Assignment B – On Paper

Eirik Isene

February 16, 2016

1 From the book

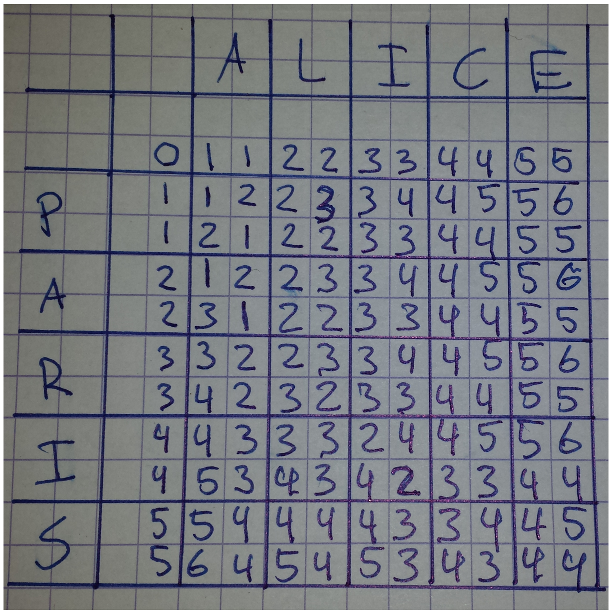
1.1 Excercise 3.3, Page 56

If you wanted to search for `s*ng` in a permuterm wildcard index, what key(s) would one do the lookup on?

Before rotating the term `s*ng` wer mark the end to make it `s*ng$`, then we rotate it so that we can treat it as a trailing wildcard query, the resulting key to use in a search is then: `ng$s*`

1.2 Excercise 3.8, Page 62

Compute the edit distance between `paris` and `alice`. Write down the 5 × 5 array of distances between all prefixes as computed by the algorithm in Figure 3.5



		A	L	I	C	E
	0	1 1	2 2	3 3	4 4	5 5
P	1 1	1 2 2 1	2 3 2 2	3 4 3 3	4 5 4 4	5 6 5 5
A	2 2	1 2 3 1	2 3 2 2	3 4 3 3	4 5 4 4	5 6 5 5
R	3 3	3 2 4 2	2 3 3 2	3 4 3 3	4 5 4 4	5 6 5 5
I	4 4	4 3 5 3	3 3 4 3	2 4 4 2	4 5 3 3	5 6 4 4
S	5 5	5 4 6 4	4 4 5 4	4 3 5 3	3 4 4 3	4 5 4 4

2 String matching algorithms

2.1 Kurts large dicitionary

Kurt has a large dictionary with millions of elements. As part of a document processing system, he wants to develop a module which can efficiently detect whether a word in a document is also present in the dictionary.

2.1.1 What data structure should Kurt use to represent his dictionary?

I would recommend Kurt to use a Trie to represent the dictionary. This makes finding words in the dictionary a simple and efficient operation. It will also save space since a lot of the words in the dictionary wil have common prefixes that share nodes in the trie structure. It would also be wise to tightly pack the trie, this way not only the prefixes are shared, but also the suffixes. For a large dictionary this will save a lot of space!

2.1.2 What algorithm should be used to search the data structure?

To search the data structure a Trie walk algorithm should be used.

2.2 Kurts spellchecking

Kurt also has a big dictionary containing the surface forms of the most common words in Norwegian. As part of a spellchecking application, Kurt wants to be able to query the dictionary with a word w and get back the set of all words which have an edit distanec of at most k from w. More formally, the resulting set is thus defined as

$$\{w' : editdistance(w', w) <= k\}$$

You can assume that the maximum distance k is small.

2.2.1 How should Kurt represent his dictionary and perform the search?

Seeing as words following the same path in a Trie have the same prefix, a Trie should still be used for this. When finding words that have edit distance within the limitations, one can calculate columns untill the edit distance exceeds the limitation, and cut away all branches from there and out since the editdistance won't decrease further down.