Data Structures and Algorithms

Practice sheet

Range minima data structure

Recall the $O(n \log n)$ size data structure for range minima problem that we discussed in the lectures. It took $O(1)$ time to answer a query. Attempt the following problems that are based on this data structure.

1. **Preprocessing the data structure**
   As discussed in the lectures, the time needed to build the data structure is called its preprocessing time. Realize that the preprocessing time of a data structure is different from the query time of the data structure. The data structure for range minima required an $n \times \log_2 n$ matrix C such that $C[i, k]$ stored the smallest element from $C[i], \ldots, C[i + 2^k]$. Show that the matrix C can be built in $O(n \log n)$ time only.

2. **Range minima with $\mathcal{O}(n)$ space**
   The data structure for range minima we discussed takes $O(n \log n)$ space. Some applications of range minima can not afford more than linear space, that is, $O(n)$ space. You need to suitably use the data structure we discussed in the lectures to design another data structure for range minima that takes smaller space at the expense of increased query time. In particular, the new data structure needs to occupy $O(n)$ space only though the query time it needs to guarantee is $O(\log n)$.

Note: Hints are given on the next page.

1. Fill the entries of C in the increasing order of the columns. How will you compute $C[\_, 0]$. Try to express $C[i, k]$ using just 2 entries of the same matrix. Make use of the query answering algorithm to determine these 2 entries.

2. Try to reduce the array size to $\frac{n}{\log n}$ suitably and then build the range minima data structure we discussed in the lectures for this reduced size array. How much is the size of this data structure ? Notice that this data structure will not be sufficient to report the minimim element for each range $[i, j]$. How will you answer a range minima query for any arbitrary range $[i, j]$? Here, you should utilize the fact that you can afford to answer a query in $O(\log n)$ time. You may assume n is power of 2.