Tempo de execução de algoritmos

profº Mauricio Conceição Mario

Exemplo 2

Complexidade de Algoritmos – conceitos → número de passos:

→ algoritmo: *Soma de duas matrizes de dimensão $n_x n$*:

Dadas as matrizes A_{ij} e B_{ij} de dimensões $n_x n$, determinar *a matriz* C_{ij} igual a soma das duas:

para $i := 1, ..., n$ ***fazer***
para $j := 1, ..., n$ ***fazer***
um passo → $C_{ij} := A_{ij} + B_{ij}$

Sempre que A e B forem matrizes de mesma dimensão $n_x n$, a variável à qual a expressão matemática avaliará o tempo de execução, ou seja, ***a variável independente****, é o parâmetro n. Cada passo do algoritmo de soma corresponderá à soma $A_{ij} + B_{ij}$. O número de passos será igual ao número de somas n. O algoritmo executa, então, n^2 passos.*

(Szwarcfiter & Markenzon, 2020)

Soma Matricial - exemplo

Exemplo 2

Dadas duas matrizes $\boldsymbol{A}$ e $\boldsymbol{B}$, sua soma será possível se tiverem a mesma dimensão, ou seja, $\boldsymbol{A}$ de m linhas e n colunas, e $\boldsymbol{B}$ com m linhas e n colunas.

$$\boldsymbol{A} = \begin{pmatrix} \mathbf{2} & \mathbf{-1} & \mathbf{3} \\ \mathbf{0} & \mathbf{4} & \mathbf{-6} \\ \mathbf{-6} & \mathbf{10} & \mathbf{-5} \end{pmatrix} \quad \boldsymbol{B} = \begin{pmatrix} \mathbf{4} & \mathbf{7} & \mathbf{-8} \\ \mathbf{9} & \mathbf{3} & \mathbf{5} \\ \mathbf{1} & \mathbf{-1} & \mathbf{2} \end{pmatrix}$$

$$\boldsymbol{A} + \boldsymbol{B} = \begin{pmatrix} \mathbf{2+4} & \mathbf{-1+7} & \mathbf{3+(-8)} \\ \mathbf{0+9} & \mathbf{4+3} & \mathbf{6+5} \\ \mathbf{-6+1} & \mathbf{10+(-1)} & \mathbf{-5+2} \end{pmatrix} = \begin{pmatrix} \mathbf{6} & \mathbf{6} & \mathbf{-5} \\ \mathbf{9} & \mathbf{7} & \mathbf{11} \\ \mathbf{-5} & \mathbf{9} & \mathbf{-3} \end{pmatrix}$$

(Zill & Cullen, 2009)

Exemplo 2

soma_matricial.py

```
import time
#from datetime import datetime
#matrizes de 3 x 3
A = [[2, -1, 3], [0, 4, 6], [-6, 10, -5]]
B = [[4, 7, -8], [9, 3, 5], [1, -1, 2]]

C = [[0, 0, 0], [0, 0, 0], [0, 0, 0]]
print("MATRIZ A:")
print ("A[0][0] = ", A[0][0])
print ("A[0][1] = ", A[0][1])
print ("A[0][2] = ", A[0][2])
print ("A[1][0] = ", A[1][0])
print ("A[1][1] = ", A[1][1])
print ("A[1][2] = ", A[1][2])
print ("A[2][0] = ", A[2][0])
print ("A[2][1] = ", A[2][1])
print ("A[2][2] = ", A[2][2])
print("MATRIZ B:")
print ("B[0][0] = ", B[0][0])
print ("B[0][1] = ", B[0][1])
print ("B[0][2] = ", B[0][2])
print ("B[1][0] = ", B[1][0])
print ("B[1][1] = ", B[1][1])
print ("B[1][2] = ", B[1][2])
print ("B[2][0] = ", B[2][0])
print ("B[2][1] = ", B[2][1])
print ("B[2][2] = ", B[2][2])

tempo_inicial = time.time()
for p in range(1):
  for i in range(0, 3):
    for j in range(0, 3):
       C[i][j] = A[i][j]+B[i][j]
tempo_final = time.time()
print("SOMA MATRICIAL:")
print("MATRIZ C = ", C)
print("tempo de execução = ", tempo_final - tempo_inicial)
```

