

Evaluación de rendimiento

Sistemas Operativos



Pontificia Universidad
JAVERIANA
Colombia

Daniel Felipe Castro Moreno

Profesor:

John Corredor Franco

Dpto. de Ingeniería de Sistemas

Pontificia Universidad Javeriana

Bogotá D.C.

31 de octubre del 2024

Informe de laboratorio

La máquina virtual asignada por el profesor (10.43.103.151), la cual se llamará simplemente máquina 1, tiene las siguientes características:

- **Procesadores:** 4 núcleos, cada uno con un hilo y un socket.
- **Sistema operativo:** Rocky Linux - Fedora
- **Memoria caché:**
 - L1d: 4 instancias de 128 KiB cada una.
 - L1i: 4 instancias de 128 KiB cada una.
 - L2: 4 instancias de 4 MiB cada una.
 - L3: 4 instancias de 143 MiB cada una.
- **Memoria RAM:** 11673.0 MiB.
- **Arquitectura:** Soporte para 32 y 64 bits, con direccionamiento de 43 bits para la memoria física y 48 bits para la memoria virtual. El procesador utilizado es un Intel(R) Xeon(R) Gold 6240R CPU @ 2.40GHz, modelo 85.
- **Almacenamiento persistente:** 21 GB.

Por otro lado, la máquina virtual asignada por la universidad (10.43.103.153), la cual se llamará máquina 2, tiene las siguientes características:

- **Procesadores:** 4 núcleos, cada uno con un hilo y un socket.
- **Sistema operativo:** Ubuntu - Debian
- **Memoria caché:**
 - L1d: 4 instancias de 128 KiB cada una.
 - L1i: 4 instancias de 128 KiB cada una.
 - L2: 4 instancias de 1 MiB cada una.
 - L3: 4 instancias de 120 MiB cada una.
- **Memoria RAM:** 11914.2 MiB.
- **Arquitectura:** Soporte para 32 y 64 bits, con direccionamiento de 43 bits para la memoria física y 48 bits para la memoria virtual. El procesador utilizado es un Intel(R) Xeon(R) Gold 6240R CPU @ 2.40GHz, modelo 79.
- **Almacenamiento persistente:** 59 GB.

Realizando la comparación entre las máquinas virtuales, la principal diferencia radica en las memorias, ya que la máquina 1 tiene 4 veces más memoria L2 y 23 MiB adicionales en L3, mientras que la 2 es ligeramente superior en su memoria RAM y almacenamiento persistente (más del doble).

No obstante, tanto la arquitectura como procesadores son idénticos, salvo por el hecho de que la 1 tiene un modelo más reciente (85), y ambas configuraciones siguen la jerarquía de memoria, donde los niveles más cercanos al procesador son más rápidos, pero con menor capacidad de almacenamiento.

En resumidas cuentas, la máquina virtual 1, aunque con menor RAM y almacenamiento persistente, es mucho mejor en cuanto a la memoria caché y esto lo hace una opción más atractiva para programación de alto rendimiento.

A continuación, se presentan los comandos empleados para ambas máquinas:

```
[estudiante@ING-PDGE27 evaluacion]$ top
top - 10:14:04 up 105 days, 20:05, 2 users, load average: 0.05, 0.02,
Tasks: 253 total, 1 running, 252 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.2 us, 0.2 sy, 0.0 ni, 99.5 id, 0.0 wa, 0.1 hi, 0.1 si,
MiB Mem : 11673.0 total, 4642.6 free, 2749.6 used, 4631.1 buff/c
MiB Swap: 4100.0 total, 4100.0 free, 0.0 used, 8923.4 avail
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+
3779350	root	20	0	2622556	324308	102584	S	1.0	2.7	542:38.84
2473455	estudia+	20	0	10724	4352	3456	R	0.3	0.0	0:00.41
1	root	20	0	175024	18892	10940	S	0.0	0.2	1:45.91
2	root	20	0	0	0	0	S	0.0	0.0	0:03.42
3	root	0	-20	0	0	0	I	0.0	0.0	0:00.00
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00
8	root	0	-20	0	0	0	I	0.0	0.0	0:00.00
10	root	0	-20	0	0	0	I	0.0	0.0	0:00.00
12	root	20	0	0	0	0	I	0.0	0.0	0:00.00
13	root	20	0	0	0	0	I	0.0	0.0	0:00.00
14	root	20	0	0	0	0	I	0.0	0.0	0:00.00
15	root	20	0	0	0	0	S	0.0	0.0	0:06.49
16	root	20	0	0	0	0	S	0.0	0.0	8:37.89
17	root	20	0	0	0	0	I	0.0	0.0	21:01.17
18	root	rt	0	0	0	0	S	0.0	0.0	0:22.63
19	root	-51	0	0	0	0	S	0.0	0.0	0:00.00
21	root	20	0	0	0	0	S	0.0	0.0	0:00.00
22	root	20	0	0	0	0	S	0.0	0.0	0:00.00
23	root	-51	0	0	0	0	S	0.0	0.0	0:00.00
24	root	rt	0	0	0	0	S	0.0	0.0	0:20.33
25	root	20	0	0	0	0	S	0.0	0.0	0:05.29
27	root	0	-20	0	0	0	I	0.0	0.0	0:00.00
28	root	20	0	0	0	0	S	0.0	0.0	0:00.00
29	root	-51	0	0	0	0	S	0.0	0.0	0:00.00

