josefa-tramon / **MCOC2021-P0**

forked from jaabell/MCOC2021-P0

main 1 branch 0 tags Code

This branch is 13 commits ahead, 1 commit behind jaabell:main. Contribute

15 commits

__pycache__
timing_inv_caso_1
timing_inv_caso_2
timing_inv_caso_3
README.md
matmul.py
rendimiento.txt
rendimiento0.txt
rendimiento1.txt
rendimiento2.txt
rendimiento3.txt
rendimiento4.txt
rendimiento5.txt
rendimiento6.txt
rendimiento7.txt
rendimiento8.txt
rendimiento9.txt
timing_matmul.py

README.md

MCOC2021-P0

P01:

Mi computador principal

- Marca/modelo: DESKTOP-RV7HP2N
- Tipo: Notebook
- Año adquisición: 2018
- Procesador:
  - Marca/Modelo: Intel Core i5-7200U
  - Velocidad Base: 2,71 GHz
  - Velocidad Máxima: 2,71 GHz
  - Numero de núcleos: 2
  - Humero de hilos: 2
  - Arquitectura: AMD64
  - Set de instrucciones:
- Tamaño de las cachés del procesador
  - L1: 128 KB
  - L2: 512 KB

About

No description, website, or topics provided.

Readme

Releases

No releases published

Packages

No packages published

Languages

Python 100.0%

    - L3: 3,0 Mb
- Memoria
    - Total: 12 GB
    - Tipo memoria: DDR4
    - Velocidad 1867 MHz
    - Numero de (SO)DIMM: 2
- Tarjeta Gráfica
    - Marca / Modelo: AMD Radeon (TM) R5 M330
    - Memoria dedicada: 2048 MB
    - Resolución: 1366x768
- Disco 1:
    - Marca: ST1000LM035
    - Tipo: Duro
    - Tamaño: 1TB
    - Particiones: 4
    - Sistema de archivos: EXT4
- Disco 2:
    - Marca: KINGSTON
    - Tipo: SSD
    - Tamaño: 223 GB
    - Particiones: 3
    - Sistema de archivos: EXT4
- Dirección MAC de la tarjeta wifi: 42-9F-38-C2-0D-** (por privacidad prefiero no dar completa mi dirección, por si las moscas jaja)
- Dirección IP (Interna, del router): 192.168.1.32
- Dirección IP (Externa, del ISP): 190.196.168.111
- Proveedor internet: GTD Manquehue.

P02:

¿Cómo difiere del gráfico del profesor/ayudante? Difieren en el tiempo transcurrido al inicio de la ejecución del programa, siendo más lento en mi computador que en el del profesor/ayudante.

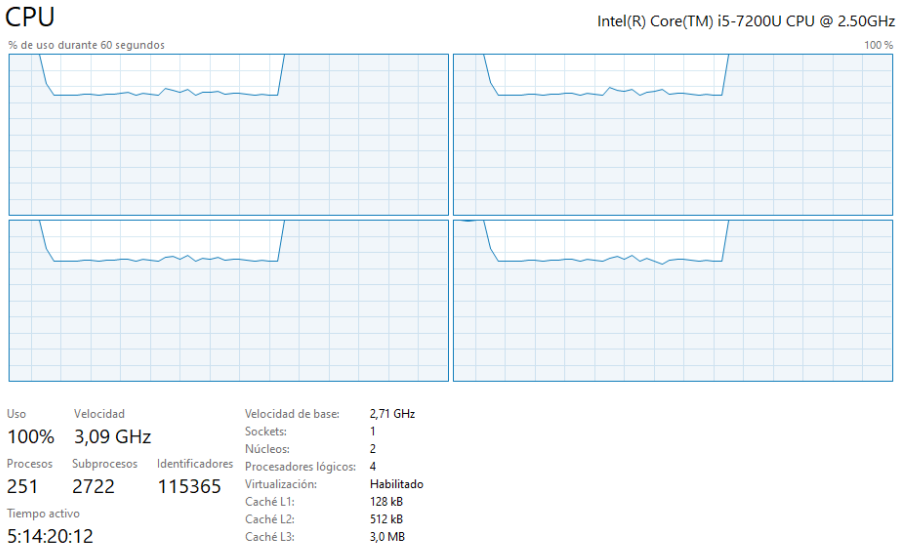
¿A qué se pueden deber las diferencias en cada corrida? Cada corrida es diferente ya que, aunque el equipo es el mismo, esta realizando diferentes tareas de trasfondo, las cuales segun la prioridad que tienen de realización, afectan el rendimiento del programa.

