CS 451/551 Introduction to Artificial Intelligence

Assignment – 1: Solving TSP with UCS & A*

Created by Anıl Doğru & Mehmet Onur Keskin for help → {anil.dogru, onur.keskin}@ozu.edu.tr

1. Definition

In this assignment, you will implement different search algorithms to solve the Travelling Salesperson Problem (TSP), where travel nodes are varied between 12-14. We provided some baseline code for the TSP environment (on LMS), and you need to implement well-known search algorithms incrementally. Mainly, we expect you to implement Uniform Cost Search (UCS) & A* algorithms for this assignment.

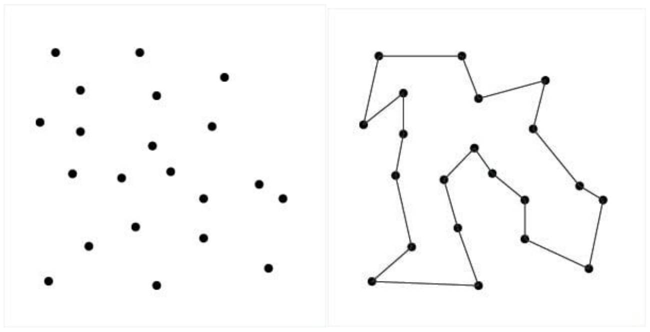
In code baseline, you will have six different script files as follows, and we strongly encourage you to check over this file to understand what a node can do. NOT allowed to change, and you might not want to change it. You are only responsible for the two classes highlighted in **red**:

- Main.py
- Node.py
- PriorityQueue.py
- DataReader.py
- Algorithm.py
- JSON Data Files
- **UniformCostSearch.py**
- **AStar.py**

2. Travelling Salesperson Problem

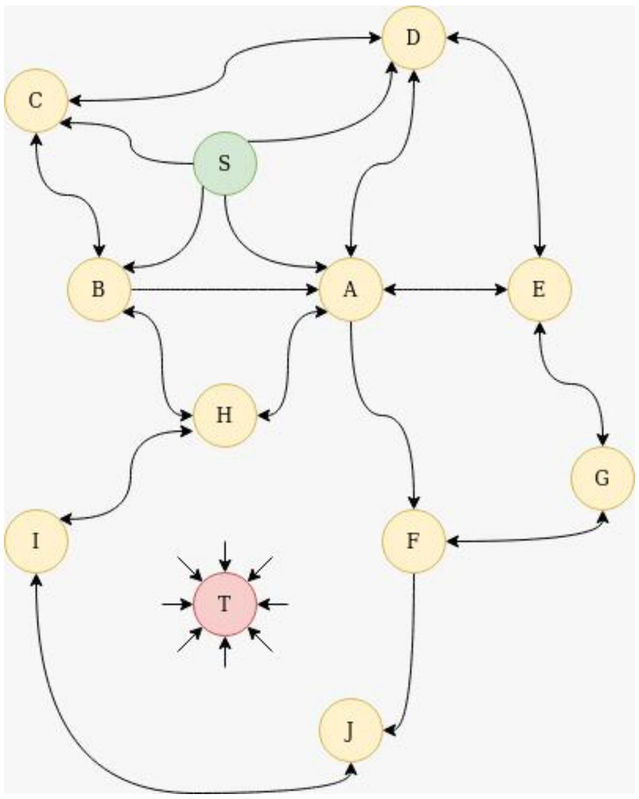
The Travelling Salesperson Problem (TSP) challenges finding the shortest yet most efficient route for a person to take given a list of specific destinations. It is a well-known algorithmic problem in computer science and operations research. There are a lot of different routes to choose from but finding the best one—the one that will require the least distance or cost—is what mathematicians and computer scientists have spent decades trying to solve.

TSP has commanded so much attention because it is so easy to describe yet challenging to solve. TSP belongs to the class of combinatorial optimization problems known as NP-complete. This means that TSP is classified as NP-hard because it has no “quick” solution, and the complexity of calculating the best route will increase when you add more destinations to the problem.



3. Implementation Details

As we provided a code baseline, you must be strict on this baseline. You are not allowed to propose any other implementation for this assignment. Node and Algorithm implementations are given to you to understand what they do sincerely, and you are not allowed to change these files.



Also, you will have a runner script named "main.py", which is responsible for running your script and making it possible to select the algorithm that the user wants to apply to the problem. It would be best to understand your JSON data file and your map deeply. Each file starts with S and ends with T. You must find your best path solution for each search method. Then you can report your homework like an example report file.

You need to implement your algorithms on the "run" method of corresponding class files (*e.g.,* UCS.py and AStar.py) and report your results accordingly. Please first try to understand the main structure of the environment and then start to implement the algorithms. Please do NOT change any code block unless it has the comment line as

“You should implement inside of this method!”

If you are not comfortable writing code in Python, you may follow this tutorial to improve your Python programming skills. Even if you do not know anything about Python, you only need 2-3 days to learn Python from scratch to complete this assignment. If you have any questions about the implementation, please do not hesitate to e-mail the assistant.

4. Criterion

- **Runtime:** Please do NOT include any third-party libraries in your project because you do not need them. Including different third-party libraries may cause a problem for running your code in different local machines (*e.g.*, dependencies etc.), for any dependency on a runtime that the assistant can solve -20 points penalty, for any dependency on a runtime that the assistant cannot solve -75 points penalty.
- **Compilation:** Please make sure that your code is compiled correctly. Not compiling code (*e.g.* any error etc.) -75 points penalty.
- **Report:** You need to write a report to explain your design in detail. One page code explanation is not a report, and such reports will be ignored. Please follow the rules of technical report writing like the example report (*e.g.*, Introduction, Algorithms, Implementation Details, Results, Conclusion etc.).
- **Submission:** All project files (*.PY) and report file (.PDF) should be zipped (.ZIP) together, and the filename of the ZIP file should be in the format of “NAME_SURNAME_ID_hw1.zip”. Please follow this structure in your submission. Not following -10 points penalty.
- **Deadline:** 27 March 2022 – 23.55 via LMS. **(Strict!)**
- **Group effort:** You may discuss the algorithms and so forth with your friends, but this is individual work, so you must submit your original work. **In any circumstances of plagiarism, first, you will fail the course, and the necessary actions will be taken immediately.**

Grading Criteria:

- UCS: 35
- A*: 40
- Report: 25