# Informe Laboratorio: Escalabilidad Paralela en CPU y GPU
# INFO128 - Arquitectura de Computadores

Acádemico: Cristóbal Navarro
Alumnos: Felipe Córdova, Sebastián Montecinos, Diego Vidal
Instituto de Informática, Universidad Austral de Chile

Julio 9, 2023

## 1. Introducción

Luego de conectarnos al servidor con el nombre de equipo *losxd*, ejecutamos el código proporcionado *main.cu*, y tenemos como objetivo comparar la escalabilidad de rendimiento entre CPU y GPU utilizando el algoritmo SAXPY, el cual es una abreviatura de "Single-precision A·X Plus Y". SAXPY se refiere a una operación matemática comúnmente utilizada en el ámbito del procesamiento numérico y la programación en paralelo. SAXPY realiza la operación $Z[i] = a * X[i] + Y[i]$ para cada elemento en un arreglo, en donde se tiene que:

- $a$: Escalar de precisión simple (un solo número en formato de punto flotante).
- $X$ e $Y$: Arreglos en formato de punto flotante de precisión simple.
- $Z$: Arreglo de salida donde se almacenan los resultados de la operación.
- $i$: Posición respetiva en cada celda del arreglo $\forall i \in \{1..k\}$, siendo $k$ el tamaño de estos.

En este laboratorio, medimos los tiempos de ejecución de SAXPY en modo CPU, utilizando diferentes números de threads, y en modo GPU, encontrando un tamaño eficiente de bloque. Además, realizamos gráficos para visualizar los tiempos de ejecución y el speedup relativo entre CPU y GPU.

## 2. Especificaciones

El servidor está equipado con un potente procesador Intel Core i7 6950X, que cuenta con 10 hilos de ejecución. Además, cuenta con una tarjeta gráfica NVIDIA GeForce RTX 3090 Ti, que ofrece un total de 84 SMs (conteo de multiprocesadores). Cabe mencionar que para realizar los experimentos mantuvimos constante el valor de $n$ (tamaño del problema) en un valor de $100 \cdot 2^{20}$ (a menos que se diga lo contrario).

## 3. SAXPY modo CPU y GPU

Para constuir el gráfico de la Figura 1 ejecutamos en reiteradas veces el código, tanto en modo CPU, como en modo GPU. Para la curva de color azul, utilizamos el modo CPU, en el cual fuimos aumentando uno a uno el numero de threads utilizados. En cambio para la curva de color naranjo, utilizamos el modo GPU, en este contexto fuimos aumentando el valor del tamaño de bloque ($BS$) en potencias de 2, tal como dice las instrucciones.

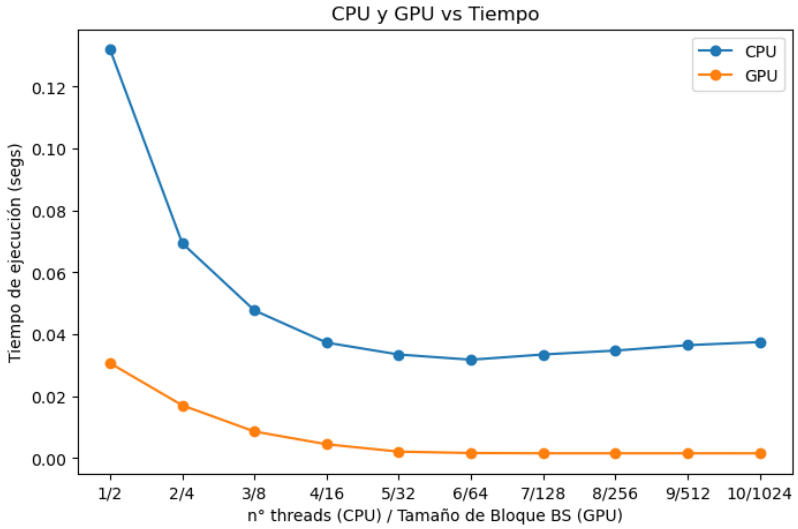


Figura 1: SAXPY CPU y GPU

El gráfico demuestra que al incrementar el número de threads en el modo CPU o aumentar el tamaño del bloque en el modo GPU, se logra una notable aceleración en la ejecución del proceso SAXPY. Sin embargo, en ambos casos se alcanza una asíntota, lo que significa que aumentar aún más el valor en el eje X no resultará en una mayor velocidad. Ademas, se aprecia claramente que el modo GPU es mucho mas rápido en términos de tiempo.

