

Tutorial 4 for COMP 526 – Applied Algorithmics, Winter 2020

Problem 1 (Fibonacci language and failure function)

The sequence of Fibonacci words $(w_i)_{i \in \mathbb{N}_0}$ is defined recursively:

$$\begin{aligned}w_0 &= \mathbf{a} \\w_1 &= \mathbf{b} \\w_n &= w_{n-1} \cdot w_{n-2} \quad (n \geq 2)\end{aligned}$$

Unfolding the recursion yields $w_2 = \mathbf{ba}$, $w_3 = \mathbf{bab}$, $w_4 = \mathbf{babba}$, and so on.

(Note that the lengths $|w_0|, |w_1|, |w_2|, \dots$ are *Fibonacci numbers* \square , hence the name. More precisely, we have $|w_n| = F_{n+1}$, with the Fibonacci numbers defined as $F_0 = 0$, $F_1 = 1$, and $F_n = F_{n-1} + F_{n-2}$, for $n \geq 2$.)

Construct the failure function F and draw the KMP automaton with failure links for w_6 .

Problem 2 (How KMP uses itself)

Recall the example $T = \mathbf{abababaabab}$ and $P = \mathbf{ababaca}$ used in the lecture to illustrate the KMP failure-link automaton.

Now consider the string $S = S[0..m+n] = P\$T$ over the extended alphabet $\Sigma' = \Sigma \cup \{\$\}$ and construct the failure-links array $fail[0..n+m]$.

Compare the result with the sequence of states from simulating the failure-link automaton for P on T ; what do you observe?

Bonus: Can you compute the values $fail[0..n+m]$ using only $\Theta(P)$ extra space? Here, it is enough to have the values available at some time during the computation; we (obviously) cannot store all of them explicitly in the allowed space.