Relatório Intermediário

Maria Eduarda Bicalho

30 de abril de 2020

1 Descrição do Problema

O projeto Maximin Share tem como objetivo fazer a divisão mais justa possível de um número de objetos com diferentes valores entre um número diferente de pessoas. O problema pincipal se centra em como realizar essa divisão. Dessa forma, três diferentes técnicas - Herística, Busca Local e Busca Global - foram utilizadas para produzir três diferentes algoritmos para executar essa partição. Neste relatório essas três implementções serão analisadas com diferentes entradas para avaliar as alterções em suas saídas. Essas entradas possuírão diferentes tamanhos, alterando significamente. Primeiramente, em relação a quantidade de pessoas e depois a quantidade de objetos. Dentro de cada uma dessas análises, serão estudados o tempo, e a qualidade da solução em relação aos diferentes dimensões de entradas. A qualidade será analisada apartir do MMS (o valor da pessoa com o menor valor), ou seja quanto maior o MMS maior a qualidade da saída.

1.1 Máquina utilizada

Falar sobre máquina

2 Efeito número de pessoas

2.1 Entrada máxima de 5

Análise do impacto de uma entrada com diferentes números de pessoas nos 3 diferentes algoritmos implementados no projeto.

```
import subprocess
import time
import matplotlib.pyplot as plt

def roda(ex, in_f):
    with open(in_f) as f:
        start = time.perf_counter()
        proc = subprocess.run([ex], input=f.read(), text=True, capture_output=Tru
        end = time.perf_counter()
    return proc.stdout, end-start
arqs = [f"inp/in{i}.txt" for i in range(10)]
arqsg = [f"inp/in{i}.txt" for i in range(10)]
heuristica=[]
local=[]
```

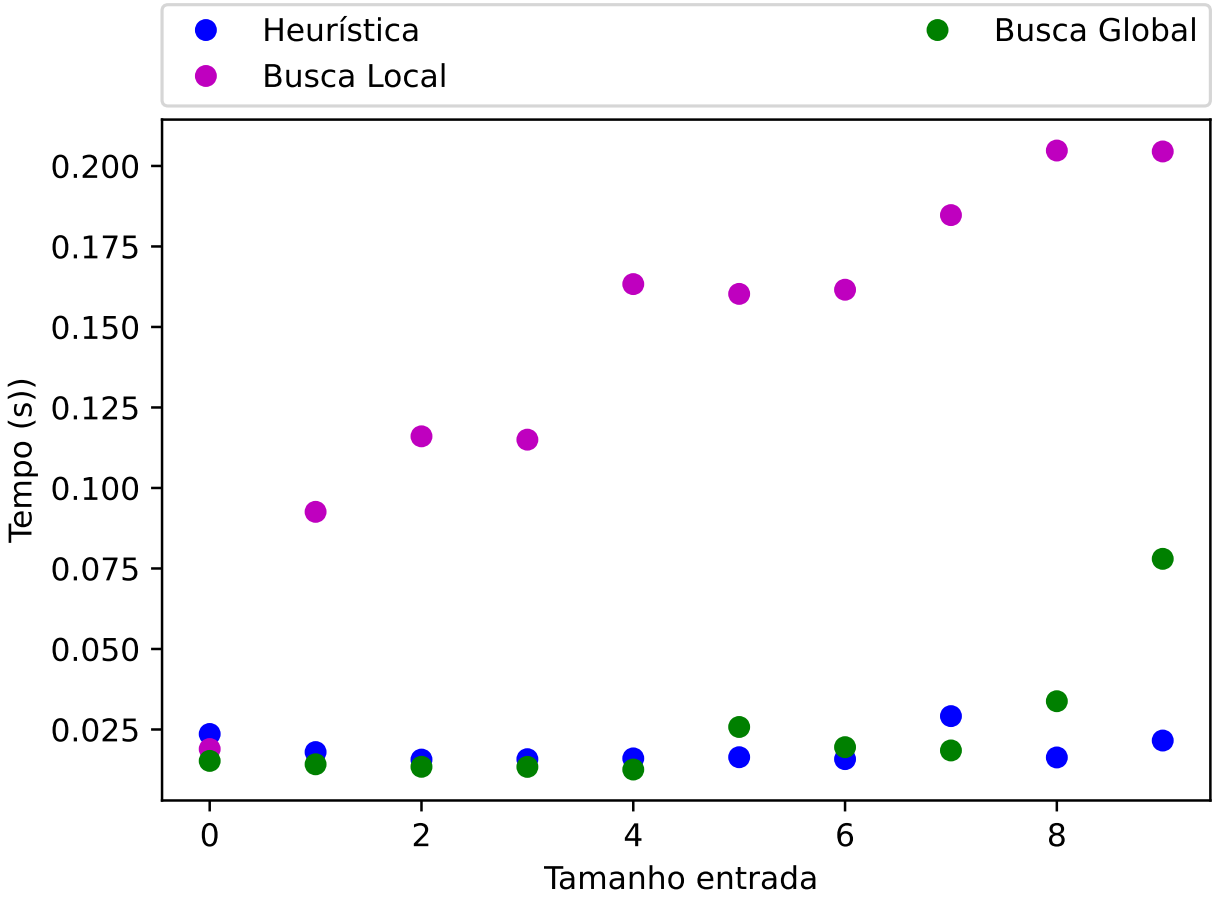
```
exaus=[]
outsh=[]
outsl=[]
outsg=[]
```