Nuestros resultados revelan que el tamaño de bloque más eficiente ($BS$) es de 1024, ya que al aumentar su valor, se observa una disminución en el tiempo de ejecución. Para respaldar esta conclusión, se presenta el gráfico en la Figura 2.

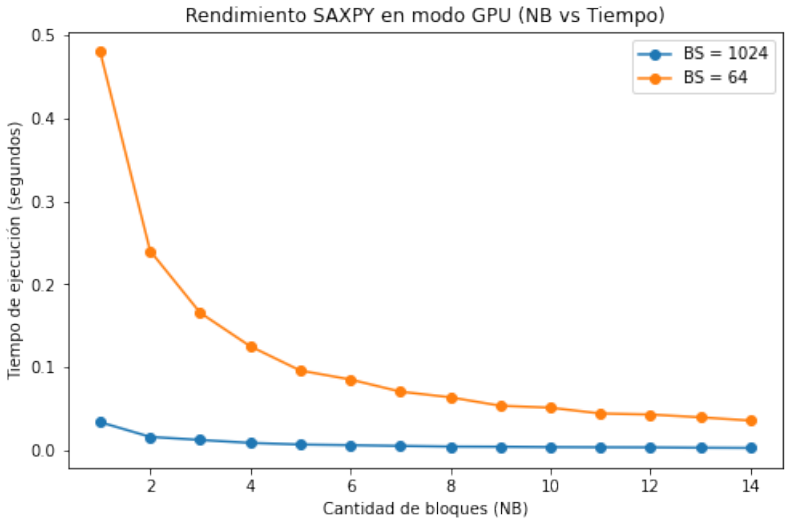


Figura 2: SAXPY GPU MODE, BS=1024 $v/s$ BS=64

La Figura 2 ha confirmado nuestra elección, lo cual nos brinda la confianza para continuar realizando experimentos con un tamaño de bloque de 1024 ($BS = 1024$).

A continuación se nos pedía hacer otro gráfico explorando el rango de 1 a numSMs*32 bloques ($NB$). La GPU en la cual se realizaron los experimentos tenía 84 SMs, por lo tanto nuestro rango de experimentación sería $[1, 2688]$. A continuación presentamos el gráfico resultante, el cual finalmente desechamos y creamos uno nuevo debido a que no era posible apreciar la curva deseada.

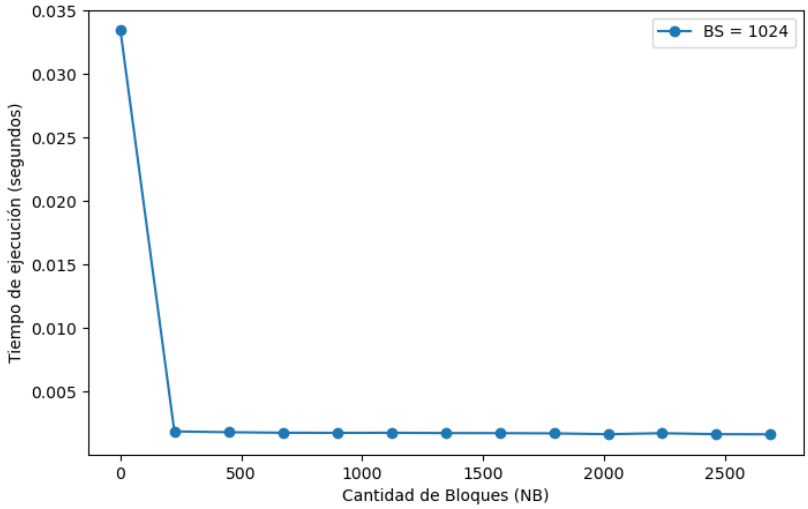


Figura 3: SAXPY GPU MODE, NB $v/s$ t

Para poder apreciar la curva que en verdad necesitabamos ajustamos el rango a $[1, 14]$.

