**CS 455 – Module 1 Homework – Search Algorithms**

This assignment can be completed individually or with one partner. It is strongly encouraged that CS 455 students partner-up with students in the CS 455 section, and likewise, the CS 595A students partner with CS 595A students.

Submission Criteria:

* Submit your work as a zip file containing
  + Solutions to written exercises
  + Source code for implementation exercise
    - Jupyter notebook with Python code (recommended)
    - Or, Python source code only
* If you are working with a partner, you must indicate clearly in your assignment with whom you are collaborating.

Problem 5 is worth 20 points, but is only assigned to CS 595A students. CS 455 students will be graded out of 100 points. CS 595A students will be assigned a grade out of 120 points scaled to 100 points (recorded grade will be 100\*hw\_grade/120).

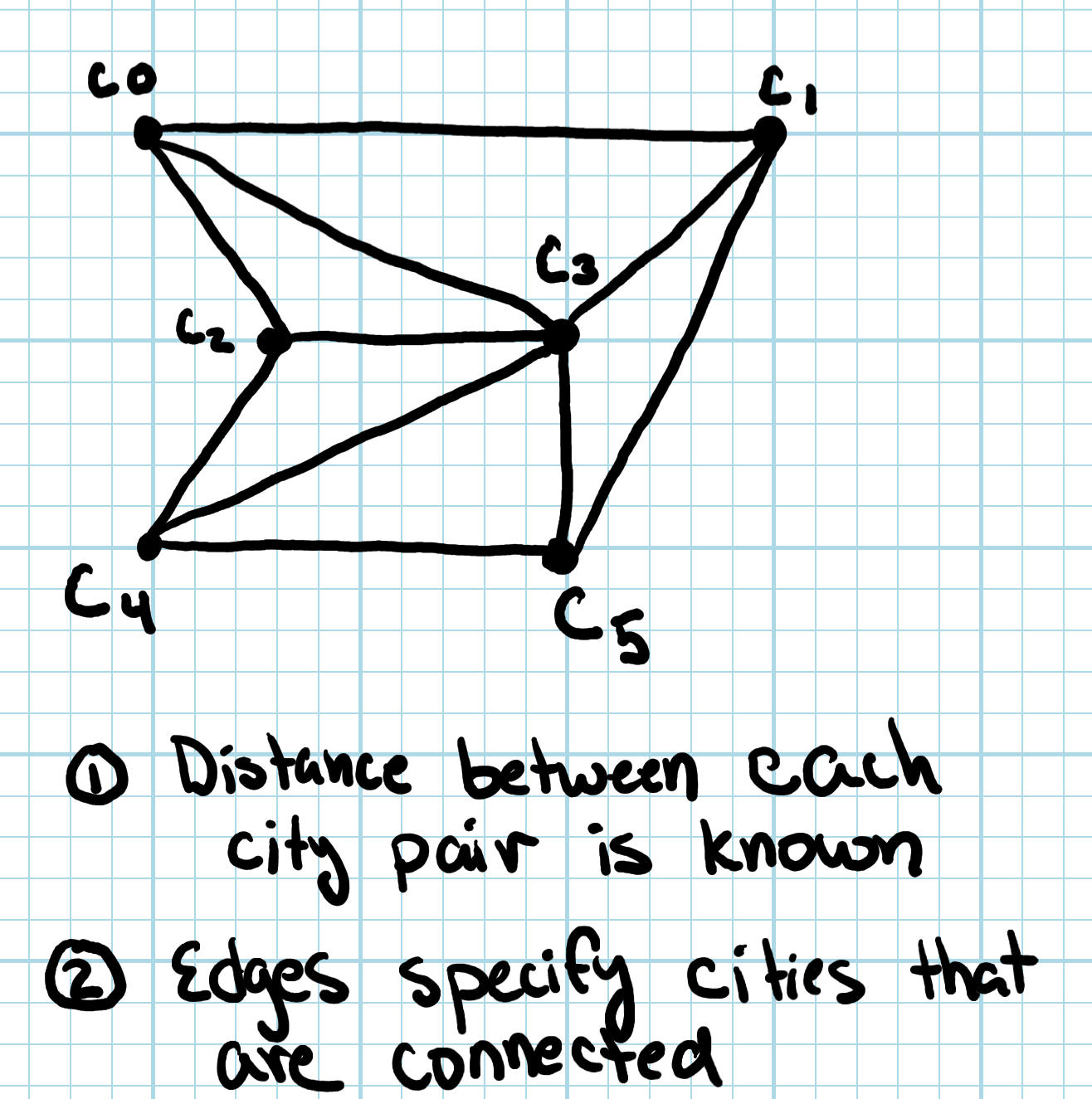
**All Students:**

**Problem 1: Demonstrate understanding of search algorithms [20 points]**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | | **Start** |  |  |  |  |  | **Goal** |   For heuristic use Manhattan (city-block) distance   * Initial State: see above * Goal State: see above * Actions: Move Up, Down, Left, or Right * Path Cost: length of path | 1. Trace the execution of the **Depth First Search Algorithm**. 2. Show your search tree as you go.    1. Update the closed and open sets as you go.    2. When you are finished, write out the final path 3. Solve the problem using Greedy **Best First Search**. 4. Show the search tree. 5. Indicate for each search tree node the h(n) and f(n) values and its parent 6. Write out the final solution and the total path cost. 7. Solve the problem using **A\* Search**. 8. Show the search tree. 9. Indicate for each search tree node the h(n), g(n), and f(n) values and its parent    1. Hint: You may want to do this using a table as shown in class to track h(n), g(n), f(n), and parent.    2. If any of these values change, cross out the original value with a single line and write in the new value. 10. Write out the final solution and the total path cost. |

**Problem 2: Problem Formulation for Informed Search [20 points]**

Consider the traveling salesperson problem (TSP) with no repeats. The environment is a set of cities with connected edges representing roads. This shall be represented by a complete graph as shown. The TSP algorithm with the no-repeats variant seeks to find the minimal path with the same start and end city that visits each city once.



If we were to apply A\* to solve this problem given a map, start location, end location, and the direct distance between each pair of cities (ignoring edges and considering a straight-line distance), how would you formulate this problem?

Your answer must address:

* How is a state within the algorithm represented?
* What is the successor function?
* What is an admissible heuristic?
* How would you define the path cost?
* What are the goal criteria?

Your answer should be sufficiently descriptive such at the reader could implement the formulation. You are free to make use of diagrams, pseudocode, text, etc. to convey your problem formulation.