2.2 Tempo

Nessa seção o tempo será analisado nos diferentes algoritmos

2.2.1 Gráficos

```
import matplotlib.pyplot as plt

for arq in arqs:
    heuristica.append(roda('./heuristica', arq)[1])
    with open("out.txt",'r') as f:
        line = f.readlines()
        outsh.append(int(line[0]))
for arq in arqs:
    local.append(roda('./local', arq)[1])
    with open("out.txt",'r') as f:
        line = f.readlines()
        outsl.append(int(line[0]))
for arq in arqsg:
    exaus.append(roda('./global', arq)[1])
    with open("out.txt",'r') as f:
        line = f.readlines()
        outsg.append(int(line[0]))

h,= plt.plot(heuristica,'bo' ,label = "Heurística")
l,= plt.plot(local,'mo' ,label = 'Busca Local')
g, = plt.plot(exaus, 'go',label = 'Busca Global')

plt.xlabel("Tamanho entrada ")
plt.ylabel("Tempo (s))")
plt.legend(handles=[h, l, g],bbox_to_anchor=(0., 1.02, 1., .102), loc='lower left

<matplotlib.legend.Legend at 0x7f96d92d3c70>
```

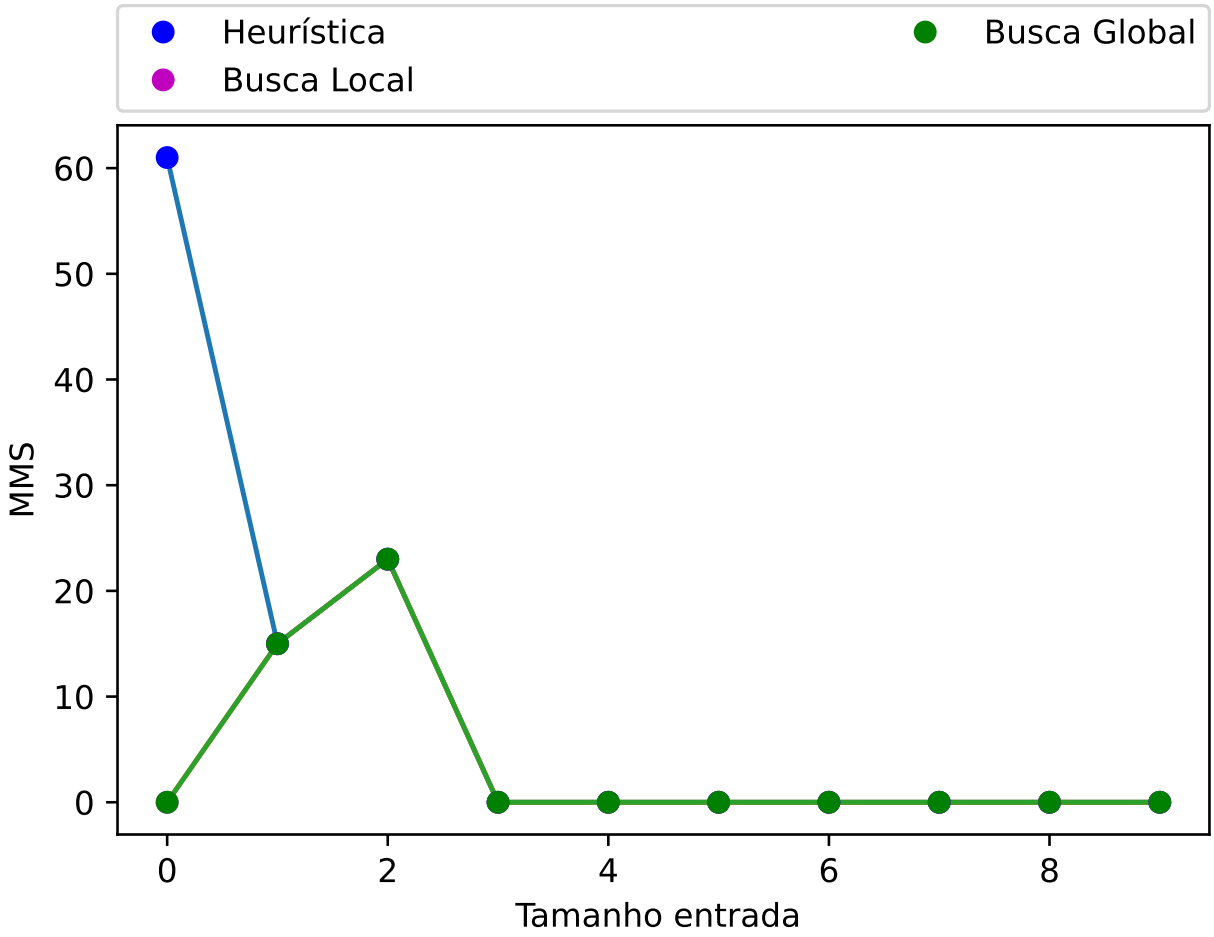


2.3 Qualidade da solução

2.3.1 Gráficos

```
import matplotlib.pyplot as plt

plt.plot(outsh)
plt.plot(outsl)
plt.plot(outsg)

h,= plt.plot(outsh,'bo' ,label = "Heurística")
l,= plt.plot(outsl, 'mo',label = 'Busca Local')
g, = plt.plot(outsg,'go' ,label = 'Busca Global')
plt.xlabel("Tamanho entrada ")
plt.ylabel("MMS")
plt.legend(handles=[h, l, g],bbox_to_anchor=(0., 1.02, 1., .102), loc='lower left
ncol=2, mode="expand", borderaxespad=0.)
```

```
<matplotlib.legend.Legend at 0x7f96d9159eb0>
```



3 Efeito número de objetos

```
arqsg = [f"ino/in{i}.txt" for i in range(10)]
arqs = [f"ino/in{i}.txt" for i in range(10)]

heuristica=[]
local=[]
exaus=[]
outsh=[]
outsl=[]
outsg=[]
```

