Department of Mathematics and Computer Science November 17, 2016
University of Southern Denmark, Odense KSL

Study Group Suggestions for Week 47 on
Online Algorithms and Multiple Choice

topics in DM534 – Introduction to Computer Science

Kim Skak Larsen

Fall 2016

Discussion 1

This is about how to grade a multiple choice exam, i.e., how does one assign credit to answers to questions such as this one?

A `while`-loop is ...

- ☐ a bad cycle of personal problems
- ☐ a programming construct
- ☐ another name for a traffic circle
- ☐ a plane loop in an air show with a certain curvature

We assume that there is only one correct answer.

For this discussion, we recommend that you allow at least 10 minutes for the group members to think about the answers to the questions below and design your own grading policy before you then present your suggestions to each other and compare.

Think about how you would handle the fact that students may guess.

In normal written exams, we try to give partial credit when students know something but not everything about a given question. We would like to do that also in a multiple choice test. Thus, we allow students to mark more than one box as in the following:

A `while`-loop is . . .

- ☐ a bad cycle of personal problems
- ☒ a programming construct
- ☒ another name for a traffic circle
- ☐ a plane loop in an air show with a certain curvature

Discuss how one could fairly assign credit to such answers.

What if the marked boxes don't contain the correct answer?

Review your thoughts on how to handle random guessing in the light of partial answers.

Discussion 2

Discuss/repeat the characteristics of an online algorithm.

Discussion 3

Brainstorm to come up with online problems from our computer world as well as from real life.

In real life, pretty much nothing is really irrevocable, but there are many problems where we must treat requests without knowing the future, and default (or a company's business model) is that the decision is irrevocable. Think of the transportation sector and packing problems as sources for online problems in the real world.

Also try to think about variations of the problems we have considered in the lecture.

Discussion 4

The problems we considered in the lecture were all *minimization* problems, i.e., we wanted to minimize a cost (waiting time for jobs to finish or the cost of providing bins for packing). Can you think of a *maximization* problem? That is, a problem where we want to maximize profit (instead of minimizing cost). (Maybe you did so already in the previous discussion.)

Which ratios will one naturally get for a maximization problem and what are we aiming for?

Discussion 5

Discuss how one could implement the First-Fit algorithm, FF, for bin packing in the most efficient manner.

That is, if we have lots of bins in use, how do we efficiently find the first bin that has sufficient space for the current item?

To make it easier, we can assume that we know an upper bound on how many bins we will need during the processing.

Hint: Build a binary tree structure on top of the bins, i.e., make nodes that connect bins two and two, then recursively make nodes that connect nodes at lower levels two and two. Finally, maintain one piece of useful information in each node, which could help in the search.

Discussion 6

If we would try to design an algorithm working *better* than the First-Fit algorithm, FF, for bin packing which property should it probably have that deviates radically from the behavior of FF?

Try to suggest alternative algorithms.