Prediction of Transcription Factor Binding Sites with Suffix Arrays

Myung Eun Lim melim@etri.re.kr
Jeong Seop Sim simjs@etri.re.kr
Myung Geun Chung cmk63697@etri.re.kr
Sun Hee Park shp@etri.re.kr

Electronics and Telecommunications Research Institute, Gajeong-dong, Yuseong-gu, Daejeon, Korea

Keywords: transcription factor, transcription factor binding site, suffix array

1 Introduction

Suffix trees and suffix arrays are very important index data structures in diverse applications in string processing and computational biology. Despite simplicity of suffix arrays, suffix trees have been the most fundamental index data structures in the literature because suffix arrays were inferior to suffix trees in the two aspects, construction time and search time. But recently, there have been vigorous works on suffix arrays, and suffix arrays are proved to be as powerful as suffix trees [1, 2, 3, 4].

The availability of the whole genome sequences of human due to the Human Genome Project makes it possible to study gene function much more effectively. We can find or predict the functions and positions of genes by analyzing the transcriptional regulation. In general, there are three issues in the field of transcriptional regulation, i) transcription factors (TF for short), ii) transcription factor binding sites, and iii) regulatory proteins.

In this poster, we focus on the second issue and suggest an algorithm of predicting TF binding sites from a given set of sequences of upstream regions of genes (transcriptional regulation regions). For this, we need two assumptions. First, we assume that there tends to be a common TF that binds with each of the upstream regions of the functionally related genes. Our second assumption is that a TF binds with similar DNA sequences in different genes or different organs.

2 Algorithm and Results

2.1 Algorithm

Given a set S of DNA sequences $s_1, s_2, \ldots, s_n$ our algorithm finds a common sequence above given thresholds. In general, the input sequences are set of sequences that seem to be functionally related. We use suffix arrays to find common DNA sequences in the given set of transcriptional regulation regions. First, we make a long sequence T by concatenating all the sequences in S. That is, $T = s_1\#_1 s_2\#_2 \ldots s_n\#_n$. Each $\#_i$ is a special character to delimit each sequence. Now, we make a suffix array SA_T of T and make another array called LCP array. An LCP array is an array of the lengths of common prefixes of two adjacent suffixes in SA_T. That is, $LCP[i]$ stores the length of common prefix of $SA_T[i]$ and $SA_T[i+1]$. We use Kärkkäinen and Sanders's [1] algorithm which constructs a suffix array of a given sequence in linear time. We are given two thresholds. One is LEN, the shortest length of the candidates of TF binding sites. The other is RTO, the frequency ratio of the candidate of TF binding sites in the given sequences. For example, if 10 sequences are given and LEN and RTO

are 4 and 0.8, respectively, our algorithm finds all the sequences that are longer than 4 and appear in at least 8 sequences in the given set of sequences. See Fig 1.

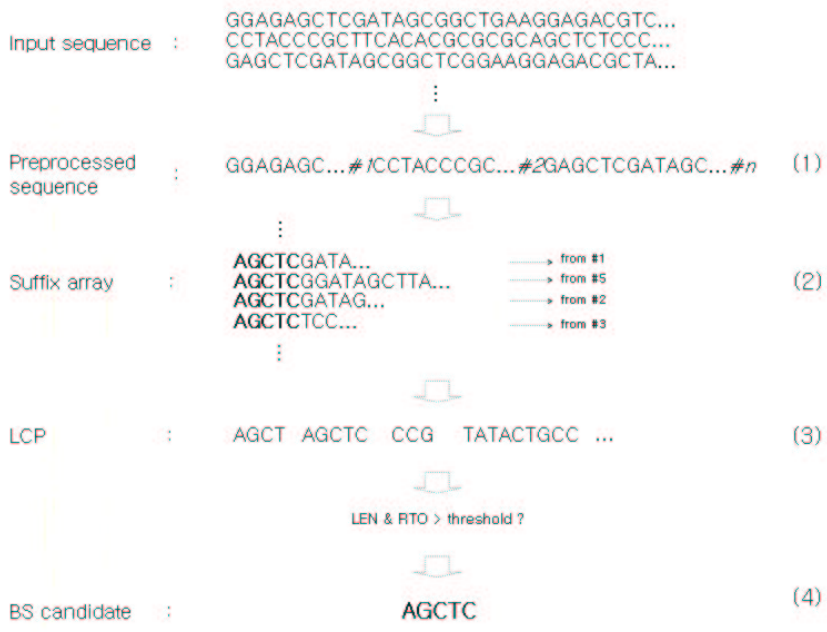

Figure 1: **Flow of algorithm**. (1) Given input sequences, our algorithm concatenates all the input sequences to make them into one sequence. (2) Build the suffix array of the concatenated sequence, and (3) make the LCP array. (4) Find sequences above given thresholds. In this example, LEN is 5 and RTO is 0.8. AGCTC is selected as a candidate.

2.2 Experimental Results

We obtain upstream regions and TF binding sites from EMBL release 75 and TRANSFAC release 3.2 to test our algorithm. We evaluated the performance of our algorithm by the measure of PPV (positive probability value) and obtained 35.5% at best case when LEN is 4 (bp) and RTO is 0.85.

3 Discussion

In this poster, we proposed an algorithm that predicts TF binding sites from transcription regulation regions by extracting common sequences using suffix array. Our algorithm is based on exact string matching. We think the performance of our algorithm can be improved using approximate string matching or multiple pattern matching.

References

[1] Kärkkäinen, J. and Sanders, P., Simple linear work suffix array construction, *International Colloquium on Automata, Languages and Programming*, 2719:943–955, 2003.

[2] Kim, D.K., Sim, J.S., Park, H., and Park, K., Linear-time construction of suffix arrays, *Combinatorial Pattern Matching*, 2676:186–199, 2003.

[3] Ko, P. and Aluru, S., Space efficient linear time construction of suffix arrays, *Combinatorial Pattern Matching*, 2676:200–210, 2003.

[4] Sim, J.S., Kim, D.K., Park, H., and Park, K., Linear-time search in suffix arrays, *Australasian Workshop on Combinatorial Algorithms*, 139–146, 2003.