Abstract

Suffix sorting — determining the lexicographical order of all the suffixes of a string — is one of the most important problems in string processing. The resulting data structure is called the suffix array (SA) and underpins dozens of applications in bioinformatics, data compression, and information retrieval. When the size of the input string or the SA exceeds that of internal memory (RAM), an external memory (EM) suffix sorting algorithm must be used. The most scalable of these EM methods is due to Bingmann et al. (Proc. ALENEX 2013), and is essentially a careful disk-based implementation of the so-called induced sorting technique used by the fastest RAM suffix sorting algorithms. In this paper we show how to greatly improve the efficiency of induced suffix sorting in external memory via a non-trivial reorganization of the computation involved. Our experiments show this new approach to be twice as fast as state-of-the-art methods, while, just as significantly, using a third of the disk memory. We also demonstrate the efficacy of our implementation for handling strings on large alphabets (with many millions of distinct symbols), which is important, e.g., for applications in natural language processing and information retrieval, but unaddressed by previous EM suffix sorting implementations. Our implementation uses a (EM) radix heap data structure and, as a side result of independent interest, we introduce a new operation for radix heaps and other monotone priority queues called min-comp, which we believe to be useful for many other applications, including discrete event simulation and sweep line algorithms, even in internal memory.