14012402-4 Algorithms

PROGRAMMING PROJECT TOPICS

You are expected to select one of the programming topics described below or other programming proposals that you may have on your own, ideas need to have a very strong relation to the contents of this course.

Once the project is completed, include the following in your presentation and project:

- A demonstration of your project in which you show the various features of your system. We will examine your code to check for code quality, code documentation, etc.
- You should also hand in a completed project report, which is essentially a polished version of the project design document, but should also include some experimental results, e.g., charts of running time versus input size, etc., using applications like Microsoft excel, tableau and etc.
- You should also turn in your code and associated documentation (e.g., README files) so that everything can be backed up for future reference.
- submit your code and all associated files to the blackboard

Project Topics:

- **Design data structures:** present five data structures including B-tree for **a very large social network** (data given), to show the shortest path and Minimum Spanning Trees between two people (e.g., Me→ Ameerah→ Fatimah→ Amal→ You). You can also make an application like Google Maps. In this project, you will implement and analyze the following.
  - The Dijkstra's shortest path algorithm, Bellman-Ford Algorithm, Floyd-Warshall Algorithm, and Johnson's Algorithm, or BFS, DFS, Best FS, A* Search, and bidirectional breadth-first search, or Prim's minimum spanning tree algorithm, Kruskal's Algorithm and Boruvka's Algorithm. Assume your data structure (i.e., adjacency list) represents the input graphs. Implement the data structures and compare them in terms of time complexity.

- **Design data structures:** present five data structures including B-tree for **big data** (you find data), for sorting algorithms. In this project, you will implement and analyze the following.
  - **Median finding, Order Statistics, and Quick Sort:** In this project, you will implement the median-finding algorithms. The user should be able to select the "k", i.e., the rank of the number desired as output (k = n/2 is the median). You should also be able to select groups of 3, 5, 7, etc. in the linear-time median finding algorithm and be able to compare the performance of each. Also, implement the randomized median finding algorithm and compare it against the linear-time one. Implement quick sorting using both algorithms and compare the performances of these different versions.

- **Design data structures:** present five data structures for **big data** (data given), for string matching algorithms. In this project, you will implement and analyze the following.
  - **String Matching algorithms:** Implement both the KMP pattern matching algorithm as well as the Longest Common Subsequence (or Edit Distance) algorithm. Your algorithm should be tested on real datasets, e.g., text files, biological sequence data, time series databases, etc. Compare the performances of each algorithm against naive algorithms for the same problems.
    Data

- **Design data structures:** present five data structures including a tree for **big data** (data given), for computational geometry algorithms. In this project, you will implement and analyze the following.
  - **Convex Hull**: Implement several versions of the convex hull algorithm: (a) the divide and conquer version, (b) the Graham Scan version, (c) the version in which a preprocessing step removes all points in the extremal quadrilateral, and (d) a convex hull algorithm when the input is not a set of points but is a nonconvex polygon. Have a GUI where one can insert new points (for (d), the user can add a new vertex to the input polygon) and the application recalculates the hull. Compare the performances of all algorithms. data

- **Design data structures:** present five data structures for **big data** (data given), for Network flow algorithms. In this project, you will implement and analyze the following.
    - **Network Flow:** Implement the Ford-Fulkerson algorithm, Edmonds–Karp, Dinic's and Dinic's with dynamic trees for computing network flow in bipartite graphs. Implement the data structures and compare them in terms of time complexity. Compare the performances of each algorithm.

Note: clear report, pseudo code, and code with comments and a clear analysis of your algorithms will be considered in the final grades.

Team Size: Max 5 members.

Due Date: any day before discussion day on February 5th, 2023