Console 1/A

```
In [1]: runfile('C:/Users/cmari/.spyder-py3/soma_matricial.py', wdir='C:/Users/
cmari/.spyder-py3')
MATRIZ A:
A[0][0] =  2
A[0][1] =  -1
A[0][2] =  3
A[1][0] =  0
A[1][1] =  4
A[1][2] =  6
A[2][0] =  -6
A[2][1] =  10
A[2][2] =  -5
MATRIZ B:
B[0][0] =  4
B[0][1] =  7
B[0][2] =  -8
B[1][0] =  9
B[1][1] =  3
B[1][2] =  5
B[2][0] =  1
B[2][1] =  -1
B[2][2] =  2
SOMA MATRICIAL:
MATRIZ C =  [[6, 6, -5], [9, 7, 11], [-5, 9, -3]]
tempo de execução =  0.0
```

Exemplo 2

```python
import time
#from datetime import datetime
#matrizes de 3 x 3
A = [[2, -1, 3], [0, 4, 6], [-6, 10, -5]]
B = [[4, 7, -8], [9, 3, 5], [1, -1, 2]]

C = [[0, 0, 0], [0, 0, 0], [0, 0, 0]]
print("MATRIZ A:")
print ("A[0][0] = ", A[0][0])
print ("A[0][1] = ", A[0][1])
print ("A[0][2] = ", A[0][2])
print ("A[1][0] = ", A[1][0])
print ("A[1][1] = ", A[1][1])
print ("A[1][2] = ", A[1][2])
print ("A[2][0] = ", A[2][0])
print ("A[2][1] = ", A[2][1])
print ("A[2][2] = ", A[2][2])
print("MATRIZ B:")
print ("B[0][0] = ", B[0][0])
print ("B[0][1] = ", B[0][1])
print ("B[0][2] = ", B[0][2])
print ("B[1][0] = ", B[1][0])
print ("B[1][1] = ", B[1][1])
print ("B[1][2] = ", B[1][2])
print ("B[2][0] = ", B[2][0])
print ("B[2][1] = ", B[2][1])
print ("B[2][2] = ", B[2][2])

tempo_inicial = time.time()
for p in range(1000):
  for i in range(0, 3):
    for j in range(0, 3):
       C[i][j] = A[i][j]+B[i][j]
tempo_final = time.time()
print("SOMA MATRICIAL:")
print("MATRIZ C = ", C)
print("tempo de execução = ", tempo_final - tempo_inicial)
```

soma_matricial.py

Console 1/A

```
In [2]: runfile('C:/Users/cmari/.spyder-py3/soma_matricial.py', wdir='C:/Users/
cmari/.spyder-py3')
MATRIZ A:
A[0][0] =  2
A[0][1] =  -1
A[0][2] =  3
A[1][0] =  0
A[1][1] =  4
A[1][2] =  6
A[2][0] =  -6
A[2][1] =  10
A[2][2] =  -5
MATRIZ B:
B[0][0] =  4
B[0][1] =  7
B[0][2] =  -8
B[1][0] =  9
B[1][1] =  3
B[1][2] =  5
B[2][0] =  1
B[2][1] =  -1
B[2][2] =  2
SOMA MATRICIAL:
MATRIZ C =  [[6, 6, -5], [9, 7, 11], [-5, 9, -3]]
tempo de execução =  0.009018421173095703
```

Demonstrar a relação entre os tempos de execução de somas entre matrizes que tenham dimensões diferentes, exemplo:

Matriz A $_{(m \times n)}$

\+

Matriz B $_{(m \times n)}$

Tempo execução 1

Matriz C $_{(2*m \times 2*n)}$

\+

Matriz D $_{(2*m \times 2*n)}$

Tempo execução 2

Relação = Tempo execução 2 ÷ Tempo execução 1

Referências Bibliográficas

- Estruturas de Dados e Seus Algoritmos
  Jayme L. Szwarcfiter & Lilian Markenzon
  3ª edição – editora *gen* LTC – 2010 - 2020

- Matemática Avançada para Engenharia
  Dennis G. Zill & Michael R. Cullen
  Álgebra Linear e Cálculo Vetorial (2) – 3ª edição, editora Bookman - 2009