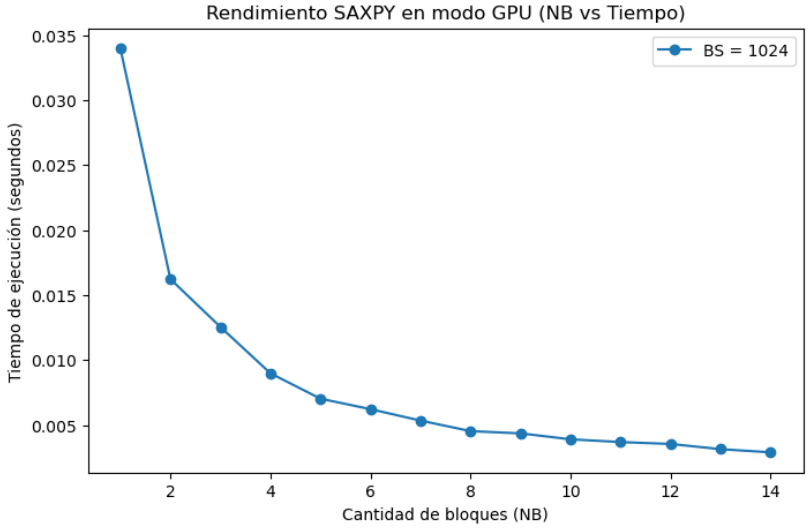


Figura 4: SAXPY GPU MODE, NB $v/s$ t

Finalmente pudimos apreciar que a medida que aumentabamos el número de bloques ($NB$) el tiempo de ejecución disminuía exponencialmente, llegando a una asíntota cercana al 0.

# 4. SpeedUp Relativo

Para analizar el rendimiento relativo, calculamos el Speed Up respecto a sí mismo en cada modo de ejecución. En el caso del modo CPU, utilizamos como referencia la ejecución con un único thread, mientras que en el modo GPU tomamos como referencia la ejecución con un solo bloque.

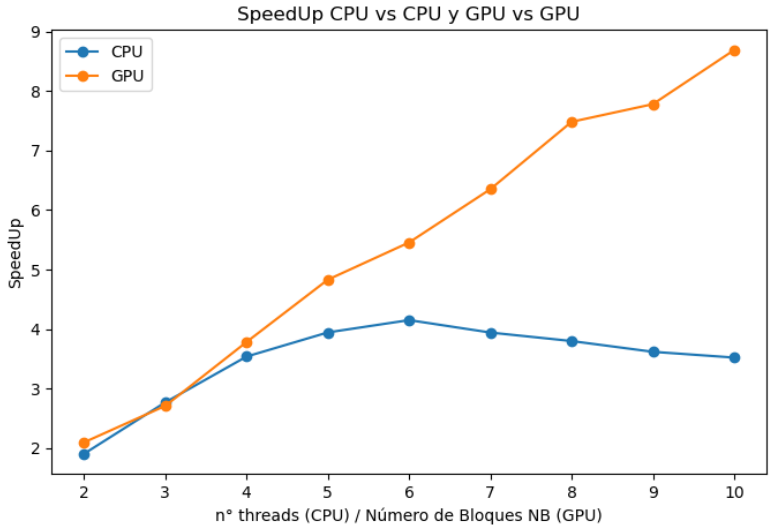


Figura 5: Speed Up relativo

Al analizar la Figura 5, se observa que el Speed Up relativo obtenido en el modo GPU es significativamente mayor en comparación con el modo CPU. En algunos casos, el Speed Up relativo de la GPU puede llegar a ser hasta 9 veces más rápido que el modo GPU con un solo bloque. Mientras que el Speed Up relativo de la CPU llega a ser solo 4 veces más rápido.

## 5. SpeedUp CPU vs GPU

En la Figura 6, hemos representado el cálculo del Speed Up de la GPU en relación con la CPU. Para ello, incrementamos gradualmente el tamaño del problema ($n$) desde 1 hasta $100 \cdot 2^{20}$, con un incremento lineal de 5242880 unidades por punto.

