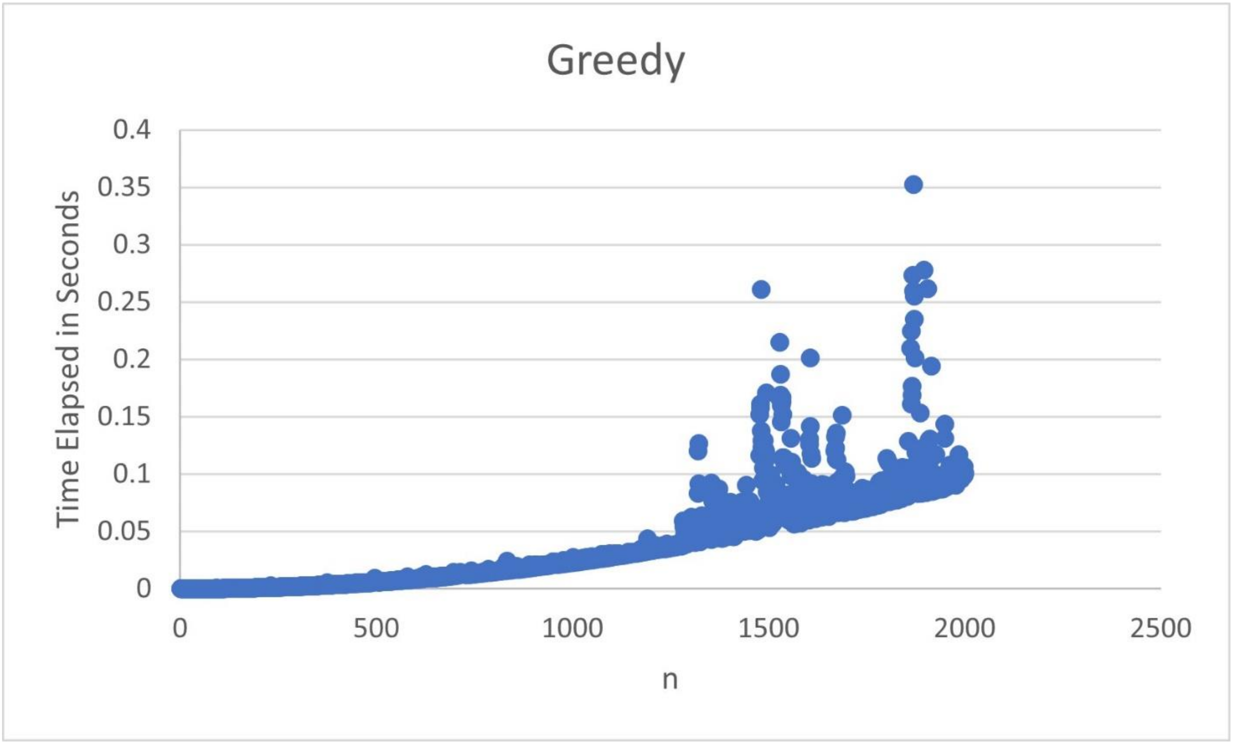
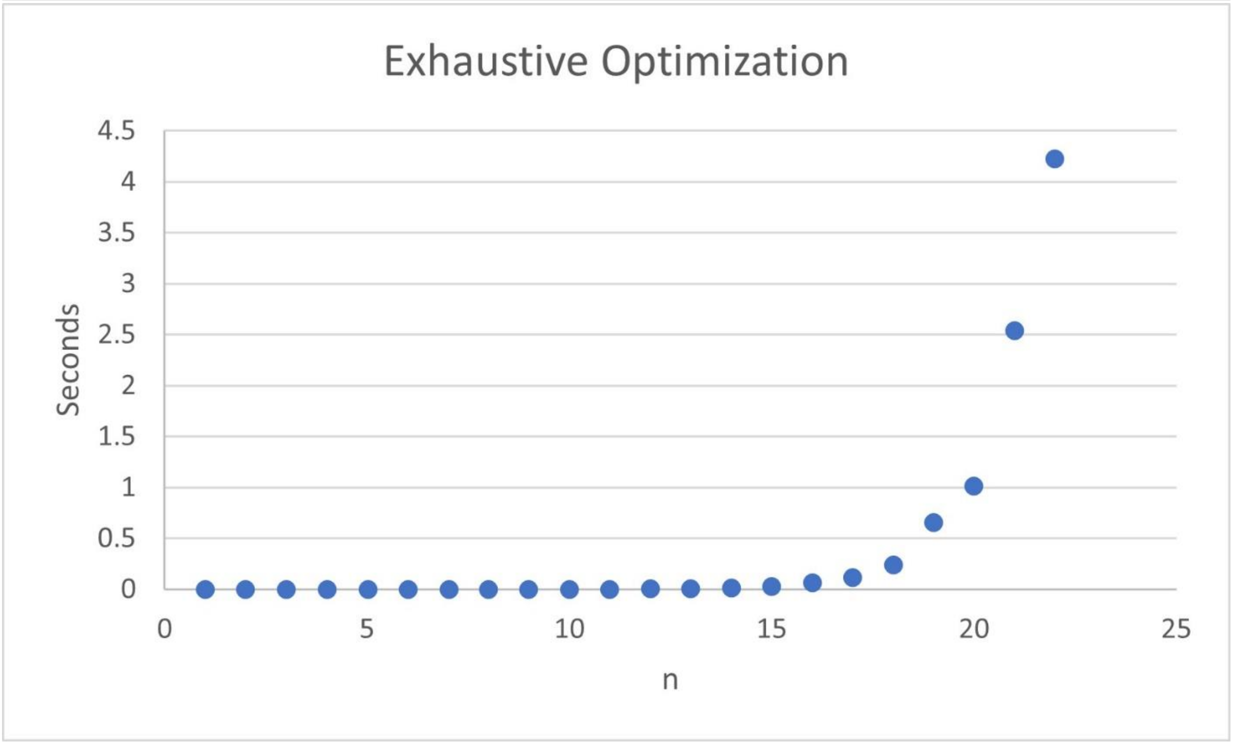
Daniel Cao

