COMP 9102 Data Management and Information Retrieval
Assignment 3
Top-k and Skyline Queries
Due Date: Nov 30th, 2022, 5:00pm

At Moodle, you will find the data that you are going to use in this assignment. These data come from a website which keeps statistics about NBA matches (https://www.basketball-reference.com) and they were downloaded from https://www.kaggle.com/drgilermo/nba-players-stats and have been post-processed. You are going to use files which record statistics about basketball players that participated in NBA matches in 2017. In file 2017_ALL.csv, you can find information about the players (name and team) and some statistics: total rebounds (TRB), assists (AST), steals (STL), blocks (BLK), points (PTS). For each one of these 5 statistics (TRB, AST, STL, BLK, PTS) there is a separate file, which has all players sorted in descending order of their performance in that statistic. For example, in file 2017_TRB.csv, player with id 138 has collected the largest number of rebounds (1116), then player with id 294 follows with 1114 rebounds, etc.

Part 1: Top-k queries

Write a program which implements the NRA algorithm that finds the top-k players based on some of the 5 statistics. You are going to use as inputs the sorted files 2017_TRB.csv, 2017_AST.csv, etc. Specifically, from the **command line**, the user of your program will specify in an array the statistics categories that he is interested in and the number k of top players. For example, by giving as command-line arguments [2,5] and 10, the user requests for the top-10 players based on assists and points.

The aggregate function should be the **sum** of the **normalized scores** in each of the selected statistics. For example, if the first parameter of the program is [2,5], then the score of each player in statistic 2 (assists) will be the number of assists by this player divided by the maximum number of assists by any player (which is 906 and you can find it at the first line of file 2017_AST.csv, since the file has the players sorted in descending order of their assists). Similarly, the score of each player in statistic 5 (points) should be the number of his points divided by the maximum number of points by any player (which is 2558). By adding these two atomic scores, we obtain the total score of the player in both statistics. We are looking for the k players with the largest total scores.

Your program should follow the logic of NRA algorithm. This means that it should read concurrently all relevant files (e.g., files 2017_AST.csv and 2017_PTS.csv in our example) and it should keep for each accessed player the lower bound of its total score in a hash table. After each round of accesses, you should maintain the set of k players so far based on the lower bounds of their total scores. It should check whether the top-k set can be finalized (i.e., whether there is no chance for any other player to be part of the top-k result). You can use the threshold T to reduce the necessary comparisons during the growing phase of the algorithm. Your program should print at the output the top-k results together with their total scores and, in addition, the total number of lines which have been read from the files until the top-k result is finalized (number of accesses).

To verify the correctness of your implementation, you can also implement a baseline top-k algorithm, which reads file 2017_ALL.csv to compute the total score of each player and keep track of the top players with the largest score. If your program is correct, it should give the same result as the baseline.

Part 2: Skyline queries

You are asked to implement a main-memory skyline algorithm, which follows the logic of BNL algorithm in the lecture notes. Write a program, which takes as command-line argument the statistics that we are interested in (e.g., array [2,5] denotes that we are interested in assists and points) and computes and prints the skyline of players which are not dominated by any player in all specified statistics (e.g., the skyline based on assists and points).

Your algorithm should read the players from file 2017_ALL.csv and maintain the set of players S who are in the skyline so far. For each player p read from the file, the program should check if p is dominated by any player in the skyline S so far (in this case, p is ignored). If p is not dominated, then p should enter S and you should remove from S any player in S which is dominated by p. Assume that the program's memory is large enough to accommodate S (hence, there is no need to perform multiple passes and create temporary files).

Submission requirements

You should submit your program(s). You should also write a short report about your implementation, in case it is hard for us to understand your program. Your report should include some sample output of your program. Your report should also include any comments about the evaluation and any other observations that you have. Submit the report together with your code at the course website. Your code should be compilable without problems and you should include basic instructions on how to compile and use it. Your program can be written in your preferred programming language (e.g., C, C++, Java, Python, etc.). Your program must run within reasonable time.

Please submit your assignment (one **ZIP** file) to Moodle on or before 5:00pm, Nov 30th, 2022. Make sure all contents are readable.

Please feel free to post your questions on **Moodle forum**, or contact the TA of the course if you encounter any difficulty in this assignment. We would be happy to help.