

Laboratory No. 01
Threads y rendimiento

Estudiantes.
Marianella Polo Peña
Daniel Eduardo Useche

Profesor.
Rodrigo Gualteros

Arquitectura de software
Universidad Escuela Colombiana de Ingeniería Julio Garavito
Bogotá, 03 de febrero de 2026

Table of Contents

Para Java:	3
Parte III. [Dado en mili-segundos]	3
Parte IV. [Dado en mili-segundos]	6
Para GO:	8
Parte III. [Dado en segundos]	8
Parte IV. [Dado en segundos]	8
Conclusión:	9

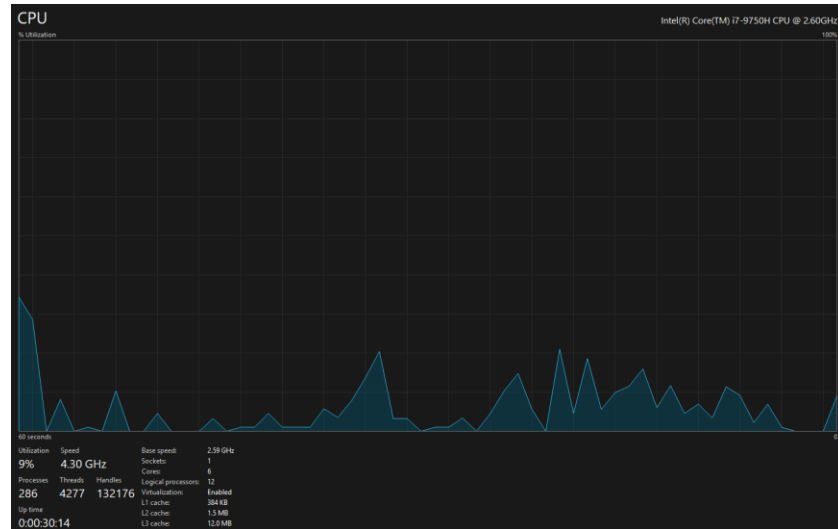
Para Java:

Parte III. [Dado en mili-segundos]

$N = 1$

```
"C:\Program Files\Java\jdk-25.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2025.3.2\lib\idea
Feb 03, 2026 2:49:46 PM edu.eci.arsw.spamkeywordsdatasource.HostBlackListsDataSourceFacade reportAsNotTrustworthy
INFO: HOST 202.24.34.55 Reported as NOT trustworthy
Feb 03, 2026 2:49:46 PM edu.eci.arsw.blacklistvalidator.HostBlackListsValidator checkHost
INFO: Checked Black Lists:9,001 of 10,000
N = 1
Tiempo (ms): 50
The host was found in the following blacklists:[100, 1500, 3500, 7000, 9000]

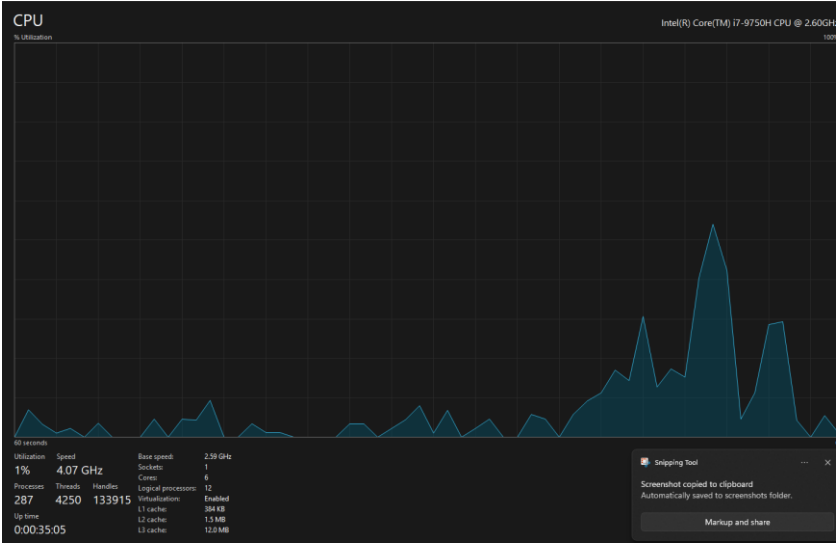
Process finished with exit code 0
```



