

INFO-0027: Programming techniques

Project 1: Performance study

Goffart Maxime  
180521

Joris Olivier  
182113

Academic year 2020 - 2021

# 1 Performance study

We decided to perform our performance study by doing 1000 different creations of the **MAGIC** A.D.T. and by measuring the mean time passed in the **MAGICindex** and **MAGICreset** functions. We chose to deal with an increasing number of addresses in order to be able to analyse the differences between the implementations on graphics. The addresses we used to measure the performances of these functions were randomly generated 4 bytes addresses.

## 1.1 First implementation : Hash table

We resumed the total time passed in the A.D.T. functions on the graphic represented on the [Figure 1](#).

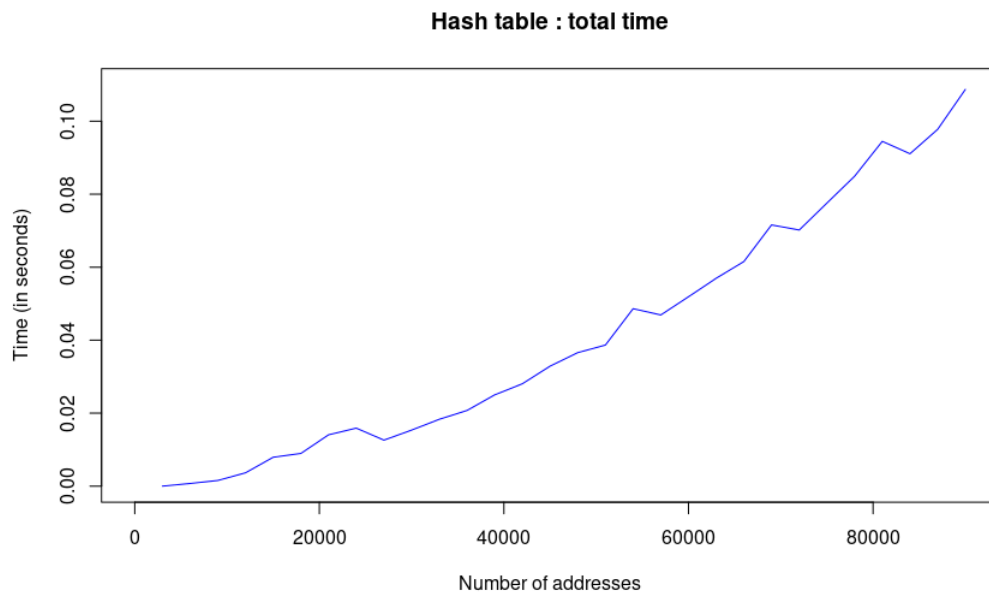


Figure 1: Total mean time passed in A.D.T. functions over 1000 tests according to the number of addresses stored in the structure.

### 1.1.1 Time passed in the **MAGICindex** function

We resumed the time passed in the **MAGICindex** function on the graphic represented on the [Figure 2](#).

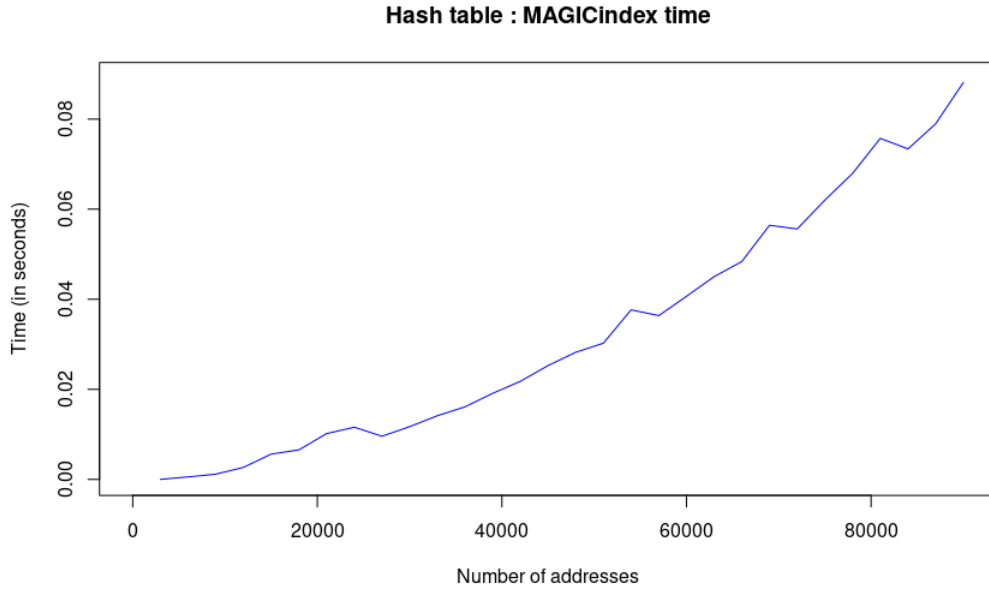


Figure 2: Mean time passed in the `MAGICindex` function over 1000 tests according to the number of addresses stored in the structure.

