analise_experimental

October 17, 2020

1 Análise experimental

Tanto o script shell e o arquivo gerador de entradas rândomicas podem ser encontrados na pasta "/experimental", juntamente com o jupyternotebook utilizado.

Algumas informações importantes: 1- Os tamanhos da entrada (N) varia de 1 em 1 até 256 e os valores das pedras não passam de 128 segundo a especificação do problema. 2- Forem gerados 10 testes para cada tamanho de entrada, a média e o desvio padrão desses valores foram usados para gerar os gráficos. 3- Estipulou-se um tempo limite de 600s(10 min) para a implementação brute-force do problema, o limite de tempo foi alcançado com o tamanho de entrada maior ou igual a N=10, não continuamos os testes para N maiores apartir do momento que o tempo limite foi atingido. 4- Foram usados arquivos de entrada CIN para realização dos experimentos, eles podem ser encontrados também na pasta "/experimental" e detalhes como o limite de tempo imposto podem ser conferidos.

2 Load dados

2.1 implementação PD:

```
In [1]: import statistics
        media_tempos_PD = dict()
        desvio_tempos_PD = dict()

        with open("temposPD.txt") as f:
            lines = f.readlines()

        value_list = list()
        value = 0
        i=0
        num_values = 0
        for line in lines:
            if(i%2==0):
                line = line.strip()
                pos = 0
                for char in line:
                    if char == ',':
                        break
```

```
                pos+=1
            line = line[0:pos] + '.' + line[pos+1:]
            line = float(line)
            value_list.append(line)
        else:
            value = line.strip()
            value = int(value)
            num_values += 1
        if(num_values == 10):
            media_tempos_PD[value] = statistics.mean(value_list)
            desvio_tempos_PD[value] = statistics.pstdev(value_list)
            value_list = list()
            num_values = 0
        i+=1;
```

2.2 implementação Brute-Force:

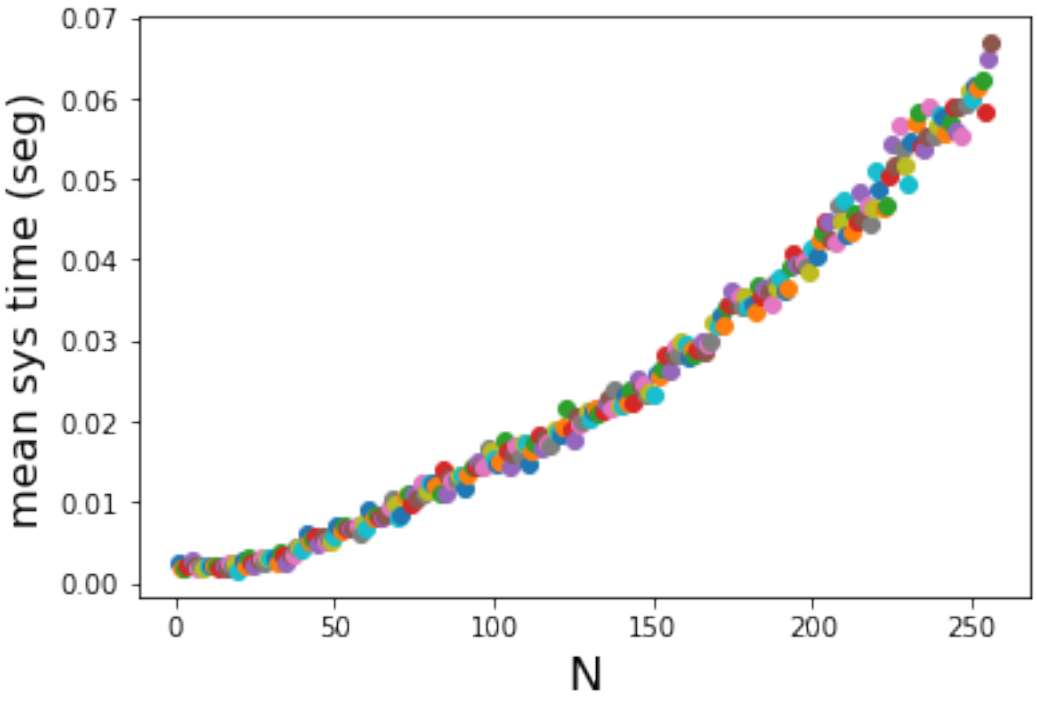
```
In [2]: import statistics
        media_tempos_BF = dict()
        desvio_tempos_BF = dict()

        with open("temposBF.txt") as f:
            lines = f.readlines()

        value_list = list()
        value = 0
        i=0
        num_values = 0
        for line in lines:
            if(len(line) > 30):
                break
            if(i%2==0):
                line = line.strip()
                pos = 0
                for char in line:
                    if char == ',':
                        break
                    pos+=1
                line = line[0:pos] + '.' + line[pos+1:]
                line = float(line)
                value_list.append(line)
            else:
                value = line.strip()
                value = int(value)
                num_values += 1
            if(num_values == 10):
                media_tempos_BF[value] = statistics.mean(value_list)
                desvio_tempos_BF[value] = statistics.pstdev(value_list)
```

```
            value_list = list()
            num_values = 0
        i+=1;
```