```
estudiante@ing-gen64:~/Documents/evaluacion$ top
top - 10:14:19 up 71 days, 3:12, 2 users, load average: 0.04, 0.02,
Tasks: 317 total, 1 running, 316 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.2 us, 0.2 sy, 0.0 ni, 99.6 id, 0.0 wa, 0.0 hi, 0.0 si,
MiB Mem : 11914.2 total, 3028.2 free, 1607.2 used, 7611.4 buff/c
MiB Swap: 4096.0 total, 4096.0 free, 0.0 used, 10307.0 avail
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+
603144	estudia+	20	0	14520	5888	3712	R	0.7	0.0	0:01.03
14083	zabbix	20	0	31852	3784	2432	S	0.3	0.0	60:34.14
1712	gdm	20	0	4714372	229488	123664	S	0.3	1.9	212:06.17
602441	root	20	0	0	0	0	I	0.3	0.0	0:00.82
1	root	20	0	23476	14848	9600	S	0.0	0.1	5:41.26
2	root	20	0	0	0	0	S	0.0	0.0	0:04.09
3	root	20	0	0	0	0	S	0.0	0.0	0:00.00
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00
7	root	0	-20	0	0	0	I	0.0	0.0	0:00.00
11	root	20	0	0	0	0	I	0.0	0.0	0:48.39
12	root	0	-20	0	0	0	I	0.0	0.0	0:00.00
13	root	20	0	0	0	0	I	0.0	0.0	0:00.00
14	root	20	0	0	0	0	I	0.0	0.0	0:00.00
15	root	20	0	0	0	0	I	0.0	0.0	0:00.00
16	root	20	0	0	0	0	S	0.0	0.0	0:00.00
17	root	20	0	0	0	0	I	0.0	0.0	0:00.00
18	root	rt	0	0	0	0	S	0.0	0.0	0:00.47
19	root	20	0	0	0	0	I	0.0	0.0	5:28.19
20	root	rt	0	0	0	0	S	0.0	0.0	0:32.69
21	root	-51	0	0	0	0	S	0.0	0.0	0:00.00
22	root	20	0	0	0	0	S	0.0	0.0	0:00.00
23	root	-51	0	0	0	0	S	0.0	0.0	0:00.00
24	root	rt	0	0	0	0	S	0.0	0.0	0:30.42
25	root	20	0	0	0	0	S	0.0	0.0	0:13.26
27	root	20	0	0	0	0	S	0.0	0.0	0:00.00

Figura 1. top.