$N = \text{Threads} = 12$

```
"C:\Program Files\Java\jdk-25.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ
Feb 03, 2026 2:55:26 PM edu.eci.arsw.spamkeywordsdatasource.HostBlackListsDataSourceFacade repor
INFO: HOST 202.24.34.55 Reported as NOT trustworthy
Feb 03, 2026 2:55:26 PM edu.eci.arsw.blacklistvalidator.HostBlackListsValidator checkHost
INFO: Checked Black Lists:9,500 of 10,000
N = 12
Tiempo (ms): 44
The host was found in the following blacklists:[100, 1500, 3500, 7000, 9000]

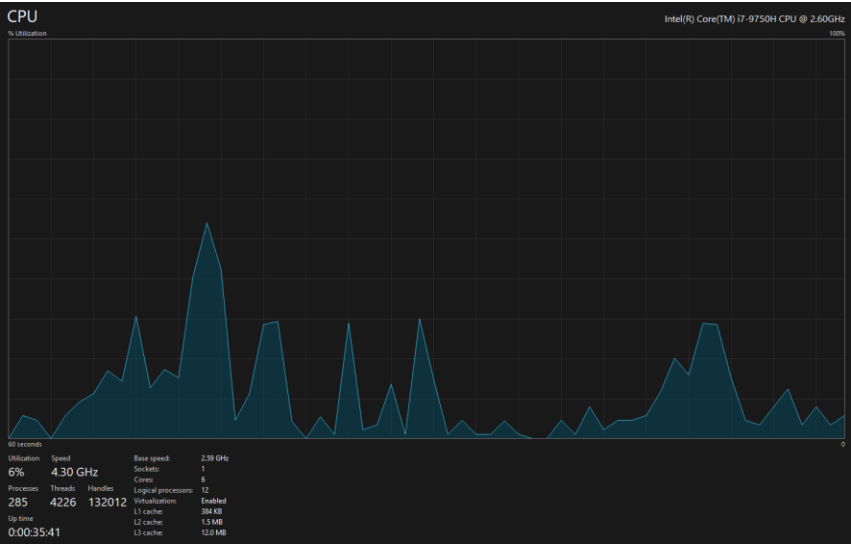
Process finished with exit code 0
```



$$N = 2 * \text{Threads} = 24$$

```
*C:\Program Files\Java\jdk-25.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ
Feb 03, 2026 2:56:03 PM edu.eci.arsw.spamkeywordsdatasource.HostBlacklistsDataSourceFacade rep
INFO: HOST 202.24.34.55 Reported as NOT trustworthy
Feb 03, 2026 2:56:03 PM edu.eci.arsw.blacklistvalidator.HostBlackListsValidator checkHost
INFO: Checked Black Lists:9,048 of 10,000
N = 24
Tiempo (ms): 44
The host was found in the following blacklists:[100, 1500, 3500, 7000, 9000]

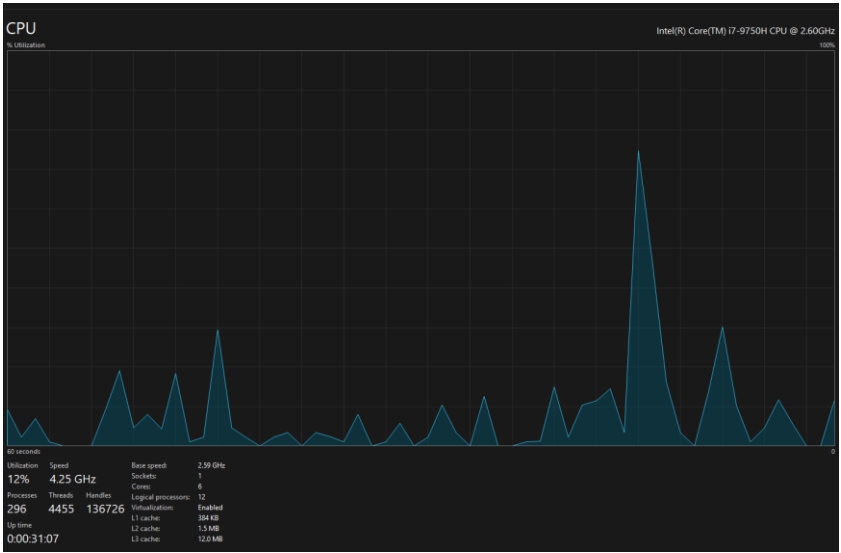
Process finished with exit code 0
```