3 Gráfico do tempo de sys médio por tamanho da entrada

3.1 Implementação PD:

```
In [3]: import matplotlib.pyplot as plt

        for tempo in media_tempos_PD:
            x = tempo
            y = media_tempos_PD[tempo]
            plt.scatter(x,y)

        plt.xlabel('N', fontsize=18)
        plt.ylabel('mean sys time (seg)', fontsize=16)
        plt.show()
```

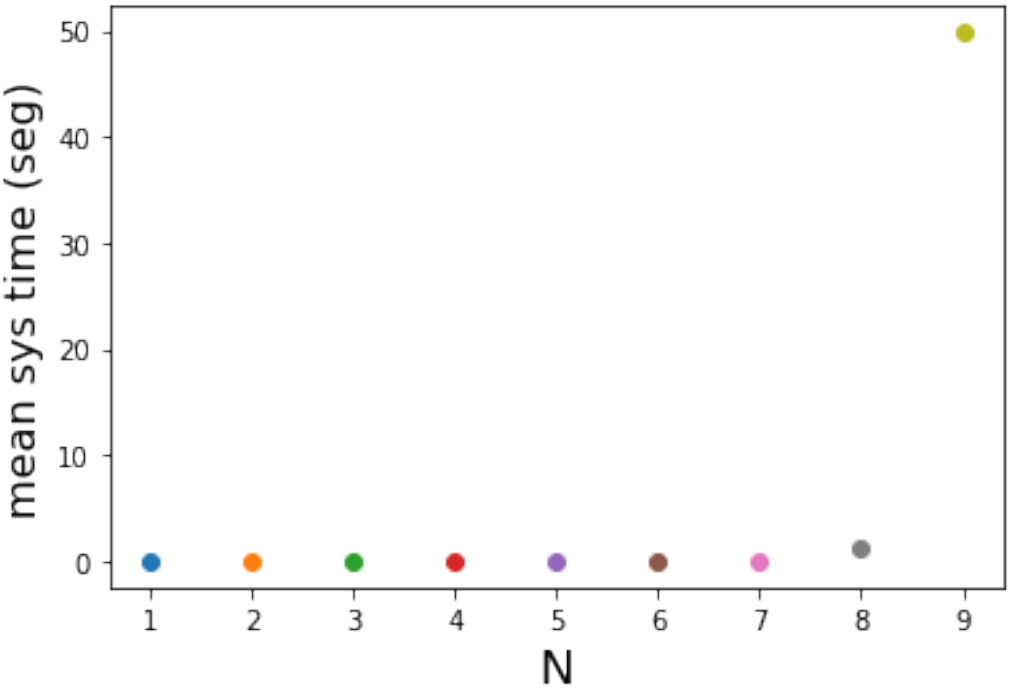


3.2 Implementação Brute-force:

3.2.1 O limite de tempo (600s) foi estourado para N >= 10

```
In [4]: import matplotlib.pyplot as plt
```

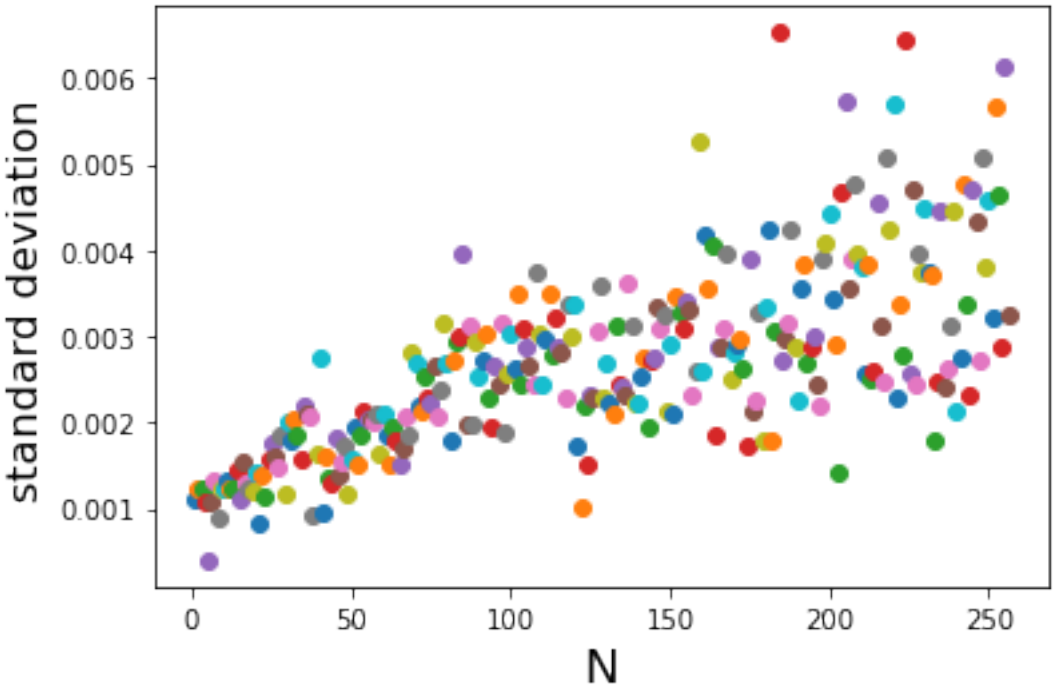
```
for tempo in media_tempos_BF:
    x = tempo
    y = media_tempos_BF[tempo]
    plt.scatter(x,y)

plt.xlabel('N', fontsize=18)
plt.ylabel('mean sys time (seg)', fontsize=16)
