Tutorial 11

Polynomial multiplication with FFT, problems on compression

CS 240E W23
University of Waterloo
Monday, April 3

Polynomial multiplication with Fast Fourier Transform (FFT)

In tutorial, we will discuss a way to multiply, in $O(n \log n)$ time, two polynomials $A(x), B(x)$. $A(x) = a_0 + a_1 x + \cdots + a_{n-1}x^{n-1}$ with n terms is $[a_0, \ldots, a_{n-1}]$.

1. Given two vectors $a = [a_1, \ldots, a_n]$ and $b = [b_1, \ldots, b_n]$, compute the dot product of a with every cyclic shift of b.

   For instance, when $n = 3$, we would like to compute:

   $$[a_1, a_2, a_3] \cdot [b_1, b_2, b_3],$$
   $$[a_1, a_2, a_3] \cdot [b_2, b_3, b_1], \text{ and}$$
   $$[a_1, a_2, a_3] \cdot [b_3, b_1, b_2].$$

2. We are given two *cyclic strips* $a = [a_1, \ldots, a_n]$ and $b = [b_1, \ldots, b_n]$ where every entry of each of a, b is either zero or one.

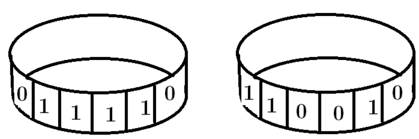
   

   Figure 1: example cyclic strips.

   Give an algorithm to find the number of ways to stack them on top of each other so that no two *one* (1) entries of both a and b are adjacent.

3. Given a string s over the alphabet $\{A, B\}$, we say that a pair of indices $0 \leq i < j \leq n-1$ forms a *k-inversion* if $s[i] = B$, $s[j] = A$, and $j - i = k$. For example, the string $BABA$ has two one-inversions, one three-inversion, and no two inversions.

   Give an algorithm to find all k-inversions in a given string s, for all $1 \leq k \leq n-1$.

4. Given two images, one smaller in both dimensions that the other, where each pixel is either of colour 0 or 1, we can overlap them and count the number of matches: i.e. the number of pixels of the same colour.

   For example, if the small image is,

```
1 0
0 1
```

and the large image is

```
1 0 0
0 1 0
0 0 1
```

then there are two placements that achieve four matches (align at top-left corner, or align at bottom-right corner), which is optimal.

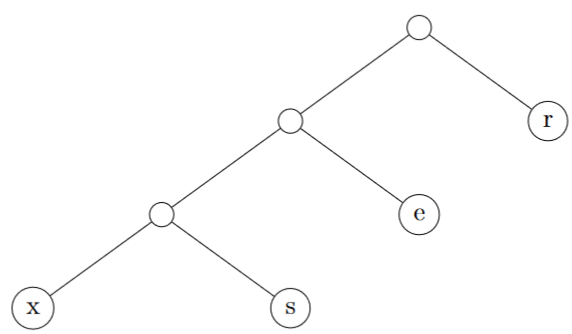
Give the placements of the smaller image on top of the larger image that are tied for having the most matches.

Compression

5. **Huffman encoding examples.**

   (a) Build a Huffman tree for $S =$ "pusheen".

   (b) Below is an encoding tree for the string "xerxes." Argue that it is not a Huffman tree.



6. **Huffman encoding.** Let $c_1, \ldots, c_k$ be the characters of a text, sorted by (not necessarily strictly) increasing frequencies. Let $s(c_1), \ldots, s(c_k)$ be the prefix-free encoding of these characters obtained with the Huffman encoding algorithm.

   (a) Prof. I.N. Correct thinks that $s(c_1)$ *must* have the shortest codeword length, i.e., $|s(c_1)| \leq |s(c_i)|$ for all $i = 2, \ldots, k$. Show that the professor is incorrect.

   (b) Show that the professor is correct if the frequency of c_1 is *strictly* larger than all other frequencies.

7. **LZW encoding examples.** For the following LZW problems, consider the initial dictionary to be the ASCII table.

   (a) Encode the following string using LZW: `BANANA_BANDANA`

   (b) Decode the following encoded string using LZW:

$$71 - 73 - 86 - 69 - 95 - 77 - 131 - 82 - 69 - 128 - 137 - 65 - 83$$

8. **LZW encoding.** Let S be a string of length n. Argue that the LZW encoding of S must use $\Omega(\sqrt{n})$ integers.