Albert-Ludwigs-Universität, Inst. für Informatik
Prof. Dr. Fabian Kuhn
H. Ghodselahi, Y. Maus November 19, 2014

Algorithm Theory, Winter Term 2014/15
Problem Set 5

hand in (hard copied) by Thursday, 10:00, November 27, 2014, either before the lecture or in the box corresponding to your group in building no. 51.

Exercise 1: Edit Distance, (1+3.5+1 points)

The algorithm to compute edit distances works as long as the triangle inequality for the cost function is satisfied. Consider the following cost model for replacing symbols: Let $\mathbb{P}$ map each symbol $\{\epsilon, A, \ldots, Z\}$ to its position in the alphabet, e.g., $\mathbb{P}(\epsilon) = 0$, $\mathbb{P}(C) = 3$ and $\mathbb{P}(Z) = 26$.
For two symbols α, β with $\mathbb{P}(\alpha) \geq \mathbb{P}(\beta)$ we define their non negative *distance*

$$c(\beta, \alpha) = c(\alpha, \beta) = min\{(\mathbb{P}(\alpha) - \mathbb{P}(\beta)), (\mathbb{P}(\beta) - \mathbb{P}(\alpha)) + 27\}.$$

E.g. $c(\epsilon, V)$ equals 5.

a) Give an intuition for the cost function c and argue why the triangle inequality is satisfied (no formal proof needed).

b) Perform the edit distance algorithm from the lecture with the modified distance function to compute the edit distance of the strings `TIME` and `TRIP`.

For easier grading please write `TIME` at the top of your DP table.

c) Write down the sequence of edit operations corresponding to your solution in *b*) (including the cost of each operation), which transform the string `TIME` to `TRIP`.

Exercise 2: Binomial Heap, (1.5+3+2 points)

You are given a Binomial Heap where (as discussed in the lecture) each binomial tree satisfies the min-heap property.

a) You need to extend the data structure by providing an operation

$$increaseKey(x, newValue)$$

to *increase* the key of an element x to the value $newValue$.
The natural way to solve this task is to increase the key of the element and to afterwards 'swap' the element down the tree until the heap property is satisfied.

Explain this natural algorithm in more detail and analyze its running time.

Remark: Binomial heaps do not provide a (fast) method to search for a certain key. Thus the x in decreaseKey(x,newValue) and in increaseKey(x,newValue) is already a pointer to the corresponding element.

b) How can one delete an element x from a binomial heap? What is the running time?

c) Describe a different solution for the task of increasing a key in a binomial heap that runs in time $\mathcal{O}(\log n)$.

Remark: This was a previous exam question.