Lecture

At the lecture we will talk about string matching algorithms: Rabin-Karp fingerprinting and the Knuth-Morris-Pratt algorithm (KMP). You should read Jeff Ericksons notes (see webpage).

Exercises

1 KMP Solve

1.1 $[w]$ Compute the prefix function π for the pattern $P = abcaba$ and draw the corresponding automaton with failure links. Run the matching algorithm on the text string $T = aaabcababcabbaabcabaab$.

1.2 $[w]$ Compute the prefix function π for the pattern $ababbabbabbababbabb$ when the alphabet is $\Sigma = \{a, b\}$ and draw the corresponding automaton with failure links.

1.3 Explain how to determine the occurrences of pattern P in the text T by examining the π function for the string $P\$T$, where \$ is a new character not in the alphabet.

2 Rabin-Karp$[w]$ Run the Karp-Rabin fingerprinting algorithm with the following fingerprint function:

$$F(P) = \sum_{i=1}^{m} 2^{m-i} P[i] \mod 5 \qquad F(T_s) = \sum_{i=1}^{m} 2^{m-i} T[s+i-1] \mod 5$$

on the following example: $T = 100101110110001$ and $P = 1011$.

3 String matching with gaps In *string matching with gaps* the pattern P can contain a *gap character* $\star$ that can match *any* string (of arbitrary length even length zero). An example of such a string is $P = ab \star ac \star a$, which occurs in the text $T = bababacbcca$ in two ways:

T:	b	ab	ab	ac	bcc	a
P:		ab	$\star$	ac	$\star$	a

or

T:	bab	ab		ac	bcc	a
P:		ab	$\star$	ac	$\star$	a

There are no gap characters in the text—only in the pattern.

Give an algorithm to find an occurrence of a pattern P containing gap characters in a text T in time $O(n+m)$. That is, preprocessing time + matching time should be $O(n+m)$).

4 Christmas songs (exam 2015) You are putting together a set of Christmas songs that will be handed out at the Christmas party. The Dean has declared that every song must contain the sentence "Merry␣Christmas␣Dear␣Dean", where "␣" denotes a blank space. E.g. the song:

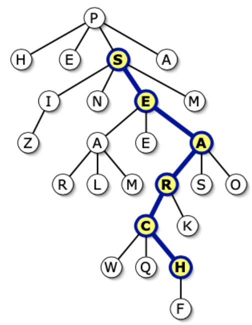
```
We␣wish␣you␣a␣Merry␣Christmas␣
We␣wish␣you␣a␣Merry␣Christmas␣
We␣wish␣you␣a␣Merry␣Christmas␣
Dear␣Dean␣
Dear␣Dean
```

contains one occurrence of of the sentence "Merry␣Christmas␣Dear␣Dean" (line breaks are disregarded).

Formally, you are given a set S of songs $S_1, \ldots, S_k$ and a sentence P. Song S_i contains n_i characters and P contains m characters. Let $n = \sum_{i=1}^{k} n_i$ denote the total number of characters in the songs. All the strings are over an alphabet of size $O(1)$. Describe an algorithm that returns all the songs that contain P. Analyze the asymptotic running time of your algorithm. Remember to argue that your algorithm is correct.

5 [†] **Implement KMP** Implement the KMP algorithm on CodeJudge.

6 Pattern matching on trees[1] Suppose we want to search for a string inside a labeled rooted tree. Our input consists of a pattern string $P[1..m]$ and a rooted text tree T with n nodes, each labeled with a single character. Nodes in T can have any number of children. Our goal is to either return a downward path in T whose labels match the string P, or report that there is no such path.



The string SEARCH appears on a downward path in the tree.

6.1 Describe and analyze a variant of KarpRabin that solves this problem in $O(m+n)$ expected time.

6.2 Describe and analyze a variant of KnuthMorrisPratt that solves this problem in $O(m+n)$ time.

Hint: If you use the optimized failure pointers described in section 7.7 in the notes, then the longest failure chain has length at most $O(\log m)$.

7 Finite String Matching Automaton Consider the folowing automaton: Instead of having failure edges as in the KMP automaton each state/node has $|\Sigma|$ edges out of it. The automaton should still have the property that if you are in state i after having read j characters from T then $P[1\ldots i]$ is the longest prefix of P that matches a suffix of $T[1\ldots j]$ (as is the case in the KMP automaton). Formally, let $Q=\{0,1,\ldots,m\}$ be the set of states in the automata. We have a transition function $\delta : Q\times\Sigma$, that for any $q\in Q$ and $a\in\Sigma$ satisfies that

$$\delta(q,a)=\max\{k : P[1\ldots k] \text{ is a proper suffix of the string } P[1\ldots q]\circ a\}\ .$$

7.1 Construct both the string-matching automaton for the pattern $P = abcaba$ and run the matching algorithm on the text string $T = aaabcababcabbaabcabaab$.

7.2 What is the running time of matching a text T given the finite string matching automaton?

7.3 Argue that it takes at least $\Omega(m|\Sigma|)$ time to construct the finite string matching automaton

7.4 [$*$] Give an efficient algorithm for computing the transition function δ for the string-matching automaton corresponding to a given pattern P. Your algorithm should run in time $O(m|\Sigma|)$. (Hint: Prove that $\delta(q,a) = \delta(\pi[q],a)$ if $q=m$ or $P[q+1]\neq a$.)

[1] Modified exercise from Jeff Ericksons notes