<p>Architecture: x86_64 CPU op-mode(s): 32-bit, 68-bit Address sizes: 43 bits physical, 48 bits virtual Byte Order: Little Endian CPU(s): 4 On-line CPU(s) list: 0-3 Vendor ID: GenuineIntel Model name: Intel(R) Xeon(R) Gold 6248R CPU @ 2.40GHz CPU family: 6 Model: 85 Thread(s) per core: 1 Core(s) per socket: 1 Socket(s): 4 Stepping: 7 BogoMIPS: 4788.74 Flags: fpu vme de pse tsc msr pae mce cx8 apic sep mtr r ppe mca cmov pat pse36 clflush mmx fxsr sse s se2 ss syscall nx pdpe1gb rdtscp lm constant-ts c arch.perfmon nopl topology tsc.reliable nops top.tsc cpuid tsc.invm_free pni pclmulqdq sse 3 fma cx16 pcid sse4.1 sse4.2 x2apic movbe popc nt tsc.deadline.timer aes xsave avx f16c rdand hypervisor lahf_lm avm 3dnowprefetch ssbd ibrs ibpb stibp ibrs_enhanced fsgsbase tsc_adjust b all avx2 smep bmi2 imvpclid avx512f avx512dq rds ad adx smap clflushopt clwb avx512cd avx512bw avx512vl xsaveopt xsavec xsavei xsavei optmpe nd_clear_flush_lid arch_capabilities</p> <p>Virtualization features: Hypervisor vendor: VMware Virtualization type: full Caches (sum of all): L1d: 128 KiB (4 instances) L1i: 128 KiB (4 instances) L2: 4 MiB (4 instances) L3: 143 MiB (4 instances)</p> <p>NUMA: NUMA node(s): 1 NUMA node0 CPU(s): 0-3</p> <p>Vulnerabilities: Gather data sampling: Unknown: Dependent on hypervisor status Itlb multihit: KVM: Mitigation: VMX unsupported L1tf: Not affected Mds: Not affected Meltdown: Not affected Mmio stale data: Vulnerable: Clear CPU buffers attempted, no mic rocode; SMT Host state unknown Mitigation: Enhanced IBRS Retbleed: Not affected Spec rstack overflow: Mitigation; Speculative Store Bypass disabled v ia prctl Spec store bypass: Mitigation; usercopy/swapgs barriers and __user pointer sanitization Spectre v1: Mitigation; usercopy/swapgs barriers and __user pointer sanitization Spectre v2: Mitigation; Enhanced / Automatic IBRS, IBPB con ditional, RSB filling, PRGB-eIBRS; SMT sequence</p>	<p>Architecture: x86_64 CPU op-mode(s): 32-bit, 68-bit Address sizes: 43 bits physical, 48 bits virtual Byte Order: Little Endian CPU(s): 6 On-line CPU(s) list: 0-3 Vendor ID: GenuineIntel Model name: Intel(R) Xeon(R) CPU E5-2650 v4 @ 2.20GHz CPU family: 6 Model: 79 Thread(s) per core: 1 Core(s) per socket: 1 Socket(s): 4 Stepping: 0 BogoMIPS: 4399.99 Flags: fpu vme de pse tsc msr pae mce cx8 apic sep mtr r ppe mca cmov pat pse36 clflush mmx fxsr sse sse2 ss syscall nx pdpe1gb rdtscp lm constant _tsc arch.perfmon nopl topology tsc.reliable nonstop tsc cpuid tsc.invm_free pni pclmulqdq ssse3 fma cx16 pcid sse4.1 sse4.2 x2apic movb e popcnt tsc.deadline.timer aes xsave avx f16c rdand hypervisor lahf_lm avm 3dnowprefetch c puid_fault pti ssbd ibrs ibpb stibp fsgsbase t sc_adjust bmi1 avx2 smep bmi2 imvpclid rdtsee d aor smap xsaveopt arat md_clear flush_lid arch. capabilities</p> <p>Virtualization features: Hypervisor vendor: VMware Virtualization type: full Caches (sum of all): L1d: 128 KiB (4 instances) L1i: 128 KiB (4 instances) L2: 1 MiB (4 instances) L3: 120 MiB (4 instances)</p> <p>NUMA: NUMA node(s): 1 NUMA node0 CPU(s): 0-3</p> <p>Vulnerabilities: Gather data sampling: Not affected Itlb multihit: KVM: Mitigation: VMX unsupported L1tf: Mitigation; PTE Inversion Mds: Mitigation; Clear CPU buffers; SMT Host state unknown Meltdown: Mitigation; PTI Mmio stale data: Mitigation; Clear CPU buffers; SMT Host state unknown Reg file data sampling: Not affected Retbleed: Mitigation; IBRS Spec rstack overflow: Not affected Spec store bypass: Mitigation; Speculative Store Bypass disabled via prctl Spectre v1: Mitigation; usercopy/swapgs barriers and __use r pointer sanitization Spectre v2: Mitigation; IBRS; IBPB conditional; STIBP disa</p>
--	--

Figura 2. lscpu.

