| | |
|---|---|
| Nombre del Alumno: | Valeria Jahzeel Castañon Hernández |
| Título de la tarea/práctica | Comparación de tiempos de ejecución con vectorización y de manera secuencial |
| Unidad de Aprendizaje | Cómputo paralelo |

- **Breve descripción del problema**
  Se requiere comparar los tiempos de ejecución de las operaciones de suma, resta y multiplicación de dos vectores de manera secuencial y utilizando la vectorización.

- **Conceptos básicos de vectorización**
  La vectorización es una técnica utilizada para optimizar el rendimiento al procesar datos en paralelo, esto se refiere a que realiza operaciones en múltiples elementos de datos al mismo tiempo mediante la aplicación de una sola instrucción.

  Tiene beneficios entre los cuales encontramos:
  - Mayor rendimiento:  Al procesar datos en bloques o vectores, se pueden aprovechar las capacidades de paralelismo del hardware moderno, lo que condice a un rendimiento significativamente mejorado.
  - Optimización de recursos: Permite utilizar de manera mas eficiente los recursos de la CPU o GPU al realizar múltiples operaciones en paralelo, lo que reduce el tiempo de ejecución y el consumo de energía
  - Simplicidad del código: A menudo se simplifica el código ya que se elimina la necesidad de bucles explícitos y operaciones de elemento por elemento, lo que puede hacer que el código se más fácil de entender y mantener.

- **Código**
  - Suma
    - Secuencial

```python
def secuential(n):
    # Se crean dos vectores de tamaño n
    a=np.random.randint(255, size=n)
    b=np.random.randint(255, size=n)
    # Se crea una lista para guardar el resultado
    suma = []
    t=time()
    for i in range(len(a)):
        suma.append(a[i]+b[i])
    return (time()-t)

# Se hacen 10 repeticiones
s=[]
for i in range(10):
    s.append(secuential(100000))

print(np.mean(np.array(s)),'s +/-',np.std(np.array(s)), 's')
```

- Vectorización

```python
def vectorization(n):
    a=np.random.randint(255, size=n)
    b=np.random.randint(255, size=n)
    # Se crea una lista para guardar el resultado
    t=time()
    c=a+b
    return (time()-t)

# Se hacen 10 repeticiones
s=[]
for i in range(10):
    s.append(vectorization(100000))

print(np.mean(np.array(s)),'=/-',np.std(np.array(s)))
```

- Resta
  - Secuencial

```
def secuential(n):
    # Se crean dos vectores de tamaño n
    a=np.random.randint(255, size=n)
    b=np.random.randint(255, size=n)
    # Se crea una lista para guardar el resultado
    resta = []
    t=time()
    for i in range(len(a)):
        resta.append(a[i]-b[i])

    return (time()-t)

# Se hacen 10 repeticiones
r=[]
for i in range(10):
    r.append(secuential(100000))

print(np.mean(np.array(r)),'r +/-',np.std(np.array(s)), 'r')
```

- Vectorización

```
def vectorization(n):
    a=np.random.randint(255, size=n)
    b=np.random.randint(255, size=n)
    # Se crea una lista para guardar el resultado
    t=time()
    c=a-b
    return (time()-t)

# Se hacen 10 repeticiones
r=[]
for i in range(10):
    s.append(vectorization(100000))

print(np.mean(np.array(r)),'=/-',np.std(np.array(r)))
```

- Multiplicación
  - Secuencial

```python
def secuential(n):
    # Se crean dos vectores de tamaño n
    a=np.random.randint(255, size=n)
    b=np.random.randint(255, size=n)
    # Se crea una lista para guardar el resultado
    multi = []
    t=time()
    for i in range(len(a)):
        multi.append(a[i]*b[i])

    return (time()-t)

# Se hacen 10 repeticiones
m=[]
for i in range(10):
    s.append(secuential(100000))

print(np.mean(np.array(m)),'m +/-',np.std(np.array(m)), 'm')
```

- Vectorización

```python
def vectorization(n):
    a=np.random.randint(255, size=n)
    b=np.random.randint(255, size=n)
    # Se crea una lista para guardar el resultado
    t=time()
    c=a*b
    return (time()-t)

# Se hacen 10 repeticiones
m=[]
for i in range(10):
    m.append(vectorization(100000))

print(np.mean(np.array(m)),'=/-',np.std(np.array(m)))
```

- **Resultados de los tiempos de ejecución**

| | Secuencial | Vectorización |
|---|---|---|
| Suma | 0.01524801254272461 s +/- 0.0007402159843058914 s | 9.968280792236329e-05 =/- 0.00029904842376708983 |
| Resta | 0.017385435104370118 s+/- 0.00029904842376708983 s | 0.00019993782043457032 =/- 0.000599813461303711 |
| Multiplicación | 0.029161548614501952 s +/- 0.005700881122848744 s | 0.0 s =/- 0.0 s |

- **Conclusión**

Como se vio en los resultados, una ejecución secuencial siempre obtuvo mayores tiempos de ejecución que una ejecución vectorizada, la diferencia si es algo grande, al menos en lo que se puede observar hay por lo menos una diferencia de $1x10^{-2}$ en los tiempos de ejecución.