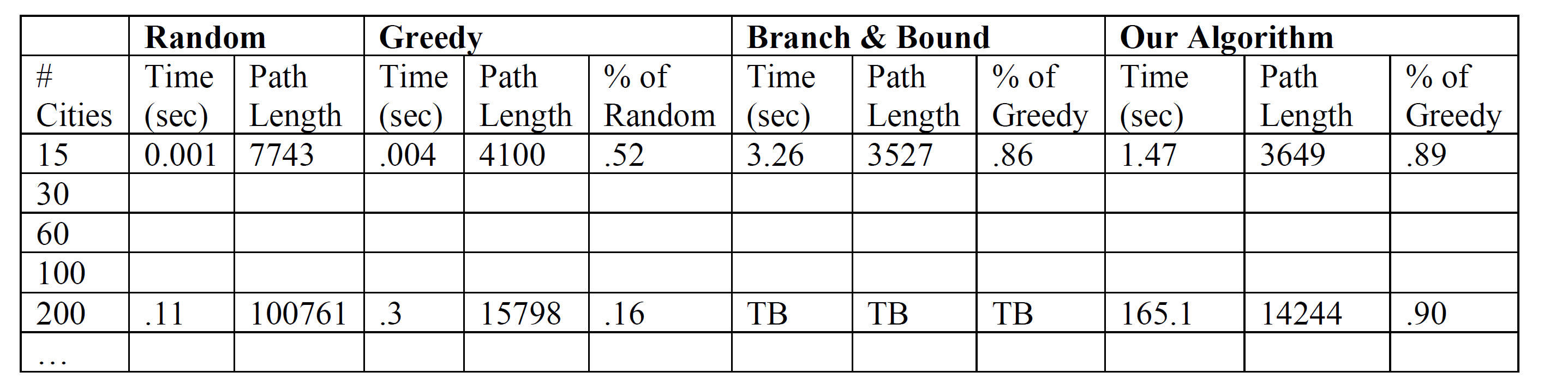
**Report:** There is no graded design experience for this project, but your group should do the design experience together before starting to code.

1. [10] Correct implementation of the greedy TSP algorithm. Brief discussion **and complexity** of the algorithm.

2. [35] Correct implementation of your own TSP algorithm including a discussion of the algorithm, why you chose it, and its pros and cons. Give proper attribution (references) for ideas you find externally. Clearly point out which ideas came from other sources, and which ideas are original contributions that you made. Include a discussion of the theoretical big-O complexity of your algorithm. Discuss how the empirical complexity matched your theoretical complexity. Include screenshot examples of typical results for your algorithm.

3. [30] Include a table with columns (see below) for each of the TSP algorithms including the Branch and Bound TSP algorithm you implemented in Project 5 (use one of your individual project 5 implementations as a representative version of B&B; however, do not set your time limit at 60 seconds as you previously did). For the random algorithm, use the default algorithm provided with the original framework (this acts as a baseline for the greedy algorithm). You can play with all three levels of problem difficulty (Easy, Normal, Hard) during testing, but just use the Hard level for all of your reporting. For the greedy algorithm report the improvement over random as a fraction [calculate this as greedy\_cost/random\_cost]. This will help calibrate and make sure your greedy algorithm is implemented correctly. For B&B and your own algorithm, report the improvement over the greedy algorithm [calculate this as your\_cost/greedy\_cost]. You should create a table just like the one below with the city sizes shown below (15, 30, 60, 100, 200). Round average tour lengths to the nearest integer. Round time and % improvement to two significant digits beyond the decimal. The results for each cell (all 4 algorithms) should be the average of 5 runs with different random seeds for that number of cities. Do not try to find a particular set of seeds on which your implementation does well. They should be randomly chosen runs. You will fill in average time (seconds) and average tour length for the different numbers of cities. If an algorithm takes more than “reasonable time” (more than about 10 minutes or so) to solve for that number of cities for the majority of your trials, then just fill in that cell for that algorithm with “TB” (Too Big). For B&B and your algorithm, if a minority of the trials do not finish, take the average of the majority that does. The numbers in the example table below are made up. (Note: We give random and greedy algorithms an unfair advantage in our comparison as we allow them an arbitrary number of restarts when they fail in the hard version, which we do not grant to B&B and your algorithm. This points out another way in which they are inferior to your algorithm and B&B.



4. [15] To complement the required five rows shown in the table above, you should also add to your table some additional rows (different numbers of cities). In particular go as large as you can before getting “Too Big.” The last row of your table should be the largest number of cities which can be done in a reasonable time limit. Discuss and analyze the results in the table. In particular, make sure you have a few rows in the "sweet spot" for your algorithm, which is around the number of cities where it shows the best balance of lots of cities with good costs. Discuss and analyze the overall results from your table.

5. [10] Your write-up should look sharp and follow the form of a brief conference paper with abstract, introduction, algorithm explanation, complexity, analysis of results, and future work.