Filesystem	Size	Used	Avail	Use%	Mounted on
devtmpfs	4.0M	0	4.0M	0%	/dev
tmpfs	5.7G	0	5.7G	0%	/dev/shm
tmpfs	2.3G	37M	2.3G	2%	/run
/dev/mapper/rl_plantillarocky9-root	24G	3.3G	21G	14%	/
/dev/sda2	960M	389M	572M	41%	/boot
/dev/sda1	1022M	7.1M	1015M	1%	/boot/efi
/dev/mapper/rl_plantillarocky9-var	15G	1.4G	14G	10%	/var
/dev/mapper/rl_plantillarocky9-home	15G	140M	15G	1%	/home
tmpfs	1.2G	4.0K	1.2G	1%	/run/user/100

estudiante@ing-gen64:~/Documents/evaluacion\$ df -h

Filesystem	Size	Used	Avail	Use%	Mounted on
tmpfs	1.2G	2.0M	1.2G	1%	/run
/dev/sda2	79G	16G	59G	22%	/
tmpfs	5.9G	0	5.9G	0%	/dev/shm
tmpfs	5.0M	0	5.0M	0%	/run/lock
tmpfs	5.9G	220K	5.9G	1%	/run/qemu
tmpfs	1.2G	100K	1.2G	1%	/run/user/120
tmpfs	1.2G	84K	1.2G	1%	/run/user/1000

estudiante@ing-gen64:~/Documents/evaluacion\$

Figura 3. df - h.

```

NAME="Rocky Linux"
VERSION="9.4 (Blue Onyx)"
ID="rocky"
ID_LIKE="rhel centos fedora"
VERSION_ID="9.4"
PLATFORM_ID="platform:el9"
PRETTY_NAME="Rocky Linux 9.4 (Blue Onyx)"
ANSI_COLOR="0;32"
LOGO="fedora-logo-icon"
CPE_NAME="cpe:/o:rocky:rocky:9:baseos"
HOME_URL="https://rockylinux.org/"
BUG_REPORT_URL="https://bugs.rockylinux.org/"
SUPPORT_END="2032-05-31"
ROCKY_SUPPORT_PRODUCT="Rocky-Linux-9"
ROCKY_SUPPORT_PRODUCT_VERSION="9.4"
REDHAT_SUPPORT_PRODUCT="Rocky Linux"
REDHAT_SUPPORT_PRODUCT_VERSION="9.4"

estudiante@ing-gen64:~/Documents/evaluacion$ cat /etc/os-release
estudiante@ing-gen64:~/Documents/evaluacion$ cat /etc/os-release
PRETTY_NAME="Ubuntu 24.04 LTS"
NAME="Ubuntu"
VERSION_ID="24.04"
VERSION="24.04 LTS (Noble Numbat)"
VERSION_CODENAME=noble
ID=ubuntu
ID_LIKE=debian
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
UBUNTU_CODENAME=noble
LOGO=ubuntu-logo
estudiante@ing-gen64:~/Documents/evaluacion$

```

Figura 4. `cat /etc/os-release`.

Actividad

1. Descomprimir y abrir carpeta del taller.
2. Copiar ficheros fuente del taller a las 2 máquinas virtuales.
3. Ejecutar ficheros en ambas máquinas virtuales para verificar su funcionamiento.
4. Tomar 4 muestras para cada ejecutable para 1h, 2h, 4h de tamaños 800, 1000 y 1800.
5. Realizar gráficos.

En las siguientes imágenes se presenta la copia de los ficheros empleando scp (Secure Copy Protocol) y la recepción y ejecución en ambas máquinas virtuales.

```

C:\Users\aulasmoviles\Downloads\TallerEvaluacion\Archivo>scp *.c estudiante@10.43.100.153:/home/estudiante/Documents/evaluacion
estudiante@10.43.100.153's password:
Fuente_Evaluacion.c          100% 2995   243.7KB/s   00:00
mm_clasico.c                 100% 2996   292.6KB/s   00:00
mm_transpuesta.c             100% 2999    1.4MB/s     00:00

```

Figura 5. `scp` desde el host para la máquina virtual 2.

```

C:\Users\aulasmoviles\Downloads\TallerEvaluacion\Archivo>scp *.c estudiante@10.43.103.151:/home/estudiante/CastroD/evaluacion
estudiante@10.43.103.151's password:
Fuente_Evaluacion.c          100% 2995   162.5KB/s   00:00
mm_clasico.c                 100% 2996    2.9KB/s     00:00
mm_transpuesta.c             100% 2999   292.9KB/s   00:00