Dcao182@csu.fullerton.edu

Project 2 Report

2.





3a. The exhaustive optimization algorithm has a noticeable difference in performance compared to the greedy algorithm. From what I notice from the scatter plots, the greedy algorithm is able to process more n elements in less time than the exhaustive optimization algorithm. It looks like the greedy algorithm faster because according to the scatter plot, the greedy algorithm is able to process thousands of elements in less than a tenth of a second. The exhaustive optimization algorithm only processes about 20 elements in about one second. Since I understood that exhaustive search algorithms were slow, the differences in the performances of the two algorithms didn't surprise me. I was surprised by how the scatter plot for the greedy algorithm turned out. I didn't imagine it would turn out the way it did. In the greedy algorithm, I noticed how the algorithm selects less optimal elements as it processes more elements. Therefore, the exhaustive optimization algorithm may be more accurate at selecting elements.

3b. My empirical analyses are consistent with my mathematical analyses. The complexity of the greedy algorithm being in O(n^2). The complexity of the exhaustive optimization algorithm being in O(2^n * n). O(n^2) is more efficient than O(2^n * n) because O(n^2) processes more n elements in less time. O(2^n * n) processes less n elements in more time elapsed. Both of these are consistent with the scatter plots shown above.

3c. I think this evidence is consistent hypothesis 1 because exhaustive search algorithms are easy to implement and reliable at producing correct outputs. They are easy to implement. However, the space complexity may be an issue to keep track of all candidates. Therefore, exhaustive search may be too slow for practical use for a problem with larger number of elements. Exhaustive search algorithms can be as slow as O(2^n) or O(n!).

3d. I think this evidence is consistent with hypothesis 2 due to the cost of generating all candidates. Although algorithms like exhaustive search are easy to implement and provides correct output through generating and checking candidates, it is impractical to use an algorithm as slow as exhaustive search on any problem. Ultimately, the cost of generating candidates will cause an issue with the space complexity, especially with problems with a larger number of elements to generate and check for candidates.