Exploration and Presentation - Assignment 4

Martin, Frederiksen
cph-mf237@cphbusiness.dk

Andreas, Vikke
cph-av105@cphbusiness.dk

Asger, Sørensen
cph-as466@cphbusiness.dk

William, Huusfeldt
cph-wh106@cphbusiness.dk

19. april 2021

Indhold

Indledning

Vi arbejder med opgaven **letterfrequencies**. I denne version af rapporten gennemgår vi også den feedback vi har fået til vores rapport og program. Vi introducere hvad vi har gjort anderledes og gennemgår de ændringer vi har lavet.

Kapitel 1

Opgaven

1.1 Opgave 1

1. Find a point in your pgrom that can be optimized (for speed), for example by using a profiler.
2. Make a measurement of the point to optimize, for example by running a number of times, and calculating the mean and standard deviation (see the paper from Sestoft).
3. If you work on the **letterfrequencies** program, make it at least 50% faster.

Kapitel 2

Løsningen

2.1 Programmet

Vi har valgt at arbejde på programmet **letterfrequencies**. Vi havde problemer med at få profileren fra intellij til at fungere korrekt og derfor har vi arbejdet ud fra det eneste hint vi fik fra intellij hvilket var at maps put funktion skulle optimeres. Vi starter lige med at angive spredningen udregnet på 10000 gennemkørsler:

$$56859375,2 \text{ ns} \pm 10085179,175$$

Oprindeligt troede vi ud fra det resultat vi fik fra profileren fra IntelliJ at problemet lå i maps put metode. Vi har siden da lært at put kører O(1) og ikke er en bottleneck for vores program. Det som virker til at have hjulpet mest på optimeringen er at vi valgte at bruge er long array i stedet for et hashmap. Vi holder os til den følgende optimering da vi stadig har fået en god hastighedsforøgelse. Vores optimiseret program ser derfor således ud:

```
1  public class Main {
2
3      public static void main(String[] args) throws FileNotFoundException, IOException {
4          String fileName = "C:/Users/Andreas Vikke/OneDrive/Documents/Skole/SoftwareUdvikling
       /CPH-Business-UFO/Week13/letterfrequencies/src/main/resources/FoundationSeries.txt";
5          Reader reader = new FileReader(fileName);
6          long freq[] = new long[256];
7          tallyChars(reader, freq);
8          print_tally(freq);
9      }
10
11     public static double Run() throws FileNotFoundException, IOException {
12         String fileName = "C:/Users/Andreas Vikke/OneDrive/Documents/Skole/SoftwareUdvikling
       /CPH-Business-UFO/Week13/letterfrequencies/src/main/resources/FoundationSeries.txt";
13         Reader reader = new FileReader(fileName);
14         long freq[] = new long[256];
15         tallyChars(reader, freq);
16         print_tally(freq);
17         return 1;
18     }
19
20     private static void tallyChars(Reader reader, long[] freq) throws IOException {
21         int b;
22         while ((b = reader.read()) != -1) {
23             freq[b]++;
24         }
25     }
26
27     private static void print_tally(long[] freq) {
28         int dist = 'a' - 'A';
29         Map<Character, Long> upperAndlower = new LinkedHashMap();
```

```
        for (Character c = 'A'; c <= 'Z'; c++) {
            upperAndlower.put(c, freq[c] + freq[c + dist]);
        }

        Map<Character, Long> sorted = upperAndlower
                .entrySet()
                .stream()
                .sorted(Collections.reverseOrder(Map.Entry.comparingByValue()))
                .collect(
                        toMap(Map.Entry::getKey, Map.Entry::getValue, (e1, e2) -> e2,
                                LinkedHashMap::new));
    }
}
```

Listing 2.1: Main.java

Vi kan nu tage et kig på den nye spredning som også er udregnet på 10000 gennemkørsler:

$$29067911,1 \text{ ns} \pm 282264,715$$

Vi kan se den procentvise hastighedforøgelse:

$$\frac{56859375,2 - 29067911,1}{56859375,2} \times 100 \approx 48,877\%$$

Etersom vi ikke havde mulighed for flere hints fra intellij prøvede vi lige en hurtig google søgning: "how to optimze a file reader"og fandt ud af at vi kunne optimisere koden yderligere vha. en bufferedReader. BufferedReader er hurtigere fordi den "buffer"16kb af filen som den så leder igennem imens den spørger harddisken om de næste 16kb af filen. På denne måde skal bufferedReader kun vente på svar fra harddisken 1 gang og det er ved de første 16kb. Vi har tilføjet bufferedReader således:

```
    public static void main(String[] args) throws FileNotFoundException, IOException {
        String fileName = "C:/Users/Andreas Vikke/OneDrive/Documents/Skole/SoftwareUdvikling
    /CPH-Business-UFO/Week13/letterfrequencies/src/main/resources/FoundationSeries.txt";
        BufferedReader reader = new BufferedReader(new FileReader(fileName), 16384);
        long freq[] = new long[256];
        tallyChars(reader, freq);
        print_tally(freq);
    }
```

Listing 2.2: NewMain.java

Vi får så følgende spredning igen udregnet på 10000 gennemkørsler:

