String Processing Algorithms 2015 - Week 2 Exercises

Rodion Efremov

October 30, 2015

Exercise 1

Outline algorithms that find the most frequent symbol in a give string.

(a) for ordered alphabet, and

(b) for integer alphabet.

The algorithms should be as fast as possible. What are their (worst case) time complexities? Consider also the case where $\sigma \gg n$.

Solution

Algorithm 1: MostFrequentSymbol(S)

```
let f be an empty map f : Σ → ℕ
μ = nil
L_μ = 0
for i = 1 to |S| do
    if S[i] is not mapped in f then
        f(S[i]) = 1
        if L_μ = 0 then
            L_μ = 1
            μ = S[i]
    else
        f(S[i]) = f(S[i]) + 1
        if L_μ < f(S[i]) then
            L_μ = f(S[i])
            μ = S[i]
return μ
```

Exercise 2

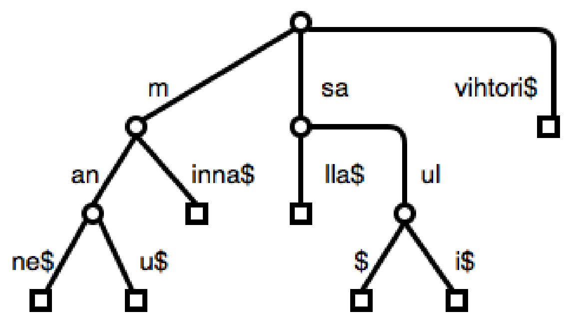
Let $\mathcal{R} = \{\texttt{manne}, \texttt{manu}, \texttt{minna}, \texttt{salla}, \texttt{saul}, \texttt{sauli}, \texttt{vihtori}\}$.

(a) Give the compact trie of $\mathcal{R}$.

(b) Give the balanced compact ternary trie of $\mathcal{R}$.

Solution

(a)



(b)

See the drawing.

Exercise 3

Exercise 4

Prove

(a) Lemma 1.14: For $i \in [2..n]$, $LCP_{\mathcal{R}}[i] = lcp(S_i, \{S_1, \ldots, S_{i-1}\})$.

(b) Lemma 1.15: $\Sigma LCP(\mathcal{R}) \leq \Sigma lcp(\mathcal{R}) \leq 2 \cdot \Sigma LCP(\mathcal{R})$.

Solution

Both by induction.

(a)

We need an auxiliary lemma first:

Lemma 1 (Distance lemma)**.** *If $S_1 < S_2 < S_3$, $lcp(S_1, S_3) \leq lcp(S_2, S_3)$.*

Proof. The proof is by contradiction: Let $\text{prefix}(A, B)$ be the longest common prefix of A and B. Assume that $l_1 > l_2$, where $l_1 = |\text{prefix}(S_1, S_3)|$ and $l_2 = |\text{prefix}(S_2, S_3)|$. Now $S_1 = a_1 a_2 \ldots a_{l_2} \ldots a_{l_1} b_1 b_2 \ldots$ and $S_2 = a_1 a_2 \ldots a_{l_2} b'_1 b'_2 \ldots$. Because $S_3 = a_1 a_2 \ldots a_{l_2} \ldots a_{l_3} c_1 c_2 \ldots$, we must have that $S_2 < S_1 < S_3$, which is a contradiction. $\square$

Example

$$\begin{aligned} S_1 &: aaaab \\ S_2 &: aaaba \\ S_3 &: aaabc \end{aligned}$$

Base step: Assume $i = 2$. Now $LCP_{\mathcal{R}}[2] = lcp(S_2, S_1) = lcp(S_2, \{S_1\})$.
Induction step: Assume that $LCP_{\mathcal{R}}[i] = lcp(S_i, \{S_1, \ldots, S_{i-1}\})$. Now

$$\begin{aligned} lcp(S_{i+1}, \{S_1, \ldots, S_i\}) &= \max\{lcp(S_{i+1}, S_i), lcp(S_{i+1}, \{S_1, \ldots, S_{i-1}\})\} \\ &\overset{DI}{=} lcp(S_{i+1}, S_i), \end{aligned}$$

which concludes the proof.