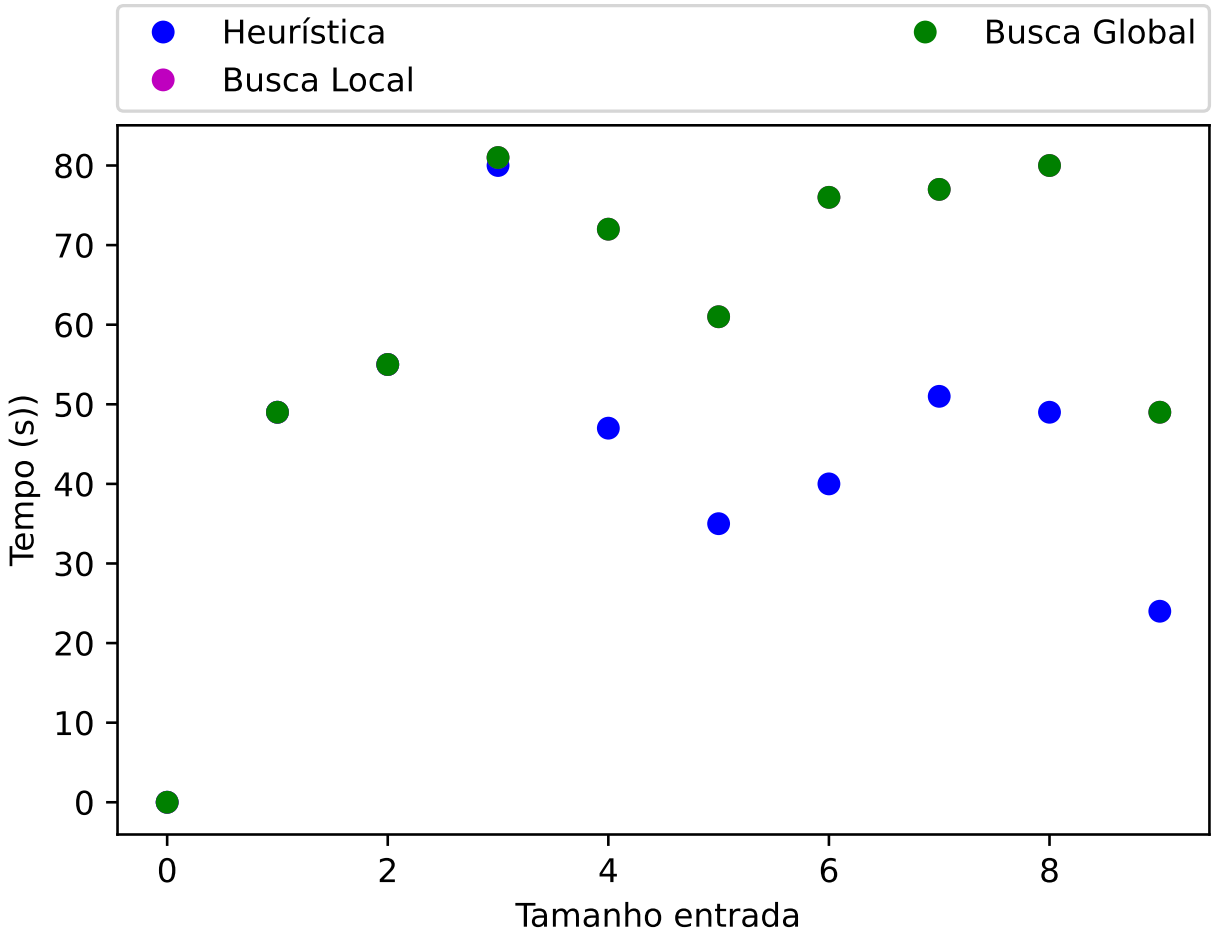
3.1 Tempo

3.1.1 Gráficos

```
import matplotlib.pyplot as plt

for arq in arqs:
    heuristica.append(roda('./heuristica', arq)[1])
    with open("out.txt",'r') as f:
        line = f.readlines()
```

```
        outsh.append(int(line[0]))
for arq in arqs:
    local.append(roda('./local', arq)[1])
    with open("out.txt",'r') as f:
        line = f.readlines()
        outsl.append(int(line[0]))
for arq in arqsg:
    exaus.append(roda('./global', arq)[1])
    with open("out.txt",'r') as f:
        line = f.readlines()
        outsg.append(int(line[0]))

h,= plt.plot(outsh,'bo' ,label = "Heurística")
l,= plt.plot(outsl, 'mo',label = 'Busca Local')
g, = plt.plot(outsg,'go' ,label = 'Busca Global')
plt.xlabel("Tamanho entrada ")
plt.ylabel("Tempo (s))")
plt.legend(handles=[h, l, g],bbox_to_anchor=(0., 1.02, 1., .102), loc='lower left
ncol=2, mode="expand", borderaxespad=0.)

print(exaus)
print(local)
print(exaus)
```

```
[0.01601716999994096, 0.01684198600014497, 0.013611418999971647,
0.013153500000044005, 0.020565202999932808, 0.02186728300057439,
0.0209101999998893, 0.02200540000012552, 0.21854538100069476,
0.9768302689999473]
[0.06425269699957425, 0.10808843899940257, 0.12203867300013371,
0.14539143499951024, 0.17520411899931787, 0.20961956399969495,
0.2360846429992307, 0.2548398440003439, 0.3070701659999031,
0.32931962800012116]
[0.01601716999994096, 0.01684198600014497, 0.013611418999971647,
0.013153500000044005, 0.020565202999932808, 0.02186728300057439,
0.0209101999998893, 0.02200540000012552, 0.21854538100069476,
0.9768302689999473]
```