### 1.1.2 Time passed in the `MAGICreset` function

We resumed the time passed in the `MAGICreset` function on the graphic represented on the [Figure 3](#).

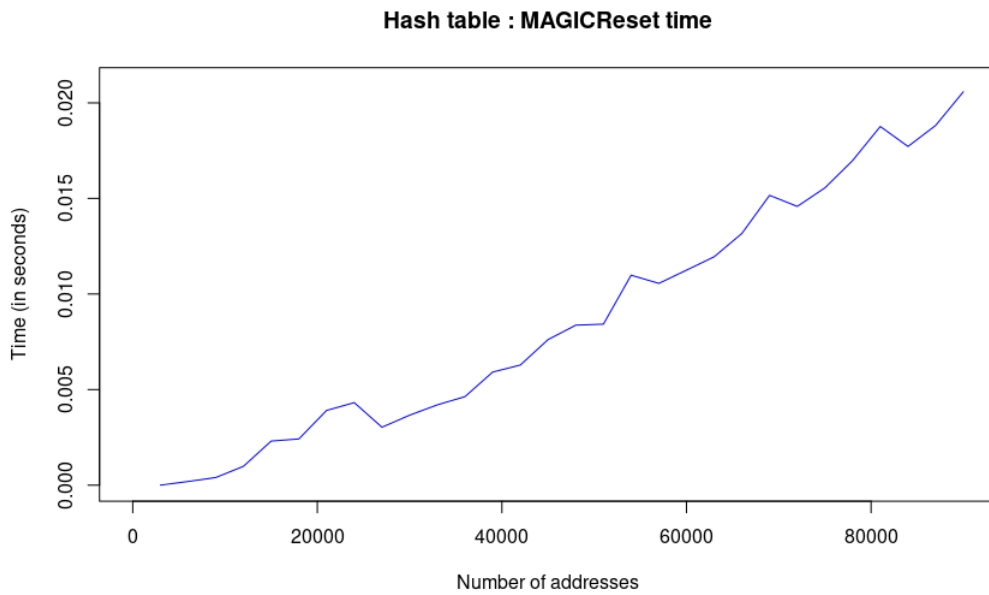


Figure 3: Mean time passed in the `MAGICreset` function over 1000 tests according to the number of addresses stored in the structure.

## 1.2 Second implementation : Ternary search trie

We resumed the total time passed in the A.D.T. functions on the graphic represented on the [Figure 4](#).

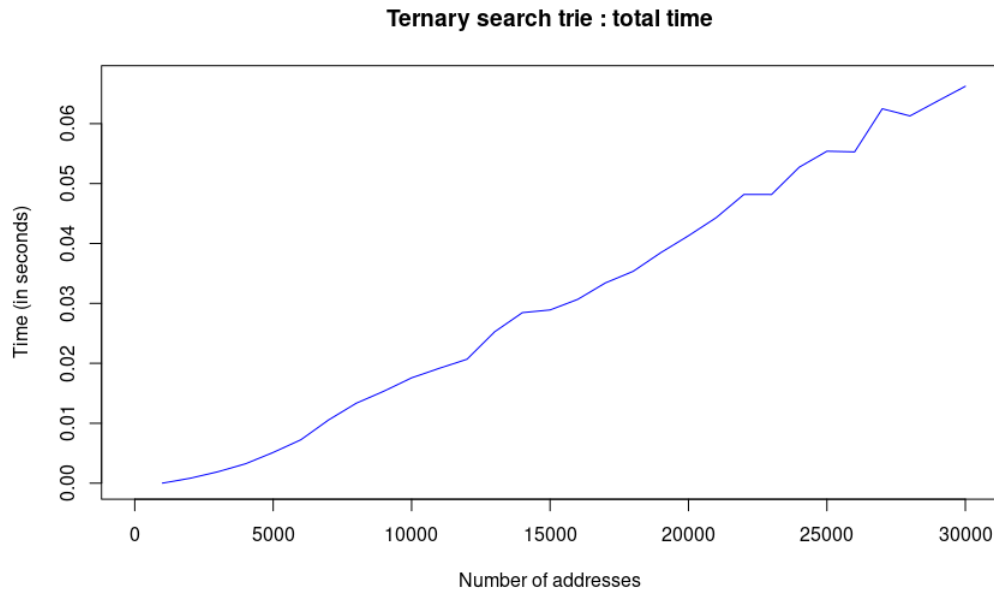


Figure 4: Total mean time passed in A.D.T. functions over 1000 tests according to the number of addresses stored in the structure.