Observando el gráfico, podremos identificar cómo varía el Speed Up a medida que el tamaño del problema aumenta. Esto nos brinda información valiosa sobre la eficiencia relativa de la GPU en diferentes escalas de problemas.

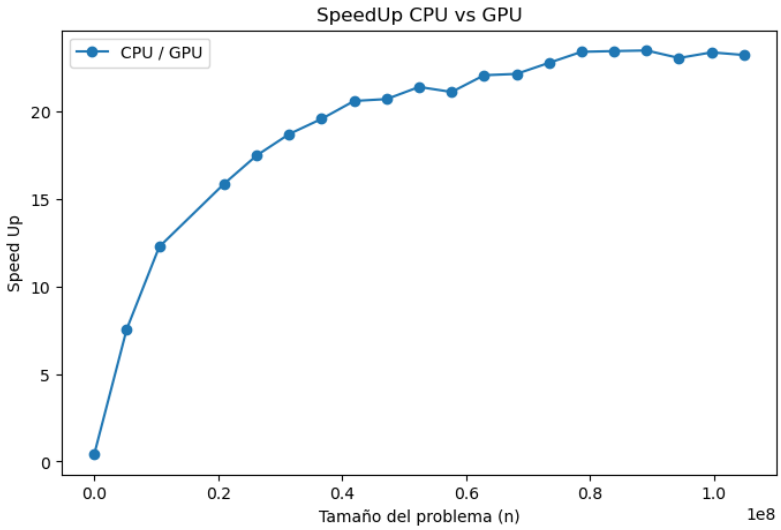


Figura 6: Speed Up GPU vs CPU

De acuerdo con la información proporcionada, la Figura 6 muestra el Speed Up de la GPU respecto a la CPU utilizando la fórmula $Tiempo_{CPU}/Tiempo_{GPU}$. Se observa que el Speed Up sigue un comportamiento similar a

una función logarítmica a medida que aumenta el tamaño del problema ($n$). A medida que el tamaño del problema crece, el Speed Up aumenta, pero lo hace en un ritmo cada vez más lento.

Este comportamiento sugiere que existe un límite en la mejora del rendimiento al utilizar la GPU en comparación con la CPU. A medida que el tamaño del problema continúa aumentando, el Speed Up alcanza su máximo y luego muestra una tendencia a estabilizarse.

Es importante destacar que en el punto máximo, la GPU puede ser hasta 23 veces más rápida que el procesamiento realizado por la CPU. Esta diferencia significativa en términos de tiempo de ejecución indica una clara ventaja al utilizar la GPU para problemas de mayor tamaño.

Estos resultados respaldan la decisión de aprovechar la capacidad de procesamiento paralelo de la GPU en casos donde el tamaño del problema es considerable, ya que se obtiene una mejora notable en el rendimiento en comparación con la CPU.

# 6. Conclusiones

SAXPY es una combinación de multiplicación escalar y suma vectorial. En estos experimentos medimos ciertas métricas con este método.

En el primer ejercicio, medimos el tiempo de ejecución a medida que usamos bloques. Como se puede apreciar, mientras más bloques, o threads, usabamos, más rápido era nuestro tiempos de ejecución.

Sin embargo, si cambiamos al modo GPU (punto 2), observamos el mismo comportamiento con los bloques, pero se ve que es un rendimiento un poco menor que el del modo CPU.

A la hora de ir al procesamiento gráfico en el punto 3. Como era de esperarse, el GPU muestra un mejor rendimiento que el CPU. Para esto usamos $n = 104857600$.

A continuación se muestra una pequeña tabla que refleja nuestra discusión sobre las ventajas y desventajas de la CPU y GPU.

| **Tipo de procesador** | **Ventajas** | **Desventajas** |
|---|---|---|
| CPU | - Alta compatibilidad con una amplia gama de aplicaciones y algoritmos. | - Menor rendimiento en cálculos altamente paralelos en comparación con las GPUs. |
| GPU | - Paralelismo masivo para cálculos intensivos. | - Limitaciones en la capacidad de memoria y no adecuado para todos los tipos de operaciones. |

# 7. Bibliografía


- https://developer.nvidia.com/blog/six-ways-saxpy/
- Clase 17 - GPUs → SiveducMD