```

Figura 6. `scp` desde el host para la máquina virtual 1.

```

[estudiante@ING-PDGE27 evaluacion]$ pwd
/home/estudiante/CastroD/evaluacion
[estudiante@ING-PDGE27 evaluacion]$ ls
Fuente_Evaluacion.c mm_clasico.c mm_transpuesta.c

estudiante@ing-gen64:~/Documents/evaluacion$ pwd
/home/estudiante/Documents/evaluacion
estudiante@ing-gen64:~/Documents/evaluacion$ gcc mm_clasico.c -o clasico
estudiante@ing-gen64:~/Documents/evaluacion$ ./clasico 100 1

```

Figura 7. Recepción y ejecución de los ficheros en ambas máquinas virtuales.

Respecto a este comando, se usa en una conexión SSH (Secure Shell) y permite la copia de archivos cifrados con el fin de proteger su confidencialidad e integridad. En cuanto a las pruebas realizadas, sus resultados se presentan en la hoja de cálculo adyacente y, para una mejor visualización, se anexan además las gráficas realizadas en Python con su respectivo análisis y el enlace del proyecto en Google Colaboratory.

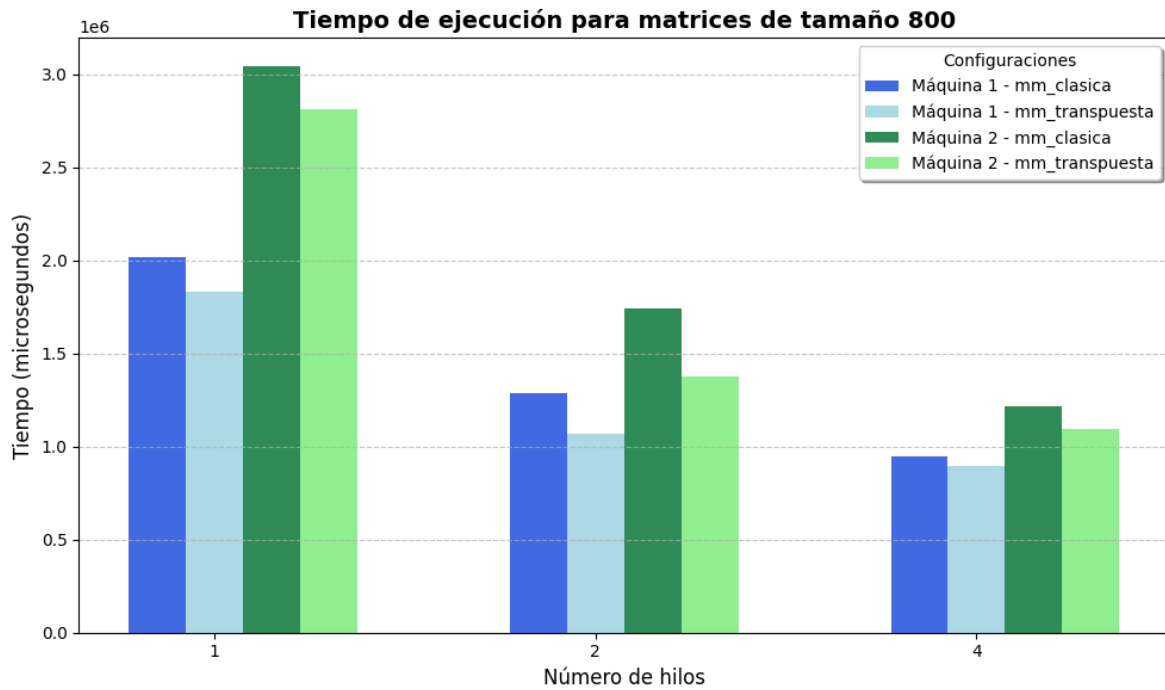


Figura 8. Tiempo de ejecución para 800.

Esta gráfica ilustra el tiempo de ejecución de ambos algoritmos de multiplicación de matrices (clásico y transpuesto) en matrices de tamaño 800x800. Se observa una reducción en el tiempo conforme se incrementa el número de hilos utilizados, destacando la eficiencia de la paralelización, particularmente en 2 hilos, donde el tiempo promedio con respecto a un solo hilo es de casi la mitad.

