Karlsruher Institut für Technologie
Institut für Theoretische Informatik

Dr. Simon Gog

1. Project in Text Indexing (WS 2016/17)

http://algo2.iti.kit.edu/3175.php
gog@kit.edu

Aufgabe 1 (*Suffix tries and suffix trees*)

Given a text T=0000100110101111$ over an alphabet of size 3. Construct the suffix trie and suffix tree of T and compare the number of nodes of the two structures.
You get a bonus point for implementing the computation of the two numbers and processing the following text:
0000000100 0001100001 0100001110 0010010001 0110001101 0001111001 0011001010 1001011100 1101100111 0100111110 1010110101 1110110111 01111111$
(The whitespace between the blocks should be ignored.)

Aufgabe 2 (*Better SA sampling*)

The sample suffix array SA' as presented in the lecture takes at most $\frac{n}{s} \cdot \log n + 2n$ bits of space, where s is the sampling parameter and n the size of the text. Show how the space can be reduced to $\frac{n}{s} \cdot \log \frac{n}{s} + 2n$ bits while the access operation still remains in $O(s \cdot t_{LF})$.

Aufgabe 3 (*ISA sampling*)

The suffix array (SA) is a permutation of the numbers $0, \ldots, n-1$. The inverse suffix array (ISA) is the inverse permutation of SA, i.e. $SA[ISA[i]] = i$. Devise a sampling scheme for ISA which just stores n/s ISA values and enables access to every ISA[i] value in $O(s \cdot t_{LF})$ worst case time. The scheme should use less space than the SA sampling scheme.

Aufgabe 4 (*Cyclic rotation*)

Write a program which uses an index structure to check if two strings of the same length n are cyclic rotations of each other. E.g. `ababba` and `abbaab` are cyclic rotations of each other but not `ababba` and `bbabaa`. The index structure should be build for one of the strings and the algorithm should perform at most n matching steps.