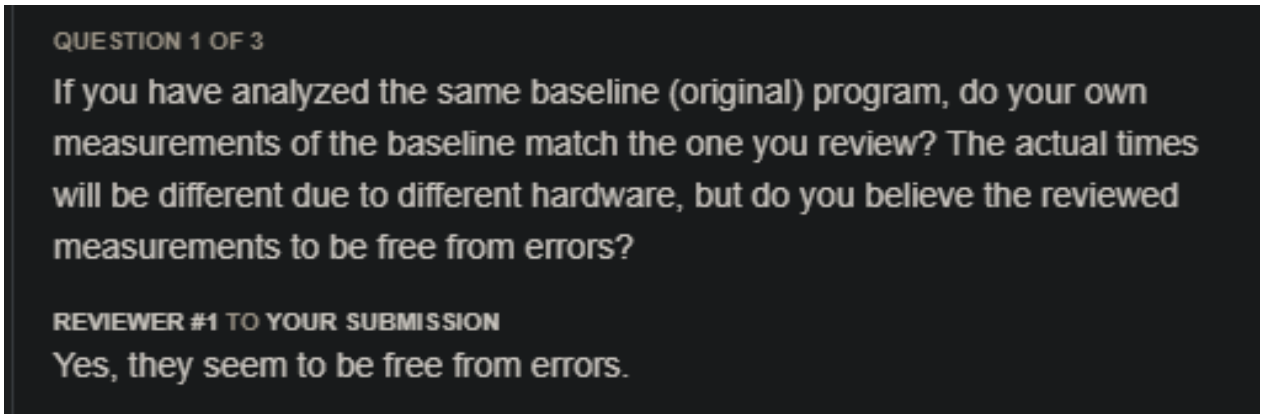
$$19755513,7 \text{ ns} \pm 806225,216$$

Afslutningsvis ender vi med følgende hastighedforøgelse i procent:

$$\frac{56859375,2 - 19755513,7}{56859375,2} \times 100 \approx 65,255\%$$

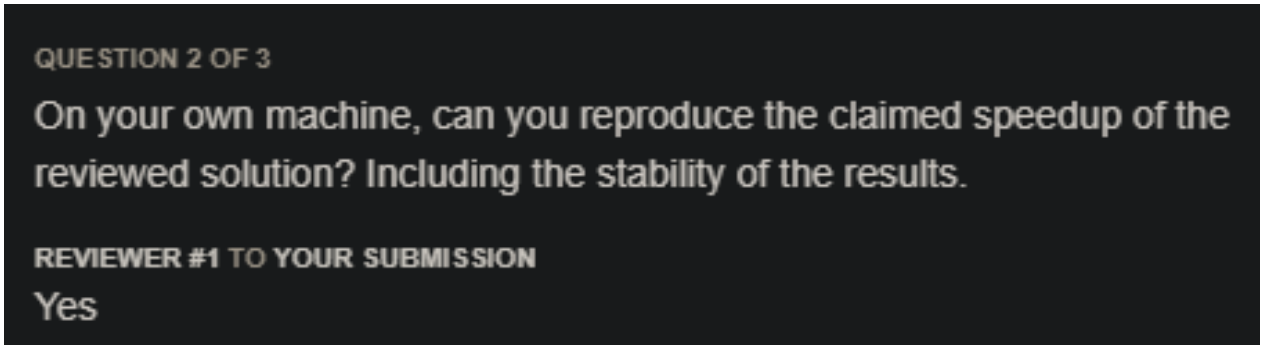
2.2 Feedback

På de næste tre billeder kan man se den feedback som vi har fået fra opgave 3. Vi har som helhed ikke modtaget meget konstruktiv kritik, som derved gør det svært at forbedre vores rapport.



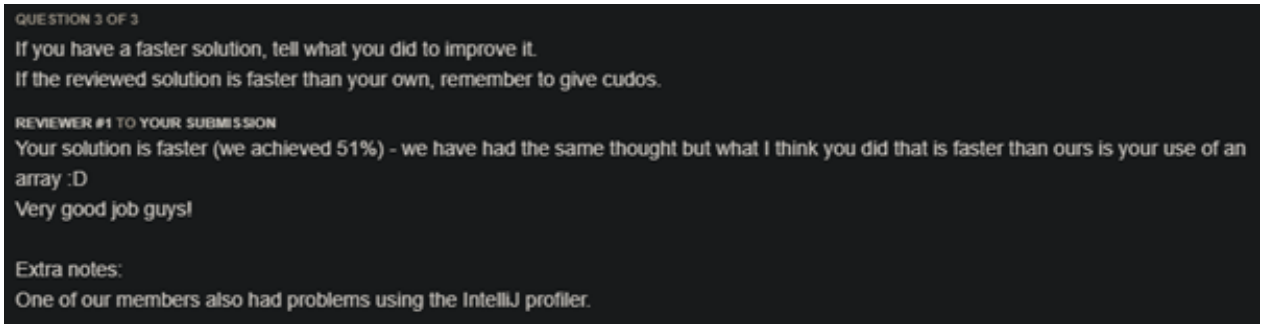
Figur 2.1: Feedback på spørgsmål1.

Som set på billedet nedenfor får vi fortalt at vi ikke har nogle "fejl" I vores beregninger, som er positivt, men ikke giver noget vi kan arbejde videre på.



Figur 2.2: Feedback på spørgsmål2.

På billede nedenfor får vi igen kort svar, havde vi fået at vide hvad for nogle resultater de fik af at køre vores program, kunne vi sammenligne og diskutere de to forskellige svar.



Figur 2.3: Feedback på spørgsmål3.

2.3 Konklusion

Vi endte med en hastighedsforøgelse på 65,255% hvilket vi selv er tilfredse med. Efter vores feedback fra peergrade har vi gennemlæst opgaven igen og rettet en enkelt ting der var forkert forklaret. Derudover har vi oprettet dette afsnit 2.3 eftersom dette manglede. Vi har endvidere oprettet et afsnit 2.2 hvor vi har skrevet lidt til den feedback vi har fået. I bund og grund var der ikke rigtig noget at ændre på baggrund af den givne feedback men vi har alligevel lavet et par småting om.