Asimismo, el algoritmo transpuesto ofrece un rendimiento superior al clásico en ambas máquinas, especialmente en la máquina 1, que posee una caché L2 y L3 de mayor capacidad. Esto indica que el rendimiento del algoritmo transpuesto se ve beneficiado en entornos con caché más amplios, posiblemente debido a una mejor localización de datos en memoria.

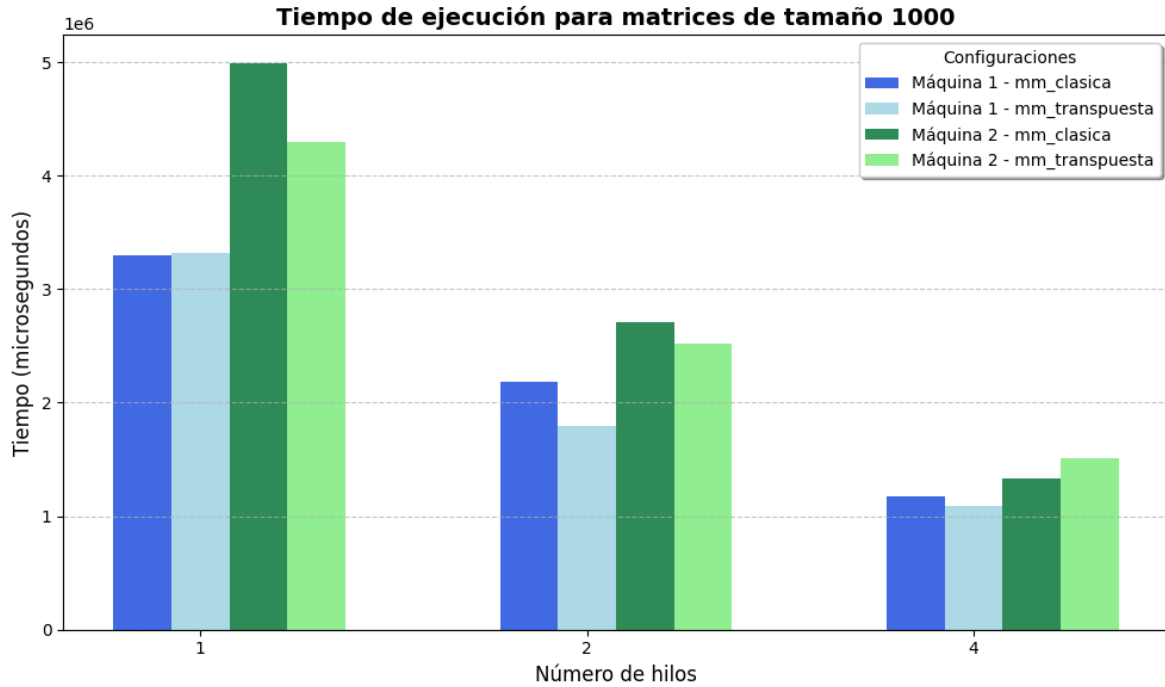


Figura 9. Tiempo de ejecución para 1000.

En matrices de tamaño 1000x1000, se mantiene la tendencia observada en la figura anterior: el tiempo de ejecución disminuye al aumentar el número de hilos. Nuevamente, el algoritmo transpuesto demuestra una ventaja notable, especialmente en la máquina 1.

La diferencia en tiempos de ejecución entre ambos algoritmos es más evidente a medida que el tamaño de la matriz aumenta, sugiriendo que el acceso optimizado a memoria en el algoritmo transpuesto ayuda a reducir los tiempos de procesamiento para tamaños de matrices más grandes, particularmente en sistemas con caché robusto.

