at each time the boat crossing the river.
  - **Goal test:** All the animals and objects are on the other side of the river.
  - **Path cost:** Each step cost 1, so path cost = number of steps in path.

3. Selecting the algorithms to be used for solving the problem.

- Generate new states:

```
def gen_brand(visited, parent_map):
    if boat on the right:
        for i in left:
            new_left = left(copy)
            new_right = right(copy)
            remove i from left
            append i in right
            for j in left:
                next_left = new_left(copy)
                next right = new_right(copy)
                remove i from left
                append j in right
                if sorted(next_left) not conflict_states and not existed_states:
                    child = new_state
                    parent_map = current_state
            if sorted(new_left) not conflict_states and not existed_states:
                child = new_state
                parent_map = current_state
        if sorted(left) not conflict_states and not existed_states:
            child = new_state
            parent_map = current_state
    else:
        ……
```

We use 2 “for” loop to remove 2 animals or objects from a list of things on the left side and append them into the list of things on the right side. After that, in the second “for” loop (bringing 2 things at a time) we check if it were a conflict states or existed states or not, we also check it in the outer “for” loop (bringing 1 thing at a time) and outside the “for” loop (bringing 0 thing at a time). Finally, all the satisfied states are saved. The processes are the same when the boat is on the other side of the river.

- Breadth-first Search and Depth-first Search:

  After the states are all generated, we use BFS and DFS to find the path from the initial state to the goal test. We chose BFS because it checked all the states and give optimal solution. On the other hand, we picked DFS because its running time and space are much smaller. Besides, the number of states is finite so we expected DFS will find a solution. Overall, we want to compare the time and space complexity, optimization of two algorithms.

4. Implementing the algorithms used for solving the problem.

- Do not know how to generate new states.

→ We refer to the original wolf-goat-cabbage problem's solutions from the Internet. Finally, we found Ethen Beaver's ideas were easy to understand and develop.

- Do not know which search algorithms to use.

→ We noticed that BFS and DFS had some aspects that we can compare.

5. Comparing the results of the algorithms used for solving the problem.

a. Providing quantitative performance indicators.

   - Breadth-first search solution:

   ['w', 'g', 'c', 's', 't']~~~[]- The boat is on the Left

   ['g', 'c', 's']~~~['t', 'w']- The boat is on the Right

   ['g', 'c', 's']~~~['t', 'w']- The boat is on the Left

   ['g']~~~['t', 'w', 's', 'c']- The boat is on the Right

   ['g', 's']~~~['t', 'w', 'c']- The boat is on the Left

   []~~~['t', 'w', 'c', 's', 'g']- The boat is on the Right

   - Depth-first search solution:

   ['w', 'g', 'c', 's', 't']~~~[]- The boat is on the Left

   ['g', 'c', 's']~~~['t', 'w']- The boat is on the Right

['g', 'c', 's']~~~['t', 'w']- The boat is on the Left

['g', 'c']~~~['t', 'w', 's']- The boat is on the Right

['g', 'c', 's', 'w']~~~['t']- The boat is on the Left

['g', 's']~~~['t', 'w', 'c']- The boat is on the Right

['g', 's']~~~['t', 'w', 'c']- The boat is on the Left

['g']~~~['t', 'w', 'c', 's']- The boat is on the Right

['g', 's', 'w']~~~['t', 'c']- The boat is on the Left

['s']~~~['t', 'c', 'w', 'g']- The boat is on the Right

['s', 'g', 't']~~~['c', 'w']- The boat is on the Left

['t']~~~['c', 'w', 'g', 's']- The boat is on the Right

['t', 's', 'w']~~~['c', 'g']- The boat is on the Left

['t', 'w']~~~['c', 'g', 's']- The boat is on the Right

['t', 'w']~~~['c', 'g', 's']- The boat is on the Left

[]~~~['c', 'g', 's', 'w', 't']- The boat is on the Right

- Overall Analytics:

	Complete?	Time	Space	Optimal?
Breadth-first Search	Yes	600	711	Yes
Depth-first Search	Yes	16	88	No

b. Explaining the results.

- Both algorithms are complete because the number of vertices is finite.

- The running time and the space complexity of BFS are much larger than those of DFS because BFS visits all the vertices on the same layer first then move to the next

layer until all the vertices are visited, but DFS visits each branch of the tree first then expand it until it find the solution.

- Depth-first search is not optimal because the algorithms will stop when finding the solution. However, Breadth-first search considers all possibilities so that it will select the optimal one.

6. Conclusion and possible extensions.

In conclusion, we feel a sense of achievement after completing this project. Afterward, we have learnt how to formulate a problem solved by searching, analyze and implement search algorithms. More than that, we have had a chance to work together and learnt so much about team work, researching, writing report skills. However, there were still some drawbacks that we did not really proud of. We only used two search algorithms, if had more time, we would try to use other search algorithms (Uniform-cost search, Depth-limited search, Iterative Deepening). On top of that, we could not finish the possible extension, which is automatically generated initial state. We think that we choose randomly one state from the state list and define it as the initial one but there was no time left so we were unable to finish it.

7. List of tasks.

- Programming tasks:
  - Writing the algorithms to generate the search tree: member 2 Huy Hoàng.
  - Writing BFS and DFS algorithms: member 2 Huy Hoàng.
  - Running instances of the problem: member 1 Đức Anh.
- Analytic tasks:
  - Proposing our subject: member 1 Đức Anh 50%; member 2 Huy Hoàng 50%.
  - Researching about how to generate the search tree: member 2 Huy Hoàng.
  - Selecting the algorithms to be used to solve this problem: member 1 Đức Anh.
  - Preparing the PDF file for presentation: member 1 Đức Anh 60%, member 2 Huy Hoàng 40%.
  - Writing the report: member 2 Huy Hoàng.

8. List of bibliographic references.

- Using notions and pictures from Search Algorithms in Artificial Intelligence, Java T point: https://www.javatpoint.com/search-algorithms-in-ai#:~:text=In%20Artificial%20Intelligence%2C%20Search%20techniques,agents%20and%20use%20atomic%20representation.
- Consulting the original wolf-goat-cabbage solution from Ethen Beaver on GitHub: https://github.com/ethanbeaver/Wolf-Goat-Cabbage-Problem
- Researching about static method on Geeks for Geeks: https://www.geeksforgeeks.org/class-method-vs-static-method-python/