Data Structures and Algorithms

István András Seres @Eötvös Loránd University

2019 March 12, Tuesday 12:30-14:00, D-0.221

1 Warm-up and recap: I. Merge sort

Illustrate the execution of the merge-sort algorithm on the array

$$A = (3, 13, 89, 34, 21, 44, 99, 56, 9)$$

For each fundamental iteration or recursion of the algorithm, write the content of the array. Assume the algorithm performs an in-place sort.

2 Warm-up and recap: I. Insertion sort

Write the pseudo-code of the insertion-sort algorithm. Illustrate the execution of the algorithm on the array

$$A = (3, 13, 89, 34, 21, 44, 99, 56, 9)$$

, writing the intermediate values of A at each iteration of the algorithm.

3 Practicing heap sort

Insert the following elements in sequence into an empty heap:

$$A = (6, 8, 4, 7, 2, 3, 9, 1, 5)$$

. Draw both the tree and array representations of the heap. Then illustrate building a heap from the same data.

4 Heaps don't lie

Show that the element with the lowest priority (highest number)

- can be found in one of the leaf-nodes
- that it can be found in an arbitrary leaf-node (it's sufficient to show that the requirements of a binary heap holds with different arrangements)
- show that a binary heap with an even number of elements (N) will have exactly N/2 leaf-nodes. This assignment is perhaps a bit difficult, try to think about the data-structure we used to represent a binary heap

5 A boolean array

Suppose you have an array of booleans of size n. Give an $\mathcal{O}(n)$ algorithm that sorts the array so that all false elements come before all true elements. Your algorithm should use $\mathcal{O}(1)$ extra memory.

6 Las Vegas and Monte Carlo algorithms

As we have already seen randomized algorithms in certain cases are superior to deterministic algorithms. In this exercise we further investigate randomized algorithms. László Babai introduced Las Vegas algorithms as the duals of Monte Carlo algorithms. Consider the following algorithmic task.

Input: An array of $n \geq 2$ elements, in which half are 'a's and the other half are 'b's. **Output:** Find an 'a' in the array.

```
while 1 do
    k = RandInt(n);
    if A[k] == 1 then
        return k;
    else
end
```

Algorithm 1: Las Vegas algorithm

In the following let l be a predifined fixed constant.

```
i = 0;
while i == l ∨ A[k] ==' a' do
    k = RandInt(n);
    i = i + 1;
end
```

Algorithm 2: Monte Carlo algorithm

It is easy to see that Las Vegas algorithm always returns with a correct output, however its running time is a random variable. What is the expected running time of the Las Vegas algorithm? Monte Carlo is ensured to terminate in less than l steps, however it might not give correct result. What is the soundness error of Monte Carlo algorithm for a fixed l?

Randomized algorithms are particularly useful when faced with a malicious "adversary" or attacker who deliberately tries to feed a bad input to the algorithm. It is for this reason that randomness is ubiquitous in cryptography.