Masai C++ Projects

(You supposed to use STL, Algorithm, FILE IO of C++ to Solve the Projects)

Project I: Similarity Checking from Books

Write a C++ Program to identify 10 similar pairs of text books from a given set of 64 text books
in txt format.

A simple (but crude) way to measures similarity is given as following:

Identify top 100 frequent (normalize) occurring words in a text file. Consider only numeric character and alphabet character (by converting both small and capital alphabets to capital form) of the text file. Normalize the word count of a file by dividing total number of words in the file.

- remove commonly occurring six specific words “a”, “and”, “an” , “of” “in” and “the” from the file at the time of counting frequent word, total number of word and the normalize frequency.

Similarity index of two files can be calculated as sum of normalize value of all the frequent words occurred in both the files.

Create a 64x64 size similarity matrix.

Report top ten similar pairs of text books using the similarity matrix. You need to exclude the self similarity.

Data Set:
https://drive.google.com/file/d/1jcm_-q3IzrDD7ICg0yMlH_GJbg8wudir/view?usp=sharing

Project II : TSP using SA

Write a C++ Program to find optimal TSP (Travelling Sales Man Problem) tour using Simulated Annealing Meta Heuristics.

TSP Tours: Given a collection of cities and the cost of travel between each pair of them, the traveling salesman problem, or TSP for short, is to find the cheapest way of visiting all of the cities and returning to your starting point. In the standard version, the travel costs are symmetric in the sense that traveling from city X to city Y costs just as much as traveling from Y to X and in this case it is Euclidian distance between two city X and Y. This problem is know to be difficult problem
in ter of complexity and people use heuristics/meta heuristics to solve this problem.

A solution S to problem have all the city in a sequence (in a array with out repetition) and the solution is optimal if the solution have least cost.

Simulated Annealing for TSP: A neighbour solution S’ of S have an incremental one swap of two city position in the solution array as compared to S. Simulated annealing (SA) is a probabilistic technique for approximating the global optimum of a given function.

Specifically, it is a metaheuristic to approximate global optimization in a large search space for an optimization problem. SA start with initial random solution S; in each iteration it generate a neighbour solution S' with one swap of two cities position in the solution, the new solution is accepted if cost of S' is lower as compared to S, otherwise the solution S' will be accepted with a probablity (high in lower iterations and low in higher iterations).

Run simulated annealing for 10000 iterations,

Details of SA: can be found at : https://en.wikipedia.org/wiki/Simulated_annealing

Input Data :

Data Set: http://www.math.uwaterloo.ca/tsp/vlsi/index.html , xqf131.tsp, xqg237.tsp, ..

The data set contents the points and their location. Distance between two points is Euclidian
distance.

Output: final achieved cost and Solution [sequence of cities/numbers)

Projects III: Job Scheduler design and implemention in C++

Write a C++ Program to Simulate Job Scheduler for in a Supercomputer.

Suppose the supercomputer has one master scheduler node and 128 worker nodes. All the worker nodes have 24 cores and 64 GBs of RAM. All the jobs from the users get queued up at the master node and the master node schedule the jobs on to the worker node. Each job is specified with arrival time, the number of cores required to execute the job, amount of memory in GB require to execute the job, and execution time (in the number of hours) of the job.

The master node accumulates all the jobs reached to master node at beginning of current hours and all the pending older jobs (which are in the queue but not yet allocated to any workers) and try to allocate worker node for each of the jobs one after another in order. Worker node allocation for a job is successful if the master node finds a worker node with a higher amount of available resources than the required resource by the job. Otherwise, the job gets reinserted into the queue.

Master node use three policies to put the Jobs into the queue :

(a) FCFS (b) smallest job first queue, (c) short duration job first queue.

Short duration job means a job with a smaller execution time and the smallest job means a job with the smallest gross value (=execution time * CPU required * Memory required). Master node use three policies to find a worker node for the job and these policies are (a) first fit, (b) best fit, and (c) worse fit.

Simulate the job scheduler for the above queuing policies and node selection policies. Generate average CPU, Memory utilization per day basis for each combination of policies,

and show the same in graphical format (putting the same in a CSV file and general bar graphs).

Assume each job behaves ideally as specified, which means each job consumes the specified amount of resources (not less and not more) and takes the specified amount of time to execute (not less and not more).

Data Set: https://drive.google.com/file/d/1nk2vwcl0BHpmQDdOXVh2iBSaA63kcVQl/view?usp=sharing