University of Freiburg
Dept. of Computer Science
Prof. Dr. F. Kuhn

Algorithms and Datastructures
Winter Term 2021/2022
Sample Solution Exercise Sheet 12

Exercise 1: Knuth-Morris-Pratt Algorithm

Consider the pattern $P = BBABAB$ and the text $T = ABBABBABABBABABBA$.

(a) Compute the array S of the Knuth-Morris-Pratt algorithm.

(b) Use the Knuth-Morris-Pratt algorithm to find all appearances of P in T. Document the steps analogously to the lecture.

Sample Solution

(a) $S = [-1, 0, 1, 0, 1, 0, 1]$

(b)

A	B	B	A	B	B	A	B	A	B	B	A	B	A	B	B	A
B	B	A	B	A	B											
	B	B	A	B	A	B										
				B	B	A	B	A	B							✓
									B	B	A	B	A	B		✓
														B	B	A

Exercise 2: Rabin-Karp Algorithm

Let T be a given text of length n and let $P_1, \ldots, P_k$ be k patterns, each of length exactly m. The goal is to know if there is at least one pattern in the text, that is, we want to answer *True* if there exists at least one index $i \in \{1, \ldots, k\}$ such that $P_i \in T$, and answer *False* if for any $i \in \{1, \ldots, k\}$, $P_i \notin T$. It is easy to solve this problem in $O(k(n+m))$ by running the Rabin-Karp algorithm once for each pattern. Give an algorithm (based on Rabin-Karp) that requires only $O(n + km)$.

Sample Solution

On average, Rabin-Karp takes $O(n+m)$ for searching if a pattern P appears in a text T, hence we can trivially run Rabin-Karp k times, once for each pattern P_i, and in this way we would solve the exercise in $k \cdot O(n+m) = O(kn+km)$. In any case, we need to spend $O(km)$ in order to compute the k hash values of the patterns, where each pattern is of length m. So, how to avoid spending $O(kn)$, and instead spend only $O(n)$?
Let's see where we spend this $O(kn)$ time. Using Rabin-Karp, we go through all $O(n)$ positions of the text, and for each window, we check if the hash value of that portion of the text matches the hash value of the pattern, and since here we have k patterns, for each window, we perform $O(k)$ checks. What can we do such that, for each text window, on average, we spend $O(1)$ to check if the hash value of the text window matches the hash value of one of the patters? We can put the hash values of the patterns into a hash table, and now we spend $O(1)$ for performing a search, spending in total $O(n + km)$.