Ideas for presentations for the oral exam of DM582

Mads Anker Nielsen
madsn20@student.sdu.dk

May 22, 2024

This document contains some suggestions for topics that you can present at the oral exam of DM582. **Please note** that these are only suggestions and that you are free to choose any other appropriate topic you like. It is very possible that you get better ideas when reading through the course material as part of your preparation for the exam. The suggestions are merely intended to give you some ideas.

The evaluation is, of course, not only based on the choice of topic but also how well it is presented. Choosing something you find interesting and are comfortable with is likely to result in a better presentation.

For each question, we provide two suggestions. The first suggestion is intended to be a more challenging than the second. **We stress that you do not have to pick one of the suggestions provided here.**

1 Network Flows

Suggestion 1

Briefly define networks and (s,t)-flows. Explain why the capacity of an (s,t)-cut places an upper bound on the value of a feasible (s,t)-flow. Use this to prove correctness of the Edmonds-Karp algorithm for computing a maximum (s,t)-flow. Explain why the length of the shortest (s,t)-path is monotonically increasing throughout the execution of the Edmonds-Karp algorithm. Show that the flow is augmented by a path of length k at most $O(|V||E|)$ times and conclude from this that the runtime of the Edmonds-Karp is $O(|V||E|^2)$.

If you have more time (or as an alternative), you can also derive the Max-Flow Min-Cut theorem.

Suggestion 2

Briefly define networks and (s,t)-flows. Explain the role of the residual network in the Ford-Fulkerson method for computing a maximum (s,t)-flow. Explain why the capacity of an (s,t)-cut places an upper bound on the value of a feasible (s,t)-flow. Prove that, if there are no augmenting paths in the residual network N_f with respect to a flow f, then there is an (s,t)-cut with capacity equal to the value of f. Conclude from this that the Ford-Fulkerson method is correct.

If you have more time, prove the Max-Flow Min-Cut Theorem, possibly focusing on selected parts of the proof.

Additional suggestions from class

- Model a multi-source, multi-sink flow problem as a single-source, single-sink flow problem.
- Formulate the maximum bipartite matching problem as a network flow problem.

2 Randomized Algorithms

Suggestion 1

Briefly introduce the MAX-3-SAT problem. Explain how to randomly assign truth values to the variables such that 7/8 of the clauses are satisfied in expectation. Conclude from this that there exists an assignment that satisfies at least 7/8 of the clauses. Following the derivation from KT, derive a lower bound on the probability that a random assignment satisfies at least 7/8 of the clauses. Use this observation to define a Las Vegas algorithm which finds an assignment satisfying at least 7/8 of the clauses in expected polynomial time.

Suggestion 2

Briefly introduce the selection problem (finding the k-th smallest element in an array). Define and analyze the randomized algorithm selection algorithm, showing that it runs in expected linear time.

You may also have time to introduce the MAX-3-SAT problem and explain how to randomly assign truth values to the variables such that 7/8 of the clauses are satisfied in expectation. Deriving a lower bound on the probability that a random assignment satisfies at least 7/8 of the clauses is a bit more involved, and you may wish to skip this part.

Additional suggestions from class

- The randomized contraction algorithm for finding a minimum cut in a graph.

3 Amortized Analysis

Suggestion 1

Introduce the potential method for amortized analysis. Explain which conditions the potential function must satisfy and why an upper bound on the amortized cost of an operation implies an upper bound on the total cost of a sequence of operations. Using the potential function from the exercises, show that rebalancing a red-black tree has an amortized cost of $O(1)$. Do not spend time giving all details of the rebalancing operations. Mention that some rebalancing operations immediately cause the rebalancing procedure to stop and focus on showing that the potential decreases when we perform a rebalancing operation that does not cause the procedure to stop.

Suggestion 2

Introduce the potential method for amortized analysis. Explain which conditions the potential function must satisfy and why an upper bound on the amortized cost of an operation implies an upper bound on the total cost of a sequence of operations. Use a potential function to show that a queue can be implemented with two stacks such that the amortized cost of both `ENQUEUE` and `DEQUEUE` is $O(1)$.

Additional suggestions from class

- The amortized analysis of the dynamic table from CLRS using a potential function.

4 Universal and Perfect Hashing

Suggestion 1

You may wish to motivate and define a universal hash family $\mathcal{H}$. Show that the family of functions you define satisfies $\Pr[h(x) = h(y)] \leq 1/m$ (m is the size of the domain of the hash function) when $x \neq y$ and h is chosen uniformly at random from $\mathcal{H}$. Explain perfect hashing and analyze the two-level scheme for prefect hashing from CLRS (3rd edition) section 11.5.

Suggestion 2

Suggestion 1 can be simplified by omitting some details. For example, you may skip the construction of a universal hash family and instead just state that such a family exists. This should leave more time for the analysis of the two-level scheme for perfect hashing.

5 String Matching

Suggestion 1

Explain how to construct a deterministic finite automaton for the string matching problem and show that the automaton correctly finds all occurrences of a pattern. Explain the Knuth-Morris-Pratt and describe its advantages over the automaton method. The formal proof of correctness from CLRS is too formal to make a nice oral presentation. Instead, you may wish to invest more time in understanding the intuition and come up with simpler arguments better suited for an oral presentation.

Suggestion 2

Introduce the string matching problem and the naive algorithm for solving it (briefly). Present the Rabin-Karp algorithm for string matching and explain how it is better than the naive algorithm when we expect only a constant number of matches. You may also have time to introduce the DFA-based algorithm for string matching and explain how it works.

6 Online Algorithms

Suggestion 1

Briefly introduce the general online problem setting. Introduce the k-server problem on the line. Explain what it means for an algorithm to be c-competitive and define the competitive ratio. Introduce the algorithm DC and provide matching upper and lower bound on its competitive ratio of k.

Suggestion 2

Briefly introduce the general online problem setting. Introduce the k-server problem on the line. Explain what it means for an algorithm to be c-competitive. Show that the algorithm GREEDY is not competitive by showing that, for any $k > 0$ there exists a request sequence σ for which GREEDY incurs a cost of at least k while the optimal offline algorithm incurs only constant cost.

As an alternative or maybe in continuation of the above (if you have time), introduce the Treasure Hunt Problem. Show that any deterministic algorithm for the Treasure Hunt Problem has a competitive ratio of at least 2 and show how to achieve a competitive ratio of 3/2 using a randomized algorithm.

Additional suggestions from class

- Machine scheduling. Upper and lower bound for the algorithm LS.