El gráfico de uso de memoria es lineal con el tamaño de matriz, pero el de tiempo transcurrido no lo es ¿porqué puede ser? Esto se puede deber a que el espacio que ocupa una matriz es constante, por lo que si crece, crece linealmente su memoria utilizada, en cambio el tiempo de ejecución de la ponderación de las matrices varía de forma exponencial, al crecer las matrices, la dificultad del problema no varía linealmente, ya que no es constante como el crecimiento de las matrices por separado.

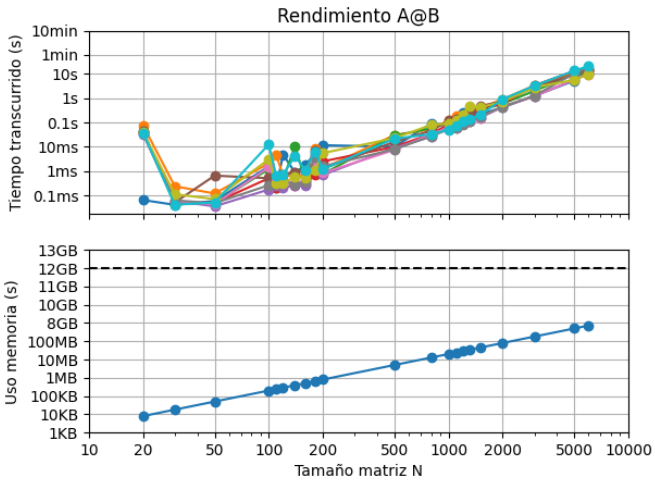
¿Qué versión de python está usando? Python 3.8

¿Qué versión de numpy está usando? Numpy 1.20.3

Durante la ejecución de su código ¿se utiliza más de un procesador? Muestre una imagen (screenshot) de su uso de procesador durante alguna corrida para confirmar. Si, se utilizan los 2 núcleos físicos, y 4 hilos, en un 100% cada uno como se puede ver en la fotografía:



El desempeño MATMUL en mi computador es el siguiente:



Se puede apreciar como los procesadores lógicos, que serían los 4 hilos mencionados previamente, estan mayoritariamente trabajando al 100% de su capacidad, cada uno representado en los graficos. La baja en los graficos y luego subida se debe a la variaci{on de matrices que se estan trabajando en el programa, siendo las de menor tamaño primero, y luego las de mayor tamaño, produciendo el uso completo del CPU.

P03:

¿Qué algoritmo de inversión cree que utiliza cada método (ver wiki)? Justifique claramente su respuesta.

- El método de numpy utiliza el algoritmo de Gauss - Jordan, en el cual se encuentra la inversa de una matriz a traves ecuaciones, donde el numero de ecuaciones sera el mismo al número de incognitas, las cuales pertenecerán a la matriz inversa de una matriz A conocida. Esto se encuentra al multiplicar la matriz A conocida con la matriz identidad I, y despejando la solución correspondiente.
- El método de Scipy utiliza el algoritmo de Gauss - Jordan, igual que el método de numpy.

  Para el caso 2: overwrite_a = True , se sobreescribira el resultado obtenido en la misma matriz A, dentro del sistema, lo cual es "más beneficioso" ya que se reutiliza espacio previamente ocupado por la matriz A original. a veces se cumple y otras no, todo dependerá de cuan especifico son los resultados esperados.

