# Relazione Algoritmi e Strutture Dati

Francesco Mauro, matricola: 949471

A.A. 2022/2023

# Contents

# Chapter 1

# Mege Binary Insertion Sort

## 1.1 Mege Binary Insertion Sort implementation

### 1.1.1 Introduction

The purpose of this report is to provide the tests that have been done to find the best value of K for the Merge Binary Insertion Sort algorithm. Considering that Merge Binary Insertion Sort is a hybrid algorithm that has ability to handle large input and for its speed, but it became inefficient when having a small input, in the case of small input the library switch to Binary Insertion sort, that is more efficient on small input.

### 1.1.2 Testing methodolgy

To test the algorithm, I used a bash script[1] that run the program with different values of K and for each field to sort, saving the time and the value of the algorithm parameter K.

[1] All the test are done on a Lenovo Thinkpad x390 yoga with an Intel Core i7-8665U and 16 gb of ram

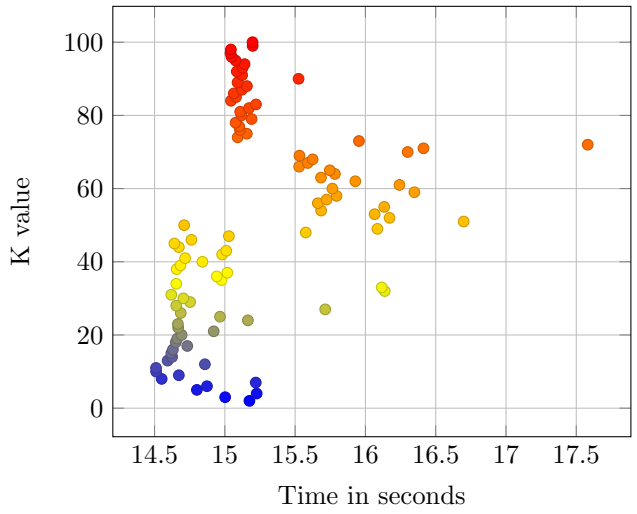


Figure 1.1: Time used to sort string field with different K values

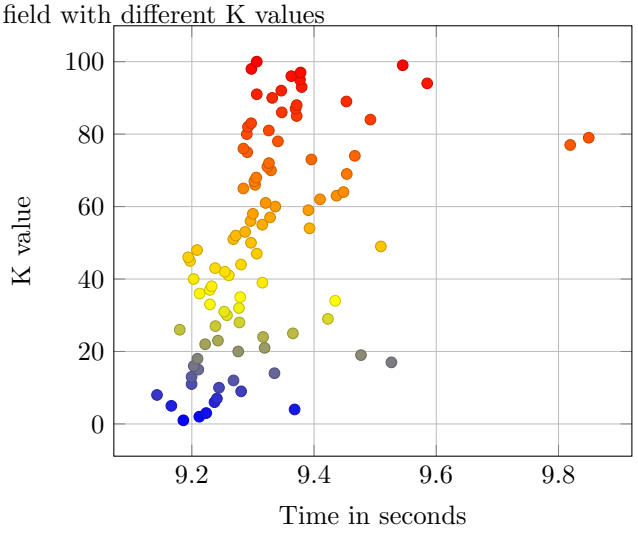


Figure 1.2: Time used to sort integer field with different K values

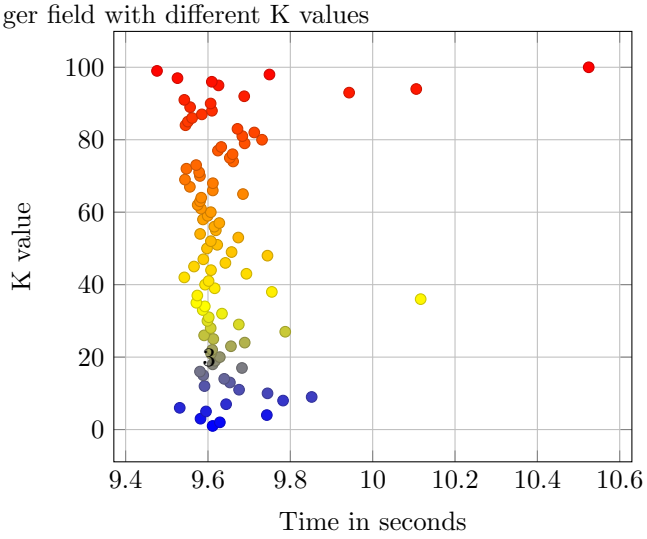


Figure 1.3: Time to sort double field

## 1.2 Conclusion based on the analysis

As we can see from the graphs, the optimal values for k is around 10 and 20, and the fact that based on the actual implementation is also important the type of field that the algorithm is sorting. In fact when sorting integer or double the algorithm is taking way less time than sorting string, probably caused by miss branch prediction. Also in the graph we can see some outliers, this can be caused by bottleneck of other processes or just the fact that the laptop in some point was charging.

