Exploration and Presentation - Exam Assignment

Martin, Frederiksen
cph-mf237@cphbusiness.dk

Andreas, Vikke
cph-av105@cphbusiness.dk

27. maj 2021

1 Resumé

Vi som studerende på professionsbacheloruddannelsen i softwareudvikling er ikke oplyst nok om, hvilket sprog der bør benyttes, når der er tale om hastighed. Dette problem er en gennemgående tendens i alle fag, da vi som studerende selv har mulighed for at vælge, hvilket programmeringssprog vi gerne vil benytte til diverse opgaver. En løsning på dette problem kan være at teste hastigheden på forskellige sprog, der udfører det samme stykke kode, hvilket vi har gjort med Python og C#. Vi fandt herved ud af at C# er markant hurtigere end Python til at gennemkøre insertion sort algoritmen og derved er C# vinderen af vores lille test.

Indhold

2 C# vs Python

Vores gruppe har valgt at undersøge hastighedsforskellen imellem Python og C#. Denne undersøgelse er blevet foretaget ud fra et stykke kode, vi skrev i starten af faget *Diskret matematik og algoritmer*. Kodestumpen blev oprindeligt skrevet i C#, men i forbindelse med denne opgave omskrev vi hurtigt kodestumpen til Python, så begge stykker kode udfører samme opgave. Begge programmeringssprog har sine fordele og ulemper og det er vigtigt at have dem in mente, når der skal vælges er programmeringssprog til en given opgave. Som eSparkbiz Technologies skriver i [1] har Python og C# følgende fordele og ulemper:

Python fordele:

- Sproget er portable, hvilket betyder at det kan køress på flere platforme som fx. Linux, Macintosh, windows osv.
- Sproget tillader programmøre at udvikle en grafisk brugergrænseflade.
- Sproget blev etableret for 28 år siden og derved huser sproget et stort fælleskab.
- Sproget er effektivt eftersom der kun skal skrives få linjer kode.

C# fordele:

- Sproget er skalerbart og nemt at opdatere.
- Sproget har hurtig kompilering.
- Sproget er type stærkt da det ikke tillader risikable casts.
- Sproget er et struktureret programmeringssprog.
- Sproget besider krydsplatform evner.
- Sproget har et standard bibliotek.

Python ulemper:

- Sproget er langsomt.
- Sproget kan ikke bruges til mobil computing.
- Sprogets database er primitiv og underudviklet.
- Sproget har fejl og mangler som let kommer til udtryk.
- Sproget behøver meget testing.

C# ulemper:

- Sproget er ikke ment til underordnet programmering.
- Sproget har ikke nogen ubegrænset kompiler.
- Sproget kræver en kompilering hver gang der er sket en ændring i koden.
- Sproget kræver grundig testing.
- Sprogets eksekverbare filer kan kun eksekveres på windows.

Disse fordele og ulemper er med til at danne et hurtigt overblik over, hvad vi kan forvente af de to sprog. Desuden præger disse fordele og ulemper også vores hypotese.

3 Hypotese for hastighed af de to sprog

Python og C# er bygget på to forskellige typer programmeringssprog, som har stor indflydelse på hastigheden af programmet, man laver. C# er et compiled sprog, hvilket vil sige at koden er oversat til maskinkode ved brugen af en compiler. På den måde vil man få et meget effektivt program, som kan køres flere gange. Python er i modsætning til C# et interpreted programmeringssprog, som skal analysere, fortolke og eksekvere for hver gang, programmet bliver kørt.

> "One of the jobs of a designer is to weigh the strengths and weaknesses of each language and then decide which part of an application is best served by a particular language" — IBM Corporation[2]

Som IBM nævner i [2] er det en vigtig kvalitet at kunne se styrker og svagheder for hver type sprog og herudfra vælge, hvilket sprog der er bedst egnet til den type applikation, der skal laves. Med disse ting in mente forudser vi at vores InsertionSort algoritme, vil kunne køre hurtigere i C# end Python, grundet Python vil skulle bruge længere tid på at eksekvere koden end C#, da den skal fortolke koden linje for linje.

