Faculty of Engineering, Mathematics & Science

School of Computer Science and Statistics

Integrated Computer Science Programme Michaelmas Term 2018
BA (Mod) CSLL
BA (Mod) Business & Computing
BA MSISS
Year 2 Annual Examinations

CS2010: Algorithms and Data Structures

⟨\examday TBD⟩ ⟨\venue TBD⟩ ⟨\examtime TBD⟩

Dr. Ivana Dusparic, Dr. Vasileios Koutavas

Instructions to Candidates:

- This exam paper has TWO PARTS.
- Use SEPARATE answer books for each part.
- Answer TWO of the three questions in PART A.
- Answer TWO of the three questions in PART B.
- Each question is worth 25 marks.

Materials permitted for this examination:

- None.

Part B – sample questions

- 3 QUESTIONS total – addressing 3 big areas we have covered:

    1. Algorithm design approaches and general sorting algorithms
    2. String manipulation algorithms (substring search, tries, string sorts…)
    3. Graph algorithms

- Each question will consist of 3-4 smaller sub-questions on the same topic. Some sample sub-questions are provided below. One full question on, for example, sorting algorithms would, for example, consist of a subquestion asking you to list 4 sorting algorithms and discuss differences in their applicability/performance, a subquestion to provide trace of execution of a specific sort algorithm, and a subquestion to write code for a specific sort algorithm.

- Please note that the list below is of course non-exhaustive! They are just sample questions, while many other variations of the questions can be asked, as well as questions on other topics within the overall topic as discussed in the lecture notes.

<u>Sorting algorithm sample questions</u>

- Explain brute-force approach to algorithm design, and give a well known example of a brute-force algorithm. Explain why does the algorithm you chose belong to brute-force category, if needed using pseudo code or diagrams to illustrate your explanation.
    - in the above question, you could also replace brute-force with divide and conquer, decrease and conquer, transform and conquer, greedy, dynamic programming
    - the question could also ask to explain 2 or 3 of the approaches, highlighting differences between them
- List 4 algorithms you could use to sort a number of objects in ascending order. Compare the approaches with respect to their best case and worst case running time, additional space requirements, and provide an example of a situation when using each of the algorithms is preferred over the others.
- Explain in English how merge sort algorithm works?
    - in the above question, you could also replace merge sort with any other sort algorithm
- Write code for insertion sort.
    - Or any other sorting algorithm
    - Note: for long algorithms, such as merge sort for example, you might be asked to provide only part of the code, eg merge method, or recursive sort method

- Write pseudocode for insertion sort.
    - Or any other sorting algorithm
- Provide execution trace for insertion sort if given input array {2,4,6,3,2,1} (i.e., values of the array at each combination of i and j counters)
    - Or any other sorting algorithm
- Explain what is meant by a stable sort, provide an example of a stable and not stable sorting algorithm, and illustrate with an example why is the algorithm stable/not stable.

String algorithms questions

- Explain in English (or write pseudocode, or write code) how key-index counting algorithm works
- What is meant by radix sort, and give an example of a radix sort algorithm
- What are LSD and MSD, and what is the difference between them? When would you use each?
- How does performance of LSD compare to the performance of insert/merge/quick sort?
- Explain in English (or write pseudocode, or write code) how LSD algorithm works
- Explain in English (or write pseudocode, or write code) how MSD algorithm works
- Provide a trace of sorting the following array of strings using LSD/MSD. {“test, “blah”, “hack”, “mars”, “team”}

- List 4 algorithms you could use to search for a substring in a larger string. Compare the approaches with respect to their best case and worst case running time (in terms of character compares), additional space requirements, and provide an example of a situation when using each of the algorithms is preferred over the others.
- Explain what is a DFA and how is it used in KMP substring search algorithm?
- Build a DFA array and provide its array representation as well as a graphical representation of its state transitions that KMP algorithm would construct when searching for a substring “BANANA” in an alphabet consisting only of letters A, B, C, N, G
    - Using the above DFA, provide a trace of state transitions when searching for a substring “BANANA” in a string “ABCNGBANCG”
- You are given a String searchin = “teststringjoananomatch” and String searchfor = “ivana”. How many character comparisons in total will Boyer-Moore make searching for a searchfor string in a searchin string? List the trace of skip variable for this search?

- What are tries and when is it suitable to use them?

- Show a graphical representation of an R-way trie storing the following strings: cat, catapult, catering, bat, battery, ameba. (you are welcome to choose R, ie, the character set you are working with)
- Show a graphical representation of a TST storing the following strings: cat, catapult, catering, bat, battery, ameba.
- What is the difference between R-way trie and a TST (ternary search tree)? Illustrate the answer with a simple sample trie. Discuss the differences in their performance and space requirements.

Graph algorithms questions

- In-class exercises we did on graphs are a good example of a graph subquestion – fill in the table with path lengths, parent node, list order in which vertices are visited etc for DFS, BFS, Prim, Kruskal, Dijkstra, Bellman-Ford, Floyd-Warshall
- Explain in English (write code, pseudocode) how DFS/BFS works. What is the difference between them
- What is a minimum spanning tree? List 2 algorithms you can use to find them, explain in English how they work and what is the difference between them in terms of performance and space requirements.
- Explain in English how Prim's MST algorithm works. What is the difference between lazy and eager Prim implementation?
- Explain in English how Kruskal's MST algorithm works.
- Explain why Prim (Kruskal) is said to be a greedy algorithm.
- List 4 algorithms that can be used to find a shortest path between 2 vertices. Explain the differences between them in terms of best/worse case running time, space requirements, and their applicability depending on graph characteristics (eg directed cycles, negative edges)
- List 2 algorithms that can be used to detect if a graph contains negative cycles. Explain in English how is each used to do the detection (illustrate your example with code, pseudocode, table or a graph if required)
- Why is Dijkstra said to be a greedy algorithm? What are the characteristics of greedy algorithms?
- Why are Bellman-Ford and Floyd-Warshall examples of dynamic programming algorithm design? What are the characteristics of dynamic programming? Illustrate your answer with code/pseudocode/table/graph example as needed.

https://www.cs.usfca.edu/~galles/visualization/Algorithms.html is a great source of algorithm visualisations, which can also be used to generate sample problems and practice answering "trace" type of questions.