N = 50

```
"C:\Program Files\Java\jdk-25.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2025.3.2\lib\idea_rt.jar=12738:
Feb 03, 2026 2:51:25 PM edu.eci.arsw.spamkeywordsdatasource.HostBlackListsDataSourceFacade reportAsNotTrustworthy
INFO: HOST 202.24.34.55 Reported as NOT trustworthy
Feb 03, 2026 2:51:25 PM edu.eci.arsw.blacklistvalidator.HostBlackListsValidator checkHost
INFO: Checked Black Lists:9,601 of 10,000
N = 50
Tiempo (ms): 63
The host was found in the following blackLists:[100, 1500, 3500, 7000, 9000]

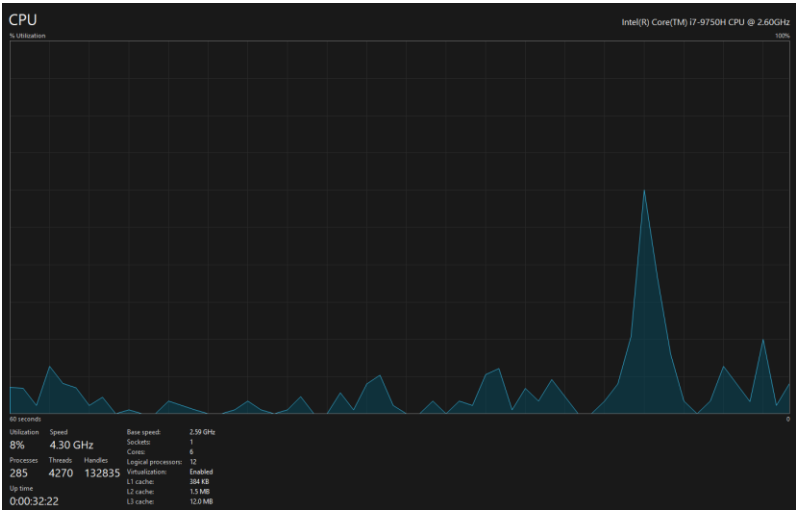
Process finished with exit code 0
```

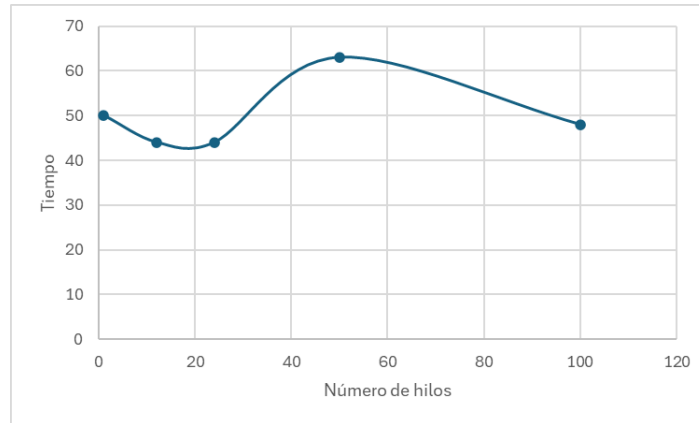


N = 100

```
"C:\Program Files\Java\jdk-25.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2025.3.2\lib\idea_rt.jar=12738:
Feb 03, 2026 2:52:42 PM edu.eci.arsw.spamkeywordsdatasource.HostBlackListsDataSourceFacade reportAsNotTrustworthy
INFO: HOST 202.24.34.55 Reported as NOT trustworthy
Feb 03, 2026 2:52:42 PM edu.eci.arsw.blacklistvalidator.HostBlackListsValidator checkHost
INFO: Checked Black Lists:9,101 of 10,000
N = 100
Tiempo (ms): 48
The host was found in the following blackLists:[100, 1500, 3500, 7000, 9000]

Process finished with exit code 0
```