### 1.2.1 Time passed in the MAGICindex function

We resumed the time passed in the MAGICindex function on the graphic represented on the [Figure 5](#).

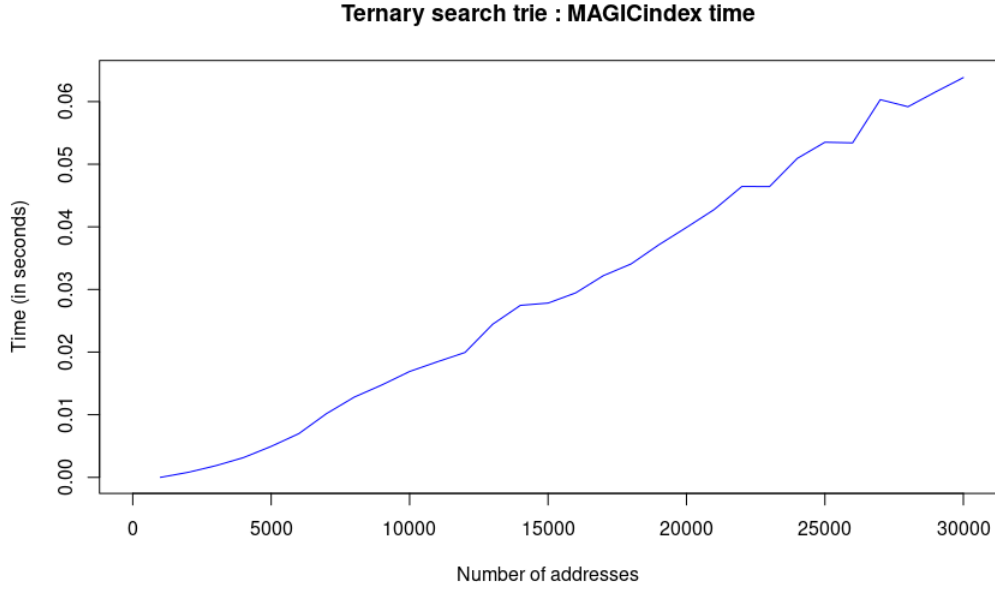


Figure 5: Mean time passed in the `MAGICindex` function over 1000 tests according to the number of addresses stored in the structure.

### 1.2.2 Time passed in the `MAGICreset` function

We resumed the time passed in the `MAGICreset` function on the graphic represented on the [Figure 6](#).

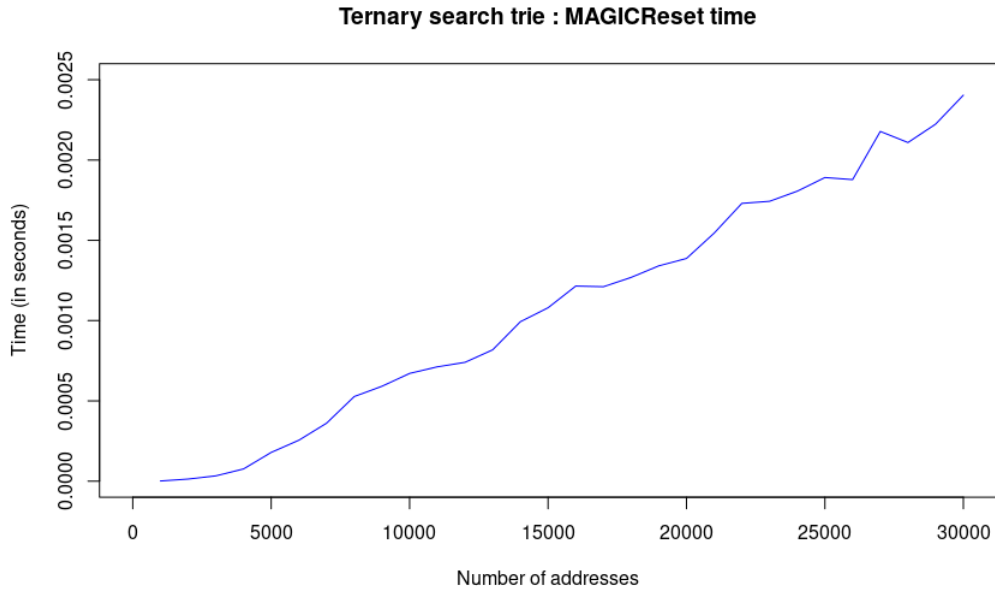


Figure 6: Mean time passed in the `MAGICreset` function over 1000 tests according to the number of addresses stored in the structure.

### 1.3 Conclusion

Our first implementation (hash table) seems to have better performances than our second one (ternary search trie) in the **MAGICindex** implementation. However, our second implementation has better performances in the **MAGICreset** implementation.

In total, our first implementation seems to have better performances than our second one in terms of time.