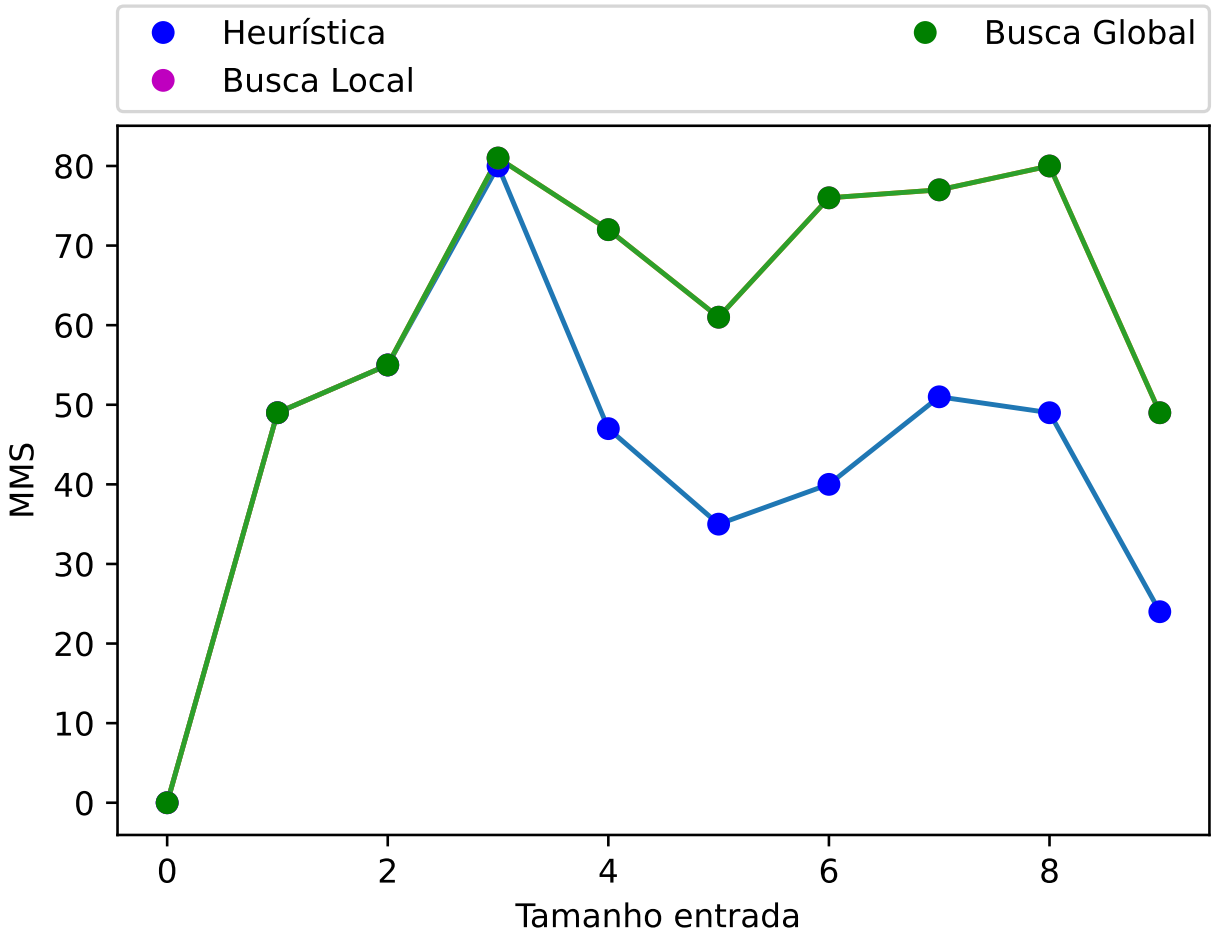
3.2 Qualidade da solução

3.2.1 Gráficos

```
import matplotlib.pyplot as plt

print(outsg[5])
print(outsl[5])
print(outsh[5])

plt.plot(outsh)
plt.plot(outsl)
plt.plot(outsg)

h,= plt.plot(outsh,'bo' ,label = "Heurística")
l,= plt.plot(outsl, 'mo',label = 'Busca Local')
g, = plt.plot(outsg,'go' ,label = 'Busca Global')
plt.xlabel("Tamanho entrada ")
plt.ylabel("MMS")
plt.legend(handles=[h, l, g],bbox_to_anchor=(0., 1.02, 1., .102), loc='lower left
ncol=2, mode="expand", borderaxespad=0.)

61
61
```

```
35
```

```
<matplotlib.legend.Legend at 0x7f96d900bd30>
```



4 Conclusão