4 Insertion sort

Vi har valgt at teste hastigheden af C# og Python ved hjælp af Insertion Sort. Valget blev truffet ud fra kodemængden og hvad vi i forvejen havde liggende. Insertion sort kan skrives i én metode, som gør at koden ikke ændrer sig betydeligt meget mellem de to sprog. Som det ses nedenfor, er Insertion Sort koden i C# og Python næsten identiske og ingen af dem importerer nogle eksterne pakker fra en tredjepart, som kan gøre at hastigheden svækkes.

```
1  public static double sort(int[] arr, int t) {
2      for (int i = 1; i < arr.Length; ++i) {
3          int tempVal = arr[i];
4          int j = i - 1;
5
6          while (j >= 0 && arr[j] > tempVal) {
7              arr[j + 1] = arr[j];
8              j = j - 1;
9          }
10         arr[j + 1] = tempVal;
11     }
12     return t;
13 }
```

Listing 1: C# Insertion Sort

```
1  def sort(arr, t):
2      for i in range(1, len(arr)):
3          tempVal = arr[i]
4          j = i - 1
5
6          while j >= 0 and arr[j] > tempVal:
7              arr[j + 1] = arr[j]
8              j = j - 1
9
10         arr[j + 1] = tempVal
11     return t
```

Listing 2: Python Insertion Sort

5 Test af hastighed

For at teste hastigheden skal der bruges en profiler, der kan måle gennemsnitstiden af metoden. Denne profiler er ikke helt ens i C# og Python, men koden vil kun teste hastigheden af Insertion Sort metoden. På Listing 3 og 4 ses koden til profileren. Det kan ses her at der blev kører 1000 iterationer af 10000 metodekald, som der så kunne findes gennemsnittet af og udregner middelværdi og difference. Begge sprogs programmer blev kørt på en Ubuntu 20.1 i et WSL2 miljø på Windows. Windows system havde udelukkende de nødvendige ting samt WSL2 kørende, for at minimere ressourceforbruget fra andre programmer. Derved kunne WSL2 bruge så mange ressourcer som muligt under kørslen, som ville give et mere præcist resultat af den målte hastighed. Resourcerne der var til rådighed var 16GB ram, AMD Ryzen 7 1880X, Nvidia 1080 grafik kort.

```
public static double CheckMethodTime() {
    int n = 1000;
    int count = 10000;
    double dummy = 0.0;
    double st = 0.0, sst= 0.0;

    Stopwatch stopWatch = new Stopwatch();
    for(int j = 0; j < n; j++) {
        stopWatch.Restart();
        for(int i = 0; i < count; i++) {
            int[] arr = new int[] {10, 22, 14, 4, 2, 50, 75, 4};
            dummy += InsertionSort.sort(arr, i);
        }
        stopWatch.Stop();
        double ns = (stopWatch.Elapsed.TotalMilliseconds * 1000000) / count;
        st += ns;
        sst += ns * ns;
    }
    double mean = st/n, sdev = Math.Sqrt((sst - mean*mean*n)/(n-1));
    Console.WriteLine($"{mean.ToString("F1")} ns +/- {sdev.ToString("F1")} ns");
    return dummy;
}
```

Listing 3: C# Profiler

```
def CheckMethodTime():
    n = 1000
    count = 10000
    dummy = 0.0
    st = 0.0
    sst = 0.0

    for j in range(n):
        start = time.perf_counter_ns()
        for i in range(count):
            arr = [10, 22, 14, 4, 2, 50, 75, 49, 4]
            dummy += InsertionSort.sort(arr, i)
        stop = time.perf_counter_ns()
        ns = (stop - start) / count
        st += ns
        sst += ns * ns

    mean = st/n
    sdev = math.sqrt((sst - mean*mean*n)/((n-1)))
    print("{:.1f} ns +/- {:.1f} ns".format(mean, sdev))
    return dummy
```

