Exercises 3 (12 November)

1. Show how to construct the compact trie \(\text{trie}(\mathcal{R})\) in \(O(|\mathcal{R}|)\) time (rather than \(O(||\mathcal{R}||)\) time) given the string set \(\mathcal{R}\) in lexicographical order and the lcp array \(LCP_{\mathcal{R}}\).

2. Use the lcp comparison technique to modify the standard insertion sort algorithm so that it sorts strings in \(O(\Sigma LCP(\mathcal{R}) + n^2)\) time.

3. Give an example showing that the worst case time complexity of string binary search without precomputed lcp information is \(\Omega(m \log n)\).

4. Let \(S[0..n]\) be a string over an integer alphabet. Show how to build a data structure in \(O(n)\) time and space so that afterwards the Karp–Rabin hash function \(H(S[i..j])\) for the factor \(S[i..j]\) can be computed in constant time for any \(0 \leq i \leq j \leq n\).

5. The Knuth–Morris–Pratt algorithm differs from the Morris–Pratt algorithm only in the failure function, which can be defined as

\[
\text{fail}_{\text{KMP}}[i] = k, \text{ where } k \text{ is the length of the longest proper border of } P[0..i] \text{ such that } P[k] \neq P[i], \text{ or } -1 \text{ if there is no such border.}
\]

(a) Compute both failure functions for the pattern \text{ananassana}.

(b) Give an example of a text, where some text character is compared three times by the MP algorithm but only once by the KMP algorithm when searching for \text{ananassana}.

6. Modify Algorithm 2.6 on the lecture notes to compute \(\text{fail}_{\text{KMP}}\) instead of \(\text{fail}_{\text{MP}}\).