- **Tendencia general:** El tiempo baja al pasar de 1 hilo (50) a 12–24 hilos (44), lo que indica que la búsqueda sí aprovecha el paralelismo en ese rango.
- **Mejor punto observado:** 12 y 24 hilos dan el menor tiempo (44). Eso sugiere que el trabajo está bien balanceado y el overhead de hilos aún es bajo.
- **Efecto de sobrecarga:** Con 50 hilos el tiempo sube a 63, señal de sobrecosto (creación de hilos, cambios de contexto, contención y sincronización).
- **Recuperación parcial:** Con 100 hilos baja a 48, pero no supera a 12–24. Esto confirma que más hilos no siempre implican mejor rendimiento.
- **Conclusión:** Para esta máquina y esta tarea, el rango óptimo está alrededor de 12–24 hilos. Por encima de eso, el overhead reduce el beneficio del paralelismo.

Parte IV. [Dado en mili-segundos]

N = 200

```
"C:\Program Files\Java\jdk-25.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2025.3.2\lib\id
Feb 03, 2026 3:11:07 PM edu.eci.arsw.spamkeywordsdatasource.HostBlackListsDataSourceFacade reportAsNotTrustworthy
INFO: HOST 202.24.34.55 Reported as NOT trustworthy
Feb 03, 2026 3:11:07 PM edu.eci.arsw.blacklistvalidator.HostBlackListsValidator checkHost
INFO: Checked Black Lists:8,851 of 10,000
N = 200
Tiempo (ms): 66
The host was found in the following blacklists:[100, 1500, 3500, 7000, 9000]

Process finished with exit code 0
```

N = 500

```

"C:\Program Files\Java\jdk-25.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2025.3.2\lib\i
Feb 03, 2026 3:11:38 PM edu.eci.arsw.spamkeywordsdatasource.HostBlacklistsDataSourceFacade reportAsNotTrustworthy
INFO: HOST 202.24.34.55 Reported as NOT trustworthy
Feb 03, 2026 3:11:38 PM edu.eci.arsw.blacklistvalidator.HostBlackListsValidator checkHost
INFO: Checked Black Lists:9,001 of 10,000
N = 500
Tiempo (ms): 68
The host was found in the following blacklists:[100, 1500, 3500, 7000, 9000]

Process finished with exit code 0

```

1. ¿Por qué no se logra el mejor desempeño con 500 hilos?

Aunque la ley de Amdahl indica que más hilos deberían mejorar el desempeño, en la práctica el tiempo **no baja** por el costo de crear y coordinar demasiados hilos. Con 500 hilos el tiempo fue 68 ms, ligeramente peor que con 200 hilos (66 ms). Esto muestra que el overhead por cambios de contexto, contención y sincronización puede superar el beneficio del paralelismo.

2. ¿Cómo se compara usar tantos hilos como núcleos vs el doble?

Usar el número de hilos cercano a los núcleos suele ser más eficiente: el sistema se mantiene balanceado sin crear demasiada sobrecarga. Al duplicar los hilos, se incrementa el overhead sin una mejora proporcional, por lo que el tiempo puede estancarse o incluso empeorar.

3. ¿Amdahl aplicaría mejor con 100 máquinas? ¿y con c hilos en $100/c$ máquinas?

Sí aplicaría mejor si cada máquina ejecuta 1 hilo, porque se reduce la contención local y los hilos no compiten por los mismos recursos. Sin embargo, aparece un nuevo costo: comunicación y coordinación entre máquinas. Si se usan c hilos en $100/c$ máquinas, mejora mientras el costo de comunicación sea bajo y el trabajo sea suficientemente paralelo. Si la fracción paralelizable P no es cercana a 1, el beneficio seguirá teniendo un límite, tal como lo predice Amdahl.

