a place of mind
THE UNIVERSITY OF BRITISH COLUMBIA

COSC 320 – 001 [3-2-0]
Analysis of Algorithms
2020 Winter Term 2
Project Topics

There are a few topics that you can choose to work on as your group project. You are encouraged to read all of them and think about the skills you want to gain out of each project. Discuss these points, and then choose **ONE** project to work on as your term-group-project.

No matter which project you choose, you should complete the following steps:

1) Problem formulation:
   Formulate the problem as an algorithmic problem.
2) Algorithm design and analysis:
   There is a straightforward solution to the problem you are trying to solve. Let's name it algorithm A. This algorithm will have at least $O(n^2)$ (or more depending on the project $O(n^3)$, etc).
   You are required to design a more efficient algorithm (call it B), prove its correctness, and compare (using the asymptotic notation) the time complexity of the two algorithms.
3) Implementation and Empirical Evaluation:
   Implement your algorithm B and the algorithm A, and perform simulations to empirically compare their running times. Your implementation shall have a class named ProjectName with two public methods
   algorithmAforProject(X)
   algorithmBforProject(X)
   Both methods shall return the required results.
4) Clearly explain your choice of data structure to implement your algorithms. Reason about why you choose this data structure over others and what are the pros and cons. In some projects, you should think of searching with O(n) when one data structure is used, compared to non-linear time using another one.

Note: In all cases, you can download some real datasets and test as an extra step. This can motivate you about the real applications of the course. Finding the proper dataset or generating synthetic dataset is part of the project.

Note 2: You can implement your project in a programming language of your choice: java, python, or C++. Matlab code is not acceptable.

Note 3: All implementations should have clear test sets and plots showing the comparison of the running times of the designed algorithms.

Note 4: If any online resource is used as a helper, you should cite it properly, or the project will receive a 50% penalty immediately.

Note 5: All milestones of the project should be delivered clearly and with proper explanations and rationales.

Note 6: A peer-review will be given at the end, for each group, meaning that all group members should give a grade to all other group members including themselves. This peer-review shows how each person has collaborated to complete the project. If all group members give a low collaboration mark to one person, this indicates the person has not put enough effort to complete the project. In this case, the grade of the individual will be lower than his/her group members significantly. The weight of peer-review marks will be decided later.

Note 7: Tentative: Consider your projects as a competition or hackathon. There are multiple groups who choose the same topic. We can choose a dataset for each topic and run all the designed algorithms to compare their run time.

Evaluations:

20 points for problem formulation, a pseudo-code description of the algorithms.
20 points for the formal analysis (including correctness proof and running time).
20 points for the choice of data structure and analyzing/reasoning about it over other options.
40 points for implementation (correctness and efficiency). Your implementation will be tested on your TAs laptops. Your algorithm will be executed for various number of inputs, from 1,000 records to half a million (at least). Your implementation should handle all cases within a few seconds/minutes.

Milestones:

Project proposal (5 marks): You should describe your problem in more details (include and example that shows your understanding), edge cases, and any expected complexities. You also need to describe your plans of obtaining a real dataset or generate a dataset which enables you to test your dataset. Your choice of programming language, timelines, and the separation of tasks among all group members are part of this delivery. Note that for some topics, you should think about the operations that should be done.

First milestone: problem formulation and pseudo-code for your algorithms. This also includes the analysis of the algorithms, proof the correctness and running time of at least one of the algorithms.

Second milestone: Includes the analysis of the second algorithm, correctness and run time proofs. Note that at this point you should have submitted the algorithm analysis for both algorithms. Description of the data structures that you will choose and the rationales.

Third milestone: Implementation of the first algorithm. You need to submit the test cases and plots showing how it runs as the input grows.

Fourth delivery: Implementation of the second algorithm, all tests, and plots. This is the final delivery of your project. You also need to include a 10 minute video showing your choices, designed algorithms, and running your implementations.

Hacking:
An implementation without a clear and formal formulation, design, and analysis, is only worth 40% of the total marks, even if your program runs well.
An implementation that crashes will get a 30% penalty.

Project Topics

Topic 1) Keyword replacement in a corpus: In text analytics, often it is required that a set of keywords are replaced with a given set for the documents in hand. For example, on Twitter, people write a lot of abbreviations. When one requires to analyze the tweets, (s)he should find all the abbreviations in a given list of abbreviations (e.g. ASAP, won't) and replace all these brief terms in the tweets with its proper phrase/keywords (e.g. ASAP -> As soon as possible, or won't -> will not). Your job is to design an algorithm that finds all of the keywords that are in the abbreviation list in each tweet, and then replace them with the appropriate given keyword/phrase. The number of tweets can be millions and the list of keywords can be hundreds. A naïve approach is that for each tweet, your algorithm checks for all of the elements in the abbreviated list and replaces them. Other than the naïve approach, design a better algorithm and apply the required four steps explained in the first page.