Listing 4: Python Profiler

6 Resultatet

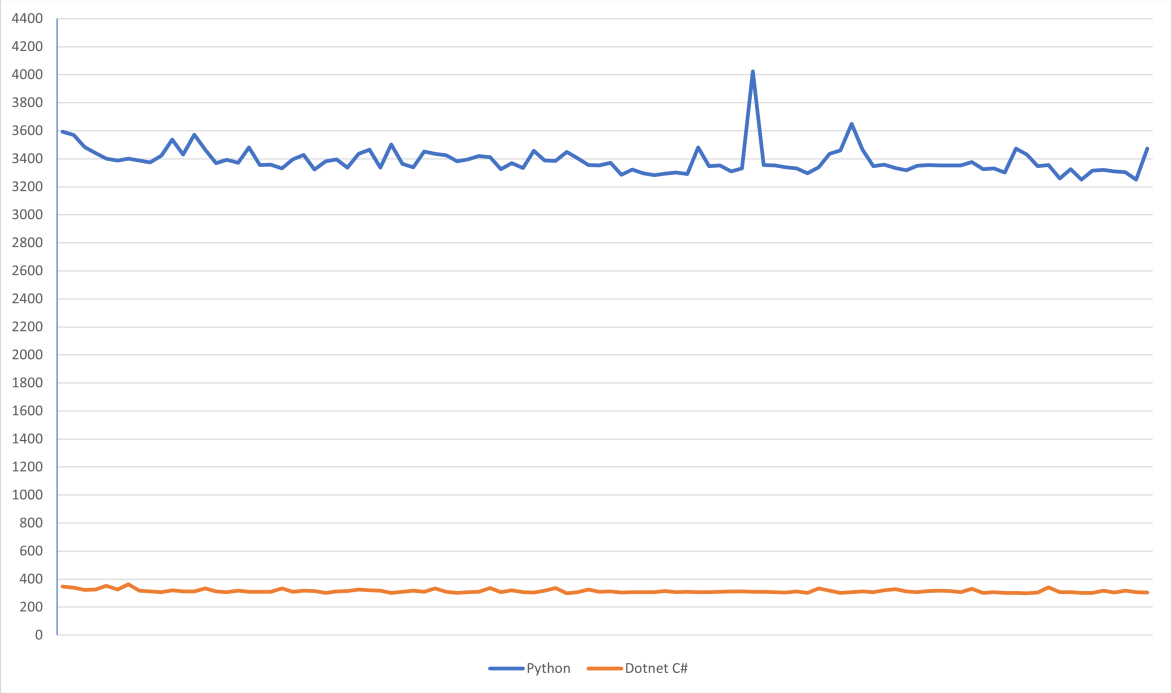
Efter kørslen af programmerne kunne vi sammenligne resultaterne, som kan ses herunder. Det ses at C# programmet kører betydeligt hurtigere end Python programmet.
Resultatets middelværdi lå meget højere i Python, som vi havde forventet, men udover dette er differencen også en del højere, som vi ikke havde forventet. Ved at skrive alle gennemsnitsværdierne ud i stedet for middelværdien, kunne vi plotte de to programmer, som kan ses på Figur 1. Dette gav et tydeligt indblik i stabiliteten af de to sprog, som viser at Python kører meget mere ustabilt også.

C#:

$$311,2 \text{ ns} \pm 13,1$$

Python:

$$3312,1 \text{ ns} \pm 88,9$$



Figur 1: Plot af hastighed på Insertion Sort

7 Verificering

Med den indsamlede empiri om fordele og ulemper af de to programmeringssprog samt test og opnåede resultater, kan vi hermed verificere vores hypotese.

Litteratur

[1] eSparkbiz Technologies. Python vs c#: Which would be a better option for you? https://www.esparkinfo.com/python-vs-c-sharp.html, 2021. [Online; accessed 03-Maj-2021].

[2] IBM Corporation. Compiled versus interpreted languages. https://www.ibm.com/docs/en/zos-basic-skills?topic=zos-compiled-versus-interpreted-languages, 2010. [Online; accessed 03-Maj-2021].