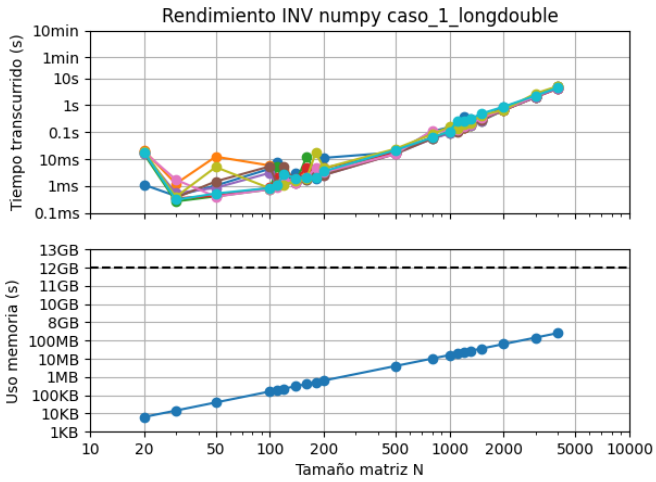
  Para el caso 3: overwrite_a = False , No se sobreescribira el resultado obtenido, sino que se ocupará nuevo espacio para escribir el resultado (matriz inversa), en la memoria del sistema, por lo que debería ser menos eficiente.

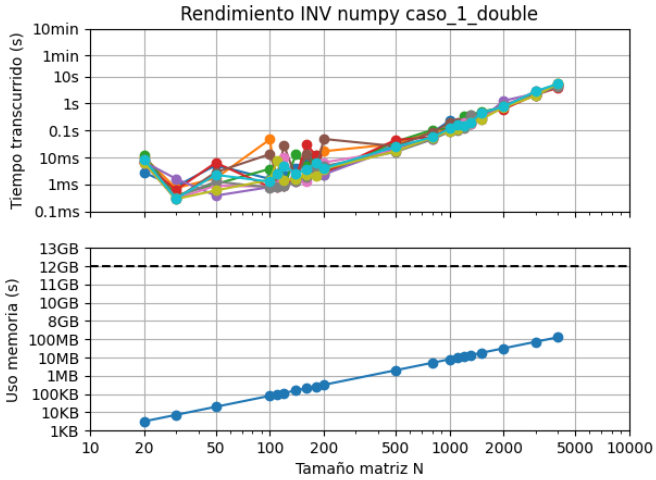
¿Como incide el paralelismo y la estructura de caché de su procesador en el desempeño en cada caso? Justifique su comentario en base al uso de procesadores y memoria observado durante las corridas.

- Cuando el equipo resuelve varios trabajos simultaneamente, y por lo mismo, genera cierta "jerarquización" sobre las tareas más importantes a las de menor relevancia, para así ejecutar lo que considera mas relevante terminar primero.

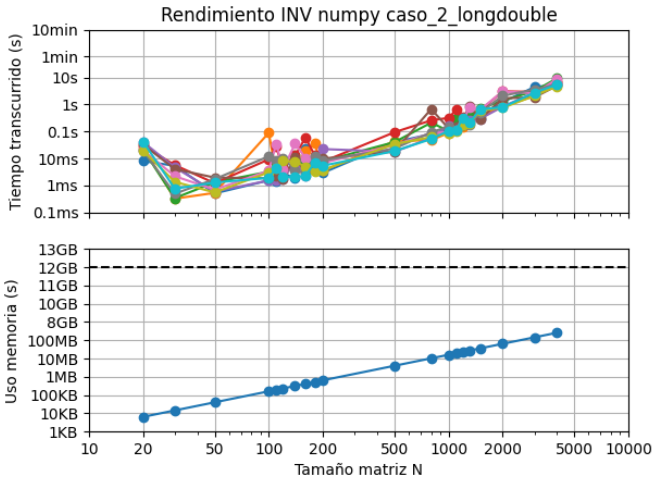
RESULTADOS OBTENIDOS:

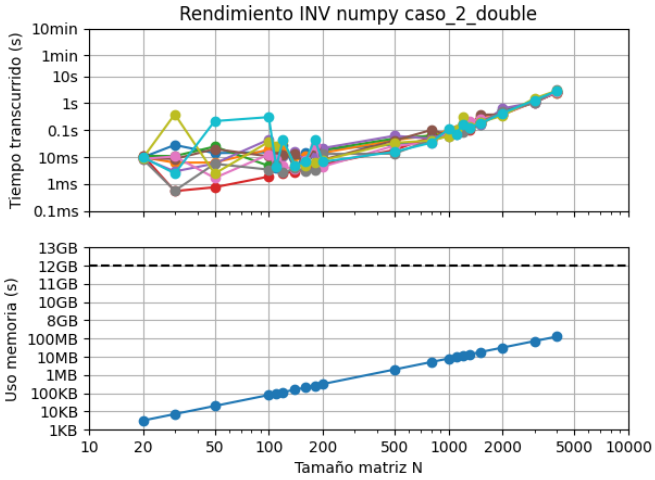
*CASO 1: Los casos 1 (numpy.inv) para half y single no fueron posibles ya que los resultados son poco exactos, y al utilizar este metodo, el programa primero ejecuta los calculos con d_type float 32, y luego aproxima más estos resultados, no realizandolo con menos decimales ya que no realiza calculos con tan poca especificidad.

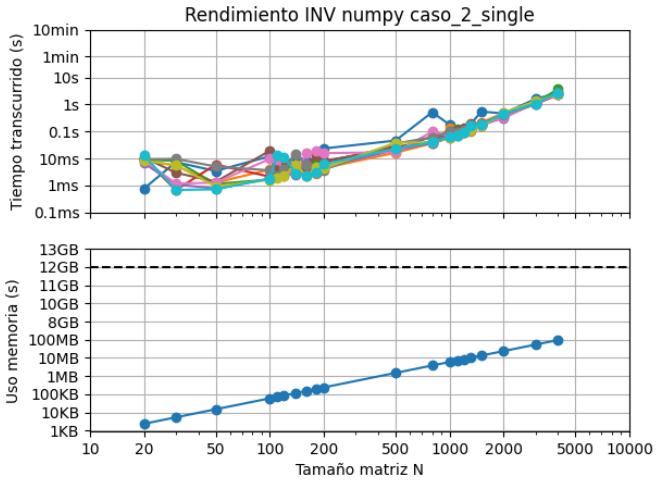


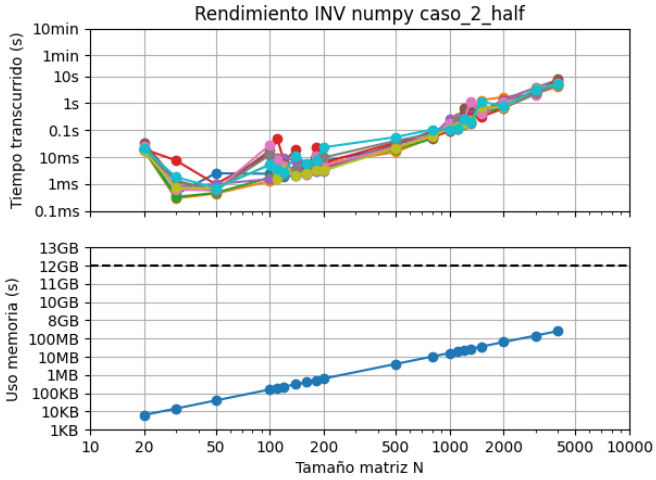


CASO 2:

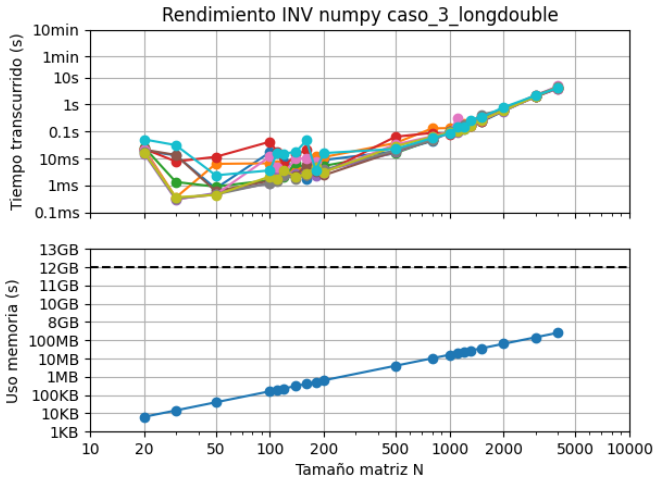

Rendimiento INV numpy caso_2_longdouble
Tiempo transcurrido (s)
10min
1min
10s
1s
0.1s
10ms
1ms
0.1ms
Uso memoria (s)
13GB
12GB
11GB
10GB
8GB
100MB
10MB
1MB
100KB
10KB
1KB
10
20
50
100
200
500
1000
2000
5000
10000
Tamaño matriz N

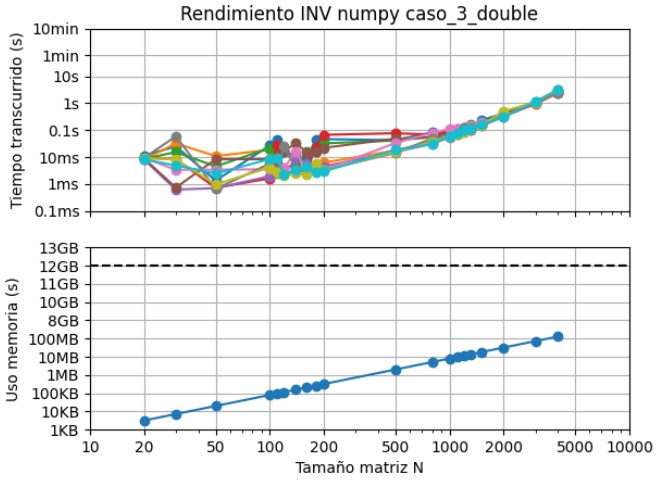

Rendimiento INV numpy caso_2_double
Tiempo transcurrido (s)
10min
1min
10s
1s
0.1s
10ms
1ms
0.1ms
Uso memoria (s)
13GB
12GB
11GB
10GB
8GB
100MB
10MB
1MB
100KB
10KB
1KB
10
20
50
100
200
500
1000
2000
5000
10000
Tamaño matriz N

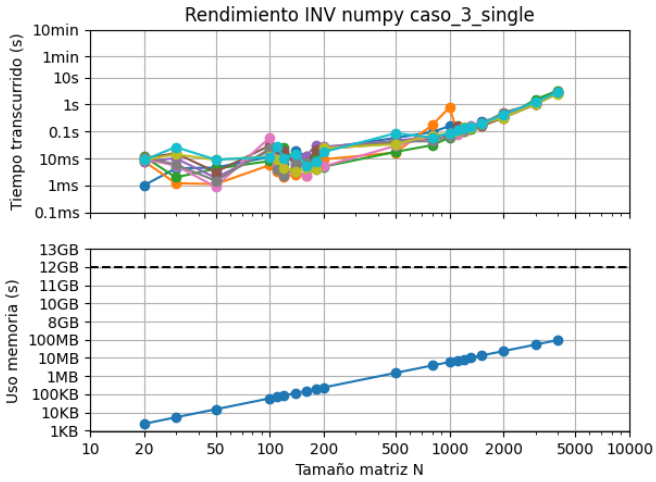

Rendimiento INV numpy caso_2_single
Tiempo transcurrido (s)
10min
1min
10s
1s
0.1s
10ms
1ms
0.1ms
Uso memoria (s)
13GB
12GB
11GB
10GB
8GB
100MB
10MB
1MB
100KB
10KB
1KB
10
20
50
100
200
500
1000
2000
5000
10000
Tamaño matriz N