Topic 2) In databases, you can apply a join on two datasets on a specific key. In this problem, you are required to design an algorithm for this task. Specifically, given two documents D1 and D2 (think about a csv file with multiple columns in each dataset), we need to include all the columns of D2 in D1, based on a specific key. An example of this is a csv file in which the rows are user feedbacks about various apps on Google Play Store. The other dataset, D2, the details of the apps are given. Your algorithm should be able to include the app details in D1 correctly, i.e. if a review is written for app a, the details of app a will be added to it, versus another review about app b that should be combined with the extracted details about app b. Design two algorithms to implement the solution to this problem. You are NOT allowed to use the available *join* functions from the available libraries. Apply the four steps of the project explained in page one.

Topic 3) A simplified explanation of a category of recommender systems is a search engine that searches for similarities (and dissimilarities) among users to recommend a list (of movies, items, news, etc.). Given a set of users and their lists of likes and dislikes, your task is to compute the similarity among users, then search for similar users to a given user U. Finally, generating a list of items that is not rated by user U and recommend items to U based on the items that are liked or disliked by the other users similar to U. Note that not all of the available items in the items-list is rated as like or dislike by each user. There can be millions of items and each user might have a list of at most a few hundred ones that (s)he likes/dislikes.
A simple formula to compute the similarity among users U1 and U2 is:

$$s(U1, U2) = (|L1 \cap L2| + |D1 \cap D2| - |L1 \cap D2| - |L2 \cap D1|) \div |L1 \cup L2 \cup D1 \cup D2|$$

Where L1 is the list of items liked by user U1, D1 is the list of items disliked by U1, similarly L2 and D2 are the list of items liked and disliked by U2. The ∩ indicates the intersection of the two sets (the shared items) and ∪ indicates the union. In the union, the shared items are counted only once.
Apply the four steps explained in page one for your project. As a dataset, you can also download a movies dataset from IMDB or the item recommendations by Amazon. The datasets are freely available.
The detailed definition of recommender system and threshold are left to your discretion.

Hint: Look at the counting inversions algorithm from Algorithms Illuminated or Algorithm Design books.

Topic 4) String Matching is a problem used in many applications such as plagiarism detection. In this project, you are required to implement a plagiarism detector. Your algorithm will receive a corpus of existing documents and a potentially plagiarized document as input. The output should be the set of documents from which the document was plagiarised from. You should implement the KMP, LCSS, and Rabin-Karp fingerprints algorithms. You need to read about some of these algorithms if we don't discuss them in class.
a) KMP : Treat each sentence in the test file as a potential pattern. Search for the pattern in the existing documents

and find the matches.
b) LCSS : Run the LCSS algorithm for each paragraph from the test file and existing corpora.
c) Rabin-Karp fingerprints: Implement Rabin-Karp algorithm for multipattern matching. You can use the language libraries for converting string to hash.
The definition of plagiarism and threshold are left to your discretion.

Topic 5) In this problem, you will analyze the best route from source A to destination B in a flight network where there is no direct flight from A to B. The task is to identify the best route, where best route is identified by a few parameters as follows. A customer requires the cheapest flight, with minimum wait time in all connection airports (the summation of the wait times in all connection airports), and minimum flight duration. Therefore, the customer can go from A to B as fast as possible with the minimum cost. For sample data, you can see some real statistics/data from Bureau of Transportation Statistics. Apply the four steps as described in page one.

Topic 6) A day trader makes money by buying and selling shares of stocks within the same trading day; At the end of a trading day, he has to sell all the shares he bought on that day and makes a profit. A negative profit is a loss. The performance of a day trader over a time period (a list of consecutive trading days) is measured by the sum of his profits over these trading days. Given his complete trading record (i.e., the profit on each trading day), the strength S of a day trader is the maximum performance over all possible time periods. Your algorithm should compute a day trader's strength. Depending on the design techniques you use (divide-and-conquer, dynamic programming, etc), quadratic-time, O(n log n)-time, and linear-time algorithms are all possible. In fact, the idea behind a quadratic algorithm is more about one's programming skill - for a good programmer, the most natural way to implement algorithm A shall lead to a quadratic-time algorithm. Follow the four steps in page one to solve this problem. (Acknowledgement: Topic 6 is designed by Dr. Gao).

Note: sample datasets for topics 1 and 2 can be provided.