# Chapter 2

# Skip List

## 2.1 Introduction

The purpose of this chapter is to report the tests that have been done to find the best value of height in a Skip List. Skip List is a probabilistic data structure that allows searching, insertion and deleting operation with time complexity of $O(logn)$.

## 2.2 Testing methodology

I wrote a bash script[1] that runs the program with different values of height and saves the output in a file, the range of tested level is between 6 and 50

[1] Included in the repo,the name is **time_taker_ex2.sh**

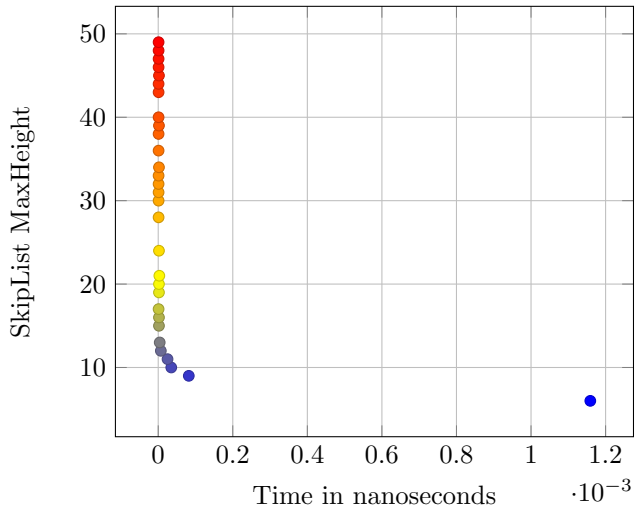


Figure 2.1: Graph 1: SkipList Max-Height vs. Time

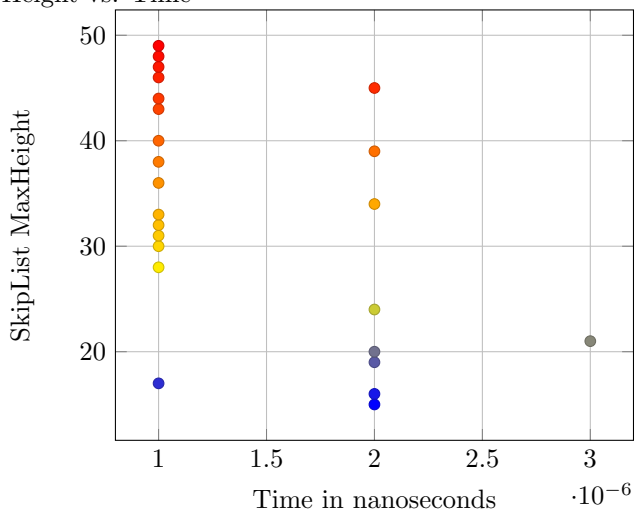


Figure 2.2: Graph 2: SkipList Max-Height vs. Time (Focused)

Range from 1 to 5 is omitted because the algorithm did not terminate for those values [2]

[2]all test are done on a Lenovo Thinkpad x390 yoga with an Intel Core i7-8565U CPU and 16GB of RAM, with Arch Linux installed as only OS

## 2.3 Conclusion based on the analysis

Looking the graph the best values is between 15 and 20, because this range is the good balance between the time used to search the words in the skip list and the memory used.

## 2.4 Knwon Issues

Sometimes the main program gives a segmentation fault error caused by the **fgets()**,used to read the phrase to correct, on 50 run the program gave segmentatin fault 7 times.

# Chapter 3

# Priority Queue and Primm Algorithm