Priority Queues and Sorting for Read-Only Data[⋆]

Tetsuo Asano[1], Amr Elmasry[2], and Jyrki Katajainen[3]

[1] School of Information Science, Japan Advanced Institute for Science and Technology
Asahidai 1-1, Nomi, Ishikawa 923-1292, Japan
[2] Department of Computer Engineering and Systems, Alexandria University
Alexandria 21544, Egypt
[3] Department of Computer Science, University of Copenhagen
Universitetsparken 1, 2100 Copenhagen East, Denmark

Abstract. We revisit the random-access-machine model in which the input is given on a read-only random-access media, the output is to be produced to a write-only sequential-access media, and in addition there is a limited random-access workspace. The length of the input is N elements, the length of the output is limited by the computation itself, and the capacity of the workspace is $O(S + w)$ bits, where S is a parameter specified by the user and w is the number of bits per machine word. We present a state-of-the-art priority queue—called an adjustable navigation pile—for this model. Under some reasonable assumptions, our priority queue supports *minimum* and *insert* in $O(1)$ worst-case time and *extract* in $O(N/S + \lg S)$ worst-case time, where $\lg N \leq S \leq N/\lg N$. We also show how to use this data structure to simplify the existing optimal $O(N^2/S + N \lg S)$-time sorting algorithm for this model.

1 Introduction

Problem Area. Consider a sequential-access machine (Turing machine) that has three tapes: input tape, output tape, and work tape. In space-bounded computations the input tape is read-only, the output tape is write-only, and the aim is to limit the amount of space used in the work tape. In this set-up, the theory of language recognition and function computation requiring $O(\lg N)$ bits[1] of working space for an input of size N is well established; people talk about log-space programs [25, Section 3.9.3] and classes of problems that can be solved in log-space [25, Section 8.5.3]. Also, in this set-up, trade-offs between space and time have been extensively studied [25, Chapter 10]. Although one would seldom be forced to rely on a log-space program, it is still interesting to know what can be accomplished when only a logarithmic number of extra bits are available.

[⋆]

[1] Throughout the paper we use $\lg x$ as a shorthand for $\log_2(\max\{2, x\})$.