Para GO:

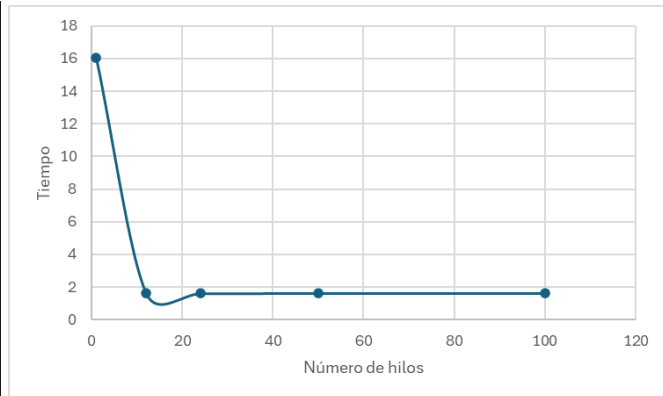
Parte III. [Dado en segundos]

```
Creado en: 368.2288ms

=== SECUENCIAL ===

=== CONCURRENTE ===
Workers 1 -> total 16.0269345s, avg 16026 µs
Workers 12 -> total 1.6459843s, avg 1645 µs
Workers 24 -> total 1.6190533s, avg 1619 µs
Workers 50 -> total 1.6367504s, avg 1636 µs
Workers 100 -> total 1.6308565s, avg 1630 µs

=== RESULTADOS ===
Repeticiones: 1000
Secuencial total: 15.9072159s
Secuencial promedio: 15907 µs
```



- **Ganancia clara con concurrencia:** El secuencial tardó ~15.91 s (15 907 µs promedio), mientras que con concurrencia baja a ~1.63–1.65 s ($\approx 1\,619$ – $1\,645$ µs). Esto muestra un speedup cercano a 10×, coherente con un problema muy paralelizable.
- **Mejor rango de hilos:** 24 hilos dan el menor tiempo (≈ 1.619 s). Con 12 hilos el tiempo es muy similar, lo que indica que el trabajo ya está bien distribuido en ese rango.
- **Sobre-paralelismo:** Al pasar a 50 y 100 hilos el tiempo no mejora (≈ 1.63 s). El overhead de crear más goroutines y la coordinación supera el beneficio adicional.
- **Conclusión:** Para esta máquina y carga, el punto óptimo está entre **12 y 24 hilos**, donde se logra el mejor equilibrio entre paralelismo y sobrecosto. Más hilos no dan mejora significativa.

Parte IV. [Dado en segundos]

```
Creado en: 394.1563ms

=== SECUENCIAL ===

=== CONCURRENTE ===
Workers 200 -> total 1.6430251s, avg 1643 µs
Workers 500 -> total 1.6409148s, avg 1640 µs

=== RESULTADOS ===
Repeticiones: 1000
Secuencial total: 15.9588557s
Secuencial promedio: 15958 µs
```

1. ¿Por qué no se logra el mejor desempeño con 500 hilos?

Porque el overhead de coordinación y scheduling crece. En tus pruebas, 200 hilos dio 1.643 s y 500 hilos 1.641 s: prácticamente igual. El beneficio extra es marginal y se pierde en el costo de administrar más hilos.

2. **¿Cómo se compara usar tantos hilos como núcleos vs el doble?**

Usar hilos cercanos a los núcleos suele ser más eficiente. Al duplicarlos, el tiempo tiende a estancarse porque los núcleos ya están ocupados y el sistema empieza a gastar más en cambios de contexto que en trabajo útil.

3. **¿Amdahl aplicaría mejor con 100 máquinas? ¿y con c hilos en $100/c$ máquinas?**

Sí, con 100 máquinas de 1 hilo se reduce contención local, pero aparece el costo de comunicación y coordinación. Con c hilos en $100/c$ máquinas puede mejorar si P es alto y la comunicación es baja, pero el límite de Amdahl sigue existiendo: la parte no paralelizable impone un techo a la aceleración.

Conclusión:

Go mostró menor overhead para manejar muchos hilos ligeros (goroutines), por eso escala mejor y se estabiliza antes; Java mejora hasta cierto punto, pero el costo de crear y coordinar muchos threads reales limita la ganancia. En ambos, el rendimiento óptimo no está en “más hilos”, sino cerca del número de núcleos y con carga balanceada.