Algorithms on Grammar-Compressed Strings

Gad M. Landau*

Department of Computer Science, University of Haifa, Israel
Department of Computer Science and Engineering, NYU-Poly, Brooklyn, NY, USA
landau@cs.haifa.ac.il

Grammar based compression, where one replaces a long string by a small context-free grammar that generates the string, is a simple and powerful paradigm that captures many of the popular compression schemes, including the Lempel-Ziv family, Run-Length Encoding, Byte-Pair Encoding, Sequitur and Re-Pair.

Let S be a string of length N given as a grammar $G(S)$ of size n. The random access problem is to compactly represent $G(S)$ while supporting fast random access queries. That is, given an index i, report $S[i]$ without decompressing S. We will first present a linear space representations of $G(S)$ that supports $O(\log N)$ random access time. This representation extends to efficiently support substring decompression. Namely, we can decompress any substring $S[i]...S[j]$ in the same complexity as a random access query and additional $O(j-i)$ time.

Once we obtain an efficient substring decompression method, it can then serve as a basis for a compressed version of classical pattern matching. Namely, we can take any black-box (uncompressed) approximate pattern matching algorithm and turn it into a corresponding algorithm over grammar compressed strings. We will then focus on a specific algorithm for computing the edit distance of two grammar-compressed strings. This algorithm requires $O(nN)$ time and uses the compression in a more complicated way (i.e., not through random access queries)

* Partially supported by the National Science Foundation Award 0904246, Israel Science Foundation grant 347/09, Yahoo, Grant No. 2008217 from the United States-Israel Binational Science Foundation (BSF) and DFG.

R. Giancarlo and G. Manzini (Eds.): CPM 2011, LNCS 6661, p. 1, 2011.