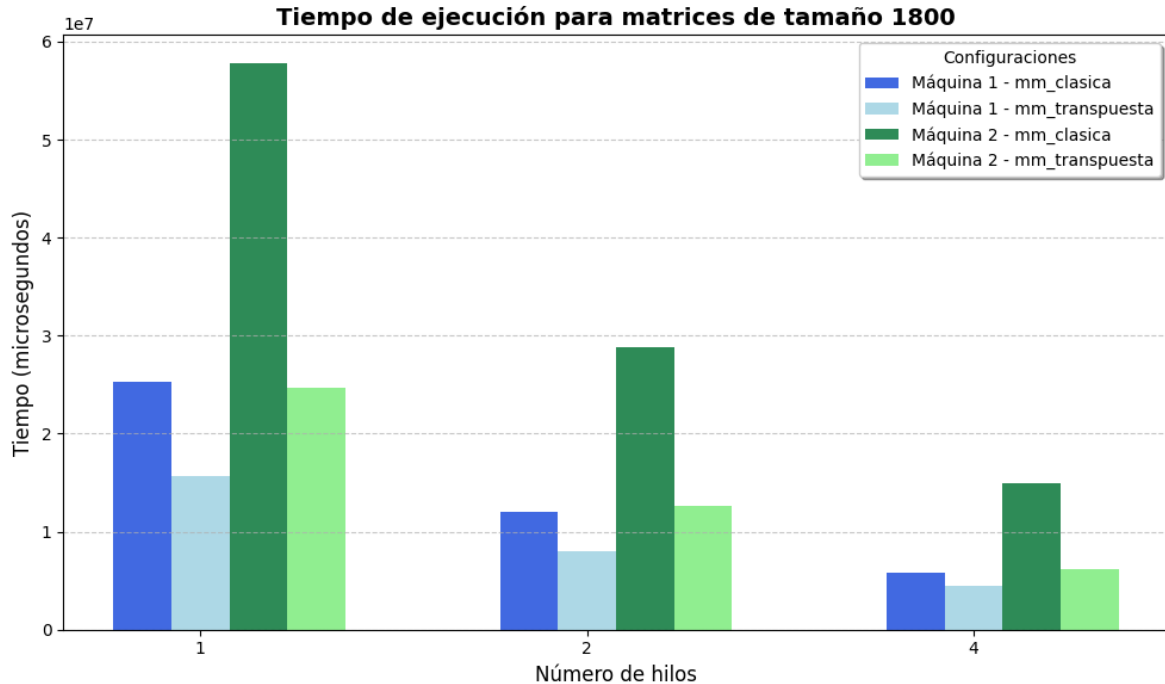


Figura 10. Tiempo de ejecución para 1800.

Esta gráfica muestra los resultados para matrices de 1800x1800, donde el rendimiento del algoritmo transpuesto supera significativamente al clásico. Aquí, el impacto de la cantidad de hilos es aún más pronunciado, evidenciando la escalabilidad de ambos algoritmos en matrices de gran tamaño.

La máquina 2 muestra un aumento considerable en el tiempo de ejecución comparado con la máquina 1 y se evidencia un pico muy alto en el algoritmo clásico ejecutado desde la máquina 2.

Con base a las gráficas anteriores, se realizó además la comparación de algoritmos. En la máquina 1, se confirma la superioridad del algoritmo transpuesto en todas las configuraciones de tamaño de matriz y número de hilos. El rendimiento del algoritmo transpuesto, que aprovecha un acceso de memoria más eficiente dado el principio de localización espacial, es particularmente evidente en esta máquina, dado que cuenta con una caché L2 y L3 significativamente mayor. La reducción en tiempo de ejecución a medida que se incrementan los hilos demuestra que esta máquina es capaz de gestionar cargas paralelas de forma eficiente.

En la máquina 2, el rendimiento del algoritmo transpuesto es igualmente superior al clásico en todas las configuraciones de prueba, aunque el impacto es menos pronunciado que en la máquina 1 debido a una caché de menor capacidad. A pesar de ello, el algoritmo transpuesto sigue siendo más eficiente al optimizar el acceso a memoria, lo cual contribuye a tiempos de ejecución reducidos. Sin embargo, al comparar con la máquina 1, se aprecia que esta máquina es menos eficiente en operaciones de multiplicación de matrices grandes debido a las limitaciones de caché.

Conclusiones

En general, se evidencia que el análisis realizado en la primera parte coincide con los resultados obtenidos en la ejecución de los programas, puesto que la mayor memoria de caché de la máquina virtual 1 le permitió tener tiempos de ejecución significativamente más cortos en el promedio de ejecuciones.

Por otro lado, el principio de localización temporal fue visible en la reducción de tiempo del algoritmo de multiplicación de matrices transpuestas en comparación con la ejecución más lenta por el método clásico.

Referencias

Repositorio en GitHub:

<https://github.com/Dani2044/Sistemas-Operativos/tree/main/Taller%206%20-%20Rendimiento>

Proyecto en Google Colaboratory:

<https://colab.research.google.com/drive/1fnH1fdPyImLAmi9Lt2BbhD6p-v3clazp?usp=sharing>