ASSIGNMENT
TCP2101 Algorithm Design & Analysis
- Trimester 2 Session 2016/2017 -

GUIDELINES & INFORMATION

Assignment Mark: **20%**
Submission Due: **5 Feb 2017 (Sun)**
Type: **Max 2 members per group**
Instruction: **All programs must be implemented in C++**
Expected deliverables: **one zip file containing both working source code and report. File name shall be ID1_ID2.zip.**
Submission format: **Upload to your lab section in MMLS**
Presentation: **20 minutes (describe the algorithm(s), report and discuss experimental results, conclude the results)**

Registration of Assignment Question

1. Each title can only be registered up to a certain number of groups per tutorial section depending on the class size, and the registration shall be done on first-come-first-serve basis.
2. Contact your lab tutor for registration of question.

Question 1

Perform a comparative analysis between two search algorithms, i.e. search by hashing and binary search. The two algorithms are used to search for a person's IC (Identity Card) number in a dataset with size 100,000 (or more). The experiments should cover the following aspects:

1. Generate a dataset with at least 100,000 unique IC numbers (2m)
2. Implement BST (preferable AVL) for binary search (2m)
3. Implement hash table with chaining for search by hashing (2m)
4. Implement good hash function by using prime number and not closed to the power of 2 (2m)
5. Correct implementation. Output result for the small size example. This is for lecturer/tutor to inspect the correctness of algorithms [2m]
6. Perform numerous random searches using the two algorithms, and get the average times for both algorithms [3m]
7. In the report, include the experiment results that can be used to perform a comparative analysis between the two algorithms. [2m]
8. Conclude your finding(s) in the report. [1m]
9. Presentation. [4m]

Question 2

Perform a comparative analysis between two collision resolution algorithms in hashing. The two algorithms are chaining and open addressing (linear probing, quadratic probing, etc.) The hashing is performed on a dataset that contains at least 100,000 unique URLs (such as www.mmu.edu.my). The experiments should cover the following aspects:

1. Generate a dataset with at least 100,000 unique URLs (2m)
2. Implement hashing with chaining (2m)
3. Implement hashing with either linear probing or quadratic probing (2m)
4. Implement good hashing by using prime number and not closed to the power of 2 (2m)
5. Correct implementation. Output result for the small size example. This is for lecturer/tutor to inspect the correctness of algorithms [2m]
6. Perform numerous URL searches on the two hash tables, and get the average times for both algorithms [3m]

7. In the report, include the experiment results that can be used to perform a comparative analysis between the two collision resolution algorithms [2m]
8. Conclude your findings in the report. [1m]
9. Presentation. [4m]

Question 3

Perform a comparative analysis of Depth-First Search (DFS) between two implementations of Graph ADT: adjacency matrix and adjacency list. The experiments should cover the following aspects:

1. Correct implementation. Output the result of DFS for a graph of 8 nodes for lecturer/tutor to inspect the correctness of algorithms [3m]
2. Different graph sizes (10,000, 100,000, etc.) [2m]
3. Different graph patterns (dense vs sparse) [2m]
4. Different cases (best, average & worst) [3m]
5. In the report, show your findings by plotting the results in graphs (running time vs input size). [3m]
6. Conclude your findings in the report. [3m]
7. Presentation. [4m]

Question 4

Implement the fractional knapsack algorithm using heap-based priority queue. (You should not use the code in Lab 8 as it is not based on heap-based priority queue.) This implemented program should cover the following aspects:

1. Correct implementation of heap-based priority queue. Your program must automatically generate the items with random benefits and random weight. Your program must allow user to specify the number of items to generate, and the maximum weight of the knapsack. The output must include the selected items, its portions, its benefits, its weights, its value, and total benefits. [7m]
2. Run your program for different cases (best, average & worst) [3m]
3. In your report, show your experiment results in graphs (running time vs input size). [3m]
4. Conclude your findings in the report. [3m]
5. Presentation. [4m]

Question 5

Implement the Boyer-Moore pattern matching algorithm. This implemented program should cover the following aspects:

1. Correct implementation. Your program must read the text T from a text file (*.txt), then allow the user to specify the pattern P to match. The output must state whether the pattern exist in the text and where the pattern is located in the text. For text with less than 60 characters, you program must show the pattern matching in every step. [7m]
2. Run your program for different cases (best, average & worst) [3m]
3. In your report, show your experiment results in graphs (running time vs input size). [3m]
4. Conclude your findings in the report. [3m]
5. Presentation. [4m]