Bundled Suffix Trees

Bortolussi L.(1) Fabris F.(2) Policriti A.(1)

(1) Department of Mathematics and Computer Science, University of Udine, Udine. (2) Department of Mathematics and Computer Science, University of Trieste, Trieste.

Motivation

A Suffix Tree (ST) is a --now classical-- data structure, computable in linear time, which represents the most algorithmically appropriate way to store a string, in order to face problems like the Exact String Matching Problem (ESM) or the Longest Common Exact Substring Problem (LCES). Even if very efficient in solving these problems, the ST data structure suffers from an important drawback, when dealing with an Approximate String Matching Problem (ASM) or with the harder Longest Common Approximate Substring Problem (LCAS), as only exact matching can be used in visiting a ST. In the approximate cases, a suitable notion of distance (most frequently Hamming or Levenshtein distances) must come into play. However, in the literature, there is no universally accepted data structure capable to deal with approximate searches just by performing algorithmic manipulations similar to ST's. This makes necessary, when using ST's in an approximate context, taking into account errors by using unnatural and complicate strategies, inevitably leading to cumbersome algorithms.

Methods

In this work we elaborate the notion of suffix tree and we propose a new data structure, the Bundled Suffix Tree (BuST), containing information about the distance between strings as a structural property. BuSTs allow us solve some approximated string matching problem (in particular LCAS, taken as representative of a large class of approximated problems) in the same manner in which we can solve the exact versions with a ST. The matching criterion we use can be very general; in fact we only require to be given a --not necessarily transitive-- relation among letters of the alphabet. For instance, the notion of Hamming distance induces a very natural non-transitive relation on an alphabet, which is constituted by tuples over a sub-alphabet. A BuST for a text S is built starting from the ST through the addition of a set of labelled nodes, which encodes how substrings of S are related. This data structure is no more linear in the length of the text itself, but, on average, its size grows with an exponent just slightly greater than one.

Results

We introduce an algorithm for the construction of a BuST, whose average time complexity is linear in the size of the structure. We also show how BuSTs lead to a natural extension of the linear solution for the Longest Common Substring problem to the approximate case. Furthermore, we briefly discuss the potential of the data structure for solving search problems in which the introduction of macro-characters seems significant. In particular, BuSTs seem suitable to deal with pattern discovery problems, like Transcription Factor Binding Sites identification. A deeper investigation in this direction is planned for the future.

Contact email: luca.bortolussi@dimi.uniud.it