Institut for Matematik og Datalogi
Syddansk Universitet

May 26, 2014
JFB

# DM508 – Algorithms and Complexity – 2014
# Lecture 9

## Lecture, May 13

We finished Fibonacci heaps and begin on string matching from chapter 32, covering up through the Rabin-Karp algorithm in section 32.2.

## Lecture, May 20

We will finish string matching from chapter 32.

## Lecture, May 21

We will not meet if we have finished string matching and discussed the exam.

## Problems to be discussed in U30a on May 26, 12–14

## No discussion section on May 28

Do problems:

1. 32.4-1, 32.4-3, 32.4-6, 32.4-7.
2. 32.4-4 and 32.4-5.

## Assignment due Wednesday, May 28, 8:15

Note that this is part of your exam project, so it must be approved in order for you to take the exam in June (note: there will be no redos allowed for this assignment), and you may not work with or get help from others not in your group (though you may talk with Christian Kudahl or myself). You may work in groups of two or three. You may write your solutions in English or Danish, but write very neatly if you do it by hand. Submit the assignment via Blackboard's "SDU Assignment" as one PDF file, with no Danish letters in the file name. Remember to keep a receipt. Turn in one assignment per group. Write all of your names on the first page (or cover page if you have one).

1. Consider the following game from the first assignment, called "Narrow Cave". In this game, you start at the entrance to a long, but narrow, cave. The cave is divided up into 2 meter blocks, with lines separating the blocks. The number of blocks is $\ell$, and the blocks are numbered from 1 to $\ell$, in order, with the block closest to the entrance being block 1.

   There is one large basket in each block, each one containing a large amount of one type of fruit (a different type for each basket). The player initially knows which type of fruit is found in each block.

   As long as the game continues (it can stop at any time, but the player cannot decide when it stops), whenever the player is at the entrance to the cave, a new round starts and the player is told which type of fruit to get. The player must go into the cave, find the basket with that type of fruit, and bring back exactly one piece of that fruit. (Assume that no basket is ever empty.)

   During one round, the player, is only allowed to take the type of fruit, which was asked for. He/she is only allowed to take one piece of it.

   On returning from getting a piece of fruit in basket $i$, the player can move that basket $i$ to any block closer to the cave entrance, by repeating switching basket $i$ with the basket currently in front of it. This is an easy operation which has no cost. No other switching of baskets is allowed.

   The cost of a round is the block number where the correct piece of fruit was found. The score for the game is the sum of the costs for each round. A low score is best.

   If the player breaks any rules, he/she get the total score of $\infty$, which is worse than any other score possible.

   Suppose the baskets are original ordered by frequency, so the fruits that are asked for most frequently are at the front of the cave.

   Consider the algorithms NoMoves which never moves the baskets and MoveToBlockOne which always move the basket with the fruit asked for all the way to block 1 (to the entrance).

   (a) Suppose that originally block 1 has apples, block 2 has oranges, and block 3 has bananas. Suppose that the order of the fruits the player gets is

   $$\langle \text{apple, orange, banana, apple, orange, apple} \rangle.$$

   What is the cost of each of the two algorithms on this ordering? Explain you answer.

   (b) Find an ordering which would have apples most frequently, oranges next and bananas leas frequently in which MoveToBlockOne has lower cost than NoMoves.

(c) Use amortized analysis to show that there is no order of asking for fruits which will cause MoveToBlockOne to have cost more than twice the cost of NoMoves for the same ordering of fruits.

Hint: Consider using the potential function which, for a particular order of the baskets for MoveToBlockOne, is the number of pairs of baskets which are relatively in a different order from how they were in the original order (which is the NoMoves always has).

For example, with the ordering

$$\langle \text{apple, orange, banana, apple, orange, apple} \rangle,$$

initially the potential is zero, since block 1 has apple in both, block 2 has orange in both, and block 3 has bananas in both. Asking for apple does not change any order or the potential. Asking for orange changes the order of apple and orange, but apple and orange still both come before banana, so the potential is 1. Asking for banana changes the order of both apple and banana with respect to each other and the order of orange and banana with respect to each other. This increases the potential by 2, giving a total of 3. The next request for apple puts both the pair (apple, orange) and the pair (apple, banana) back in the same order as originally, so it decreases the potential by 2, giving a total of 1 (orange and banana are still in the wrong order).

Hint 2: The starting order is not important.

Hint 3: Keep track both of increases and decreases in potential in each round.

Hint 4: Suppose fruit $X$ is asked for and MoveToBlockOne has fruit $X$ in block $k$. Suppose that, of the $k-1$ baskets which MoveToBlockOne has in front of block $k$, there are $k_1$ which NoMoves also has in front of where it has fruit $X$, and it $k_2$ which NoMoves has behind where it has fruit $X$. First, express the amortized cost, $\hat{c_i} = c_i + \Delta\Phi$, of finding fruit $X$ in terms of $k_1$ and $k_2$. Then argue that this is at most twice the cost for NoMoves.

2. Give a sequence of operations for a Fibonacci heap which would give as the final result a balanced binary tree of depth 2 (having 7 nodes). Start with an empty heap.

3. What is the prefix function computed by the KMP algorithm for the string P = abbdabbdabbcabbd? Explain a few of the more interesting steps in computing the prefix function.