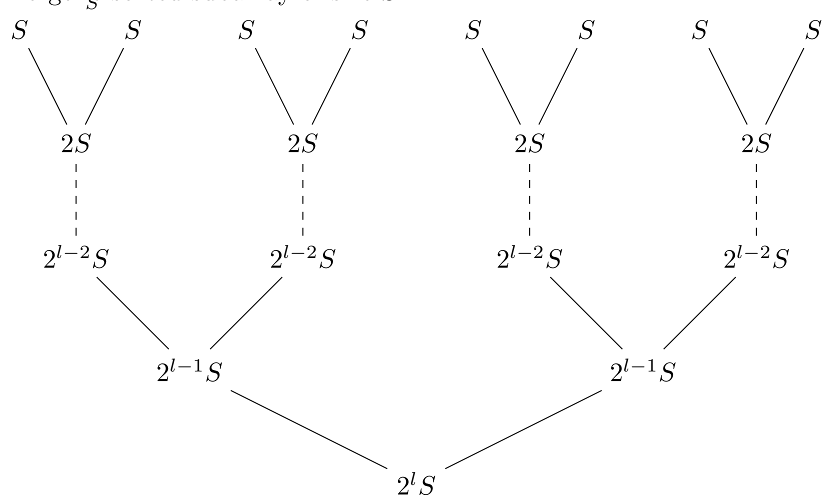
SC2001: Example class 1

Theoretical Analysis

September 16, 2024

1 Hybrid Merge-Insertion Sort Analysis

1. Let n be the input size / no. of elements to be sorted.
2. Divide n elements into $\frac{n}{S}$ subarray of size S, the size of each subarray.
3. For each subarray of size n_s, perform insertion sort:
   - Best case: $O(S)$
   - Worst case: $O(S^2)$
4. Since there are $\frac{n}{S}$ subarrays of size S, total cost of applying insertion sort as:
   - Best case: $\frac{n}{S} \cdot S = O(n)$
   - Worst case: $\frac{n}{S} \cdot S^2 = O(nS)$
5. Merge $\frac{n}{S}$ sorted subarray of size S



- Each `merge()` doubles the subarray size. Suppose l iterations of `merge()` has to be performed to recover input size n:

$$\begin{aligned} 2^l S &= n \\ 2^l &= \frac{n}{s} \\ l \lg 2 &= \lg\left(\frac{n}{s}\right) \\ l &= \log_2\left(\frac{n}{s}\right) \end{aligned} \tag{1}$$

- Since the cost of `merge()` is $O(n)$, the worst case total cost of merging M is:

$$\begin{aligned} M &= l \cdot n \\ &= \log_2\left(\frac{n}{s}\right) \cdot n \\ &= O(\lg\left(\frac{n}{s}\right) \cdot n) \end{aligned} \tag{2}$$

6. Combining insertion-sort & merging in Hybrid Merge-Insertion sort, we have:

   - Best Case: $O(n + \lg\left(\frac{n}{s}\right) \cdot n)$
   - Best Worse: $O(nS + \lg\left(\frac{n}{s}\right) \cdot n)$