Binary Heaps

This repository contains the code which provides the array-based implementation of binary heaps.

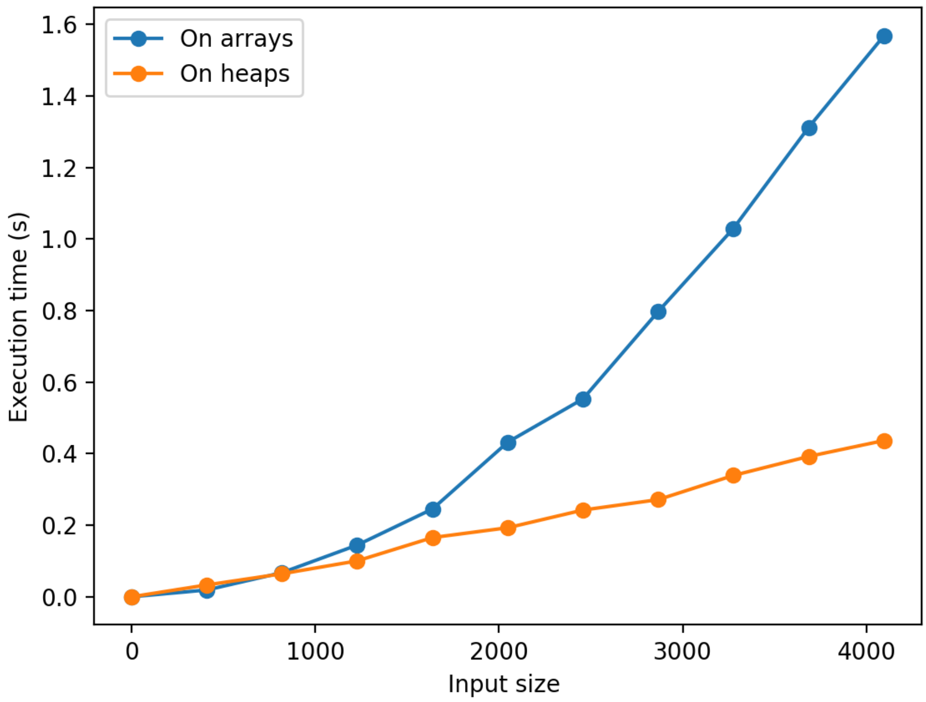
In folder `include` there is the `binheap.h` header in which is implemented the type `binheap_type`, while in folder `src` there is the `binheap.c` file in which are implemented all the functions.

A Makefile can be produced by using `cmake` as follows:

```
cmake -G "Unix Makefiles" CMakeLists.txt
```

Compiling the code, it is possible to test the performance by executing the programs in folder `tests`.

The following plot shows the execution time taken by both the array based and the heap based version when extracting the minimun from the heap.



Ex. 6.1-7

In a binary heap containing n nodes, the number of non-leaf nodes can not be greater than $\lfloor \frac{n}{2} \rfloor$, otherwise the right child of the last non-leaf node would be outside the range of the heap since its array index ($2 * \lfloor \frac{n}{2} \rfloor + 1$) would be greater than n. Hence, all parent nodes are indexed from 1 to $\lfloor \frac{n}{2} \rfloor$ and since in the array representation of a heap all non-leaf nodes are stored before leaf nodes, we can deduce that the leaves are indexed by $\lfloor \frac{n}{2} \rfloor + 1, \lfloor \frac{n}{2} \rfloor + 2, \dots, n$.

Ex. 6.2-6

If the maximum value in the heap is on the root, `heapify` is called recursively until a leaf is reached; thus, it is invoked h times, where $h = log(n)$ is the heap height, and its execution time is $\Theta(log(n))$. Hence, the worst-case running time of the procedure is $\Omega(log(n))$.

Ex. 6.3-3

From `Ex. 6.1-7`, we know that a binary heap B_0 containing n nodes has at most $N_0 = \frac{n}{2}$ leaves, which are nodes with height $h = 0$. Thus, the result is true for $h = 0$. Suppose the result is true for $h - 1$. Let be N_h the number of nodes at height h in the binary heap B_h which contains n nodes. Let be B_{h-1} the binary heap obtained by removing the leaves from B_h; thus B_{h-1} has $\frac{n}{2}$ nodes and the nodes at height h in B_h would be at height $h - 1$ in heap B_{h-1}. Thus, the number of nodes at height h is $N_h \leq \lceil \frac{n}{2^{h+1}} \rceil$.