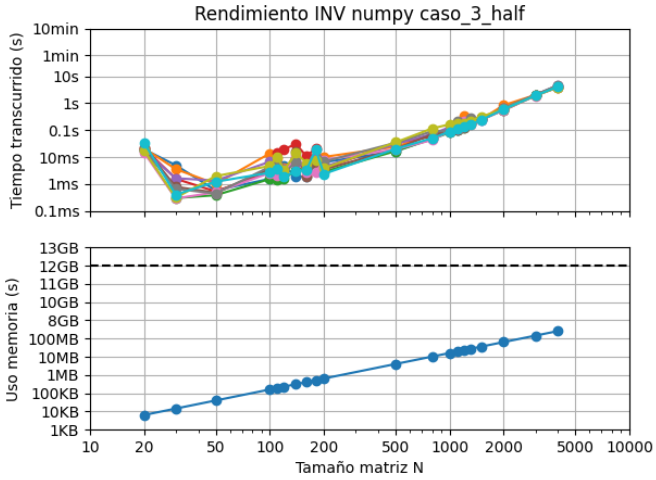

Rendimiento INV numpy caso_2_half
Tiempo transcurrido (s)
10min
1min
10s
1s
0.1s
10ms
1ms
0.1ms
Uso memoria (s)
13GB
12GB
11GB
10GB
8GB
100MB
10MB
1MB
100KB
10KB
1KB
10
20
50
100
200
500
1000
2000
5000
10000
Tamaño matriz N

CASO 3:


Rendimiento INV numpy caso_3_longdouble
Tiempo transcurrido (s)
10min
1min
10s
1s
0.1s
10ms
1ms
0.1ms
Uso memoria (s)
13GB
12GB
11GB
10GB
8GB
100MB
10MB
1MB
100KB
10KB
1KB
10
20
50
100
200
500
1000
2000
5000
10000
Tamaño matriz N

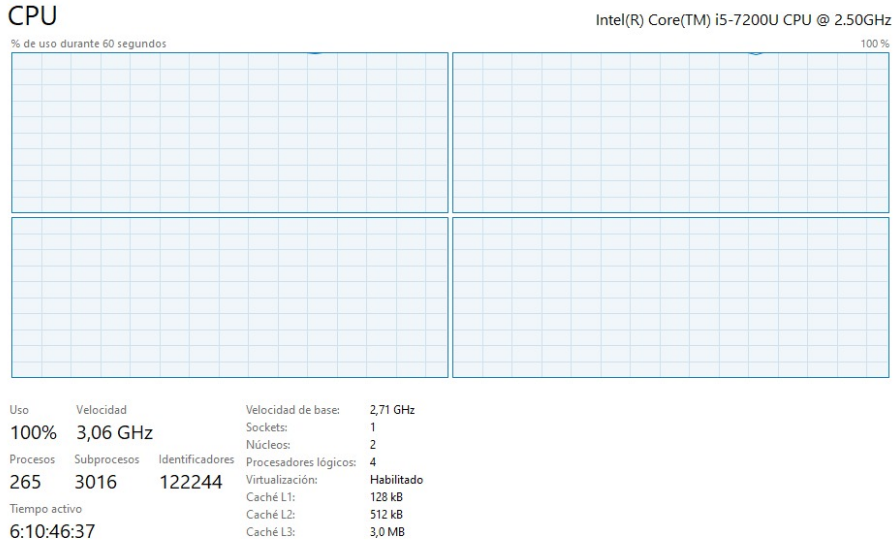

Rendimiento INV numpy caso_3_double
Tiempo transcurrido (s)
10min
1min
10s
1s
0.1s
10ms
1ms
0.1ms
Uso memoria (s)
13GB
12GB
11GB
10GB
8GB
100MB
10MB
1MB
100KB
10KB
1KB
10
20
50
100
200
500
1000
2000
5000
10000
Tamaño matriz N





COMPORTAMIENTO EQUIPO DURANTE PROCESOS:

Se puede ver el uso completo de la CPU al procesar los calculos solicitados.



El uso de memoria fue de un 50% aproximadamente.