plt.show()
```



4 Gráfico do desvio padrão por tamanho da entrada

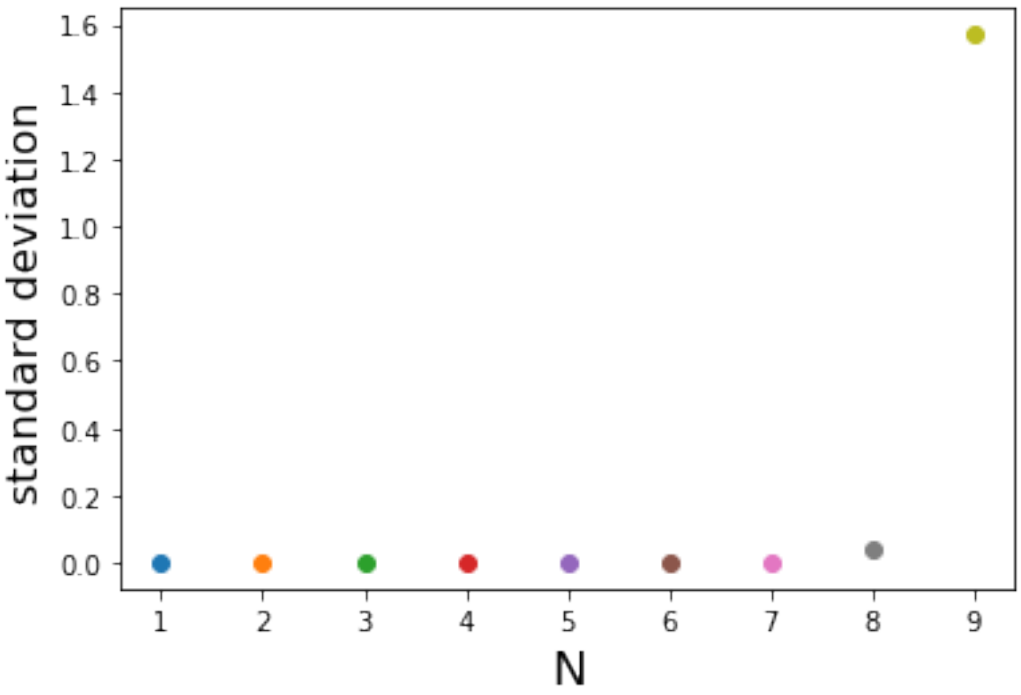
4.1 Implementação PD:

```
In [5]: for value in desvio_tempos_PD:
            x = value
            y = desvio_tempos_PD[value]
            plt.scatter(x,y)

        plt.xlabel('N', fontsize=18)
        plt.ylabel('standard deviation', fontsize=16)
        plt.show()
```



4.2 Implementação Brute-Force:

```
In [6]: for value in desvio_tempos_BF:
            x = value
            y = desvio_tempos_BF[value]
            plt.scatter(x,y)

        plt.xlabel('N', fontsize=18)
        plt.ylabel('standard deviation', fontsize=16)
        plt.show()
```


standard deviation
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
1
2
3
4
5
6
7
8
9
N