**Problem #3: Local Search Problem Formulation [20 points]**

The knapsack problem is another classic AI problem.

Consider a knapsack that is designed to hold up to some maximum weight capacity. Within the knapsack, you wish to place items that have associated weight and associated value (i.e. each item j has weight wj and value vj). The goal of the problem is to maximize the value of the knapsack’s content without exceeding its weight capacity given some collection of potential items to add.

Since this is an optimization problem, a local search would be a potentially good algorithm choice, but you need to define the problem formulation. How would you formulate this problem?

Your answer must address:

* How is state represented?
* What is the successor function?
* What is the value / reward that you are seeking to optimize?
* Is this a maximization problem or a minimization problem?

**Problem #4: Implement the Map Coloring Problem [40 points]**

In class, the map coloring problem was shown to demonstrate several search algorithms. For this problem, you will implement code in Python to solve this problem using BFS, DFS, and IDFS.

You are free to use any of the source code provided by Dr. Stansbury, as shared on Canvas via link to Github. The key task of this exercise is to demonstrate that you can design and implementation a problem formulation for a new problem.

You are encouraged, but not required to implement your program within a Jupyter notebook.

Your algorithm must take in as an input the following representation of a map. This defines a list of tuples where each tuple describes region adjacencies. Example input:

# adjacent regions within the map

*map = [(“r1”,”r2”),(“r1”, “r3”), (“r2”, “r4”), (“r3”,”r4”)]*

Your algorithms required output should be a print out of each region’s color:

*Results:*

*R1 = Red*

*R2 = Blue*

*R3 = Green*

*R4 = Red*

Recall, the goal of the map coloring problem is to assign a color to each region such that no two adjacent regions have the same color and only the minimum number of colors are used.

Your final submission should demonstrate the execution of each of the search algorithms on an input matching the example above.

Your code should be correct and well-commented to ensure maximum credit.

**CS 595A Students Only [20 points]:**

Problem #5: Perform a brief literature survey to find a research paper that describes a real-world application of search (using one of the algorithms discussed in class, preferably), and summarize the paper addressing the following:

* Problem being solved
* Type of search(es) used: uninformed, informed, local
* Describe the problem formulation.
  + In your answer, address the question: Was their problem mapped to a classic AI problem, or did they come up with a novel formulation for their problem?
* Identify the search algorithm used and identify if any variations of the algorithm were used compared to what was presented in class.
* What experiments were performed by the researchers?
* Briefly summarize the results

This should be about 1 – 1.5 pages of content.

Cite the referenced paper using the IEEE Citation Style (https://ieee-dataport.org/sites/default/files/analysis/27/IEEE%20Citation%20Guidelines.pdf).