MÉTODOS NUMÉRICOS PARA LA COMPUTACIÓN

**Tema 8: Introducción a la Programación OpenMP**

**2020/21**

10 de enero de 2021

**José Amusquívar Poppe | Prashant Jeswani Tejwani**

Universidad de Las Palmas de Gran Canaria

Escuela de Ingeniería en Informática

Índice

[Actividad práctica 1 3](#_Toc60319927)

[Actividad práctica 2 4](#_Toc60319928)

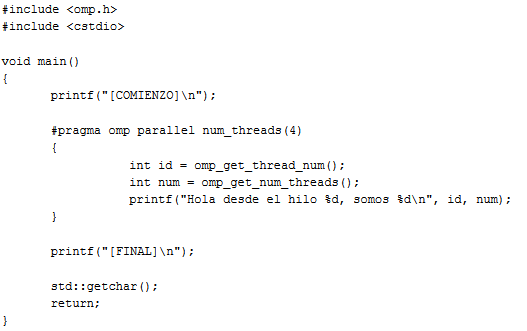
[Actividad práctica 4 8](#_Toc60319929)

[Actividad práctica 5 8](#_Toc60319930)

[Actividad práctica 6 8](#_Toc60319931)

# Actividad práctica 1

Se implementa el siguiente código suministrado:



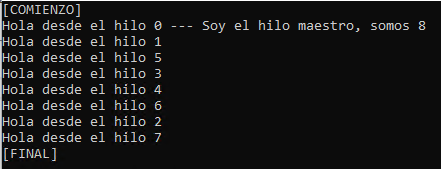
Se ejecuta el programa varias veces para comprobar que no es determinista, ya que el orden en el que aparecen los mensajes varía en cada ejecución. Se ha comprobado este comportamiento empleando 8 hilos. Además, se ha modificado el programa para que sólo el hilo principal (el hilo 0), sea el que muestre el total de hilos.

Figura 1. Primera iteración con 8 hilos. El hilo maestro muestra el total.

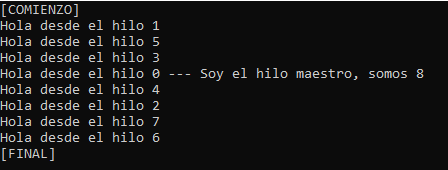


Figura 2. Segunda iteración, comprobando el cambio de orden en las ejecuciones.

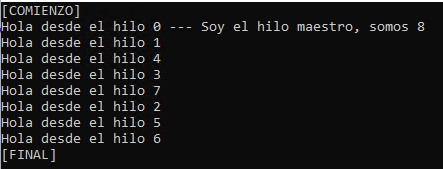


Figura 3. Tercera ejecución, comprobando una vez más su aleatoriedad.

# Actividad práctica 2

Se modifica el programa anterior para que cada hilo realice una tarea que consuma tiempo como, por ejemplo, multiplicar dos números en coma flotante varios millones de veces. Se añade al mensaje que muestra, en cada hilo, el tiempo que ha tardado en ejecutar la operación.

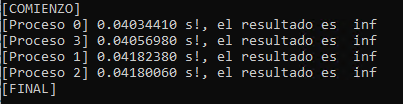
Primero se ha comprobado con 4 hilos, obteniendo unos tiempos muy similares, tal como se puede apreciar en la siguiente figura:

Figura 4. Ejecución del programa con 4 hilos, obteniendo unos tiempos similares.

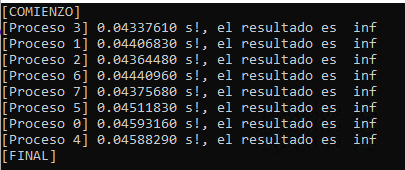
Posteriormente, se presenta la misma ejecución, pero empleando 8 hilos. Los tiempos obtenidos, una vez más, son similares entre sí, tal como se puede apreciar en la siguiente figura:

Figura 5. Ejecución del programa con 8 hilos, obteniendo unos tiempos similares.

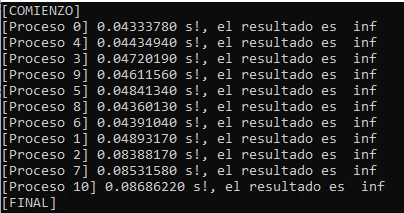
Y, finalmente, se ejecutó el mismo programa, pero con un número de hilos mayor que 8, en este caso, con 11 hilos. Esta vez los tiempos tienen un comportamiento distinto a las anteriores pruebas, tal como se puede apreciar en la siguiente figura:

Figura 6. Ejecución del programa con 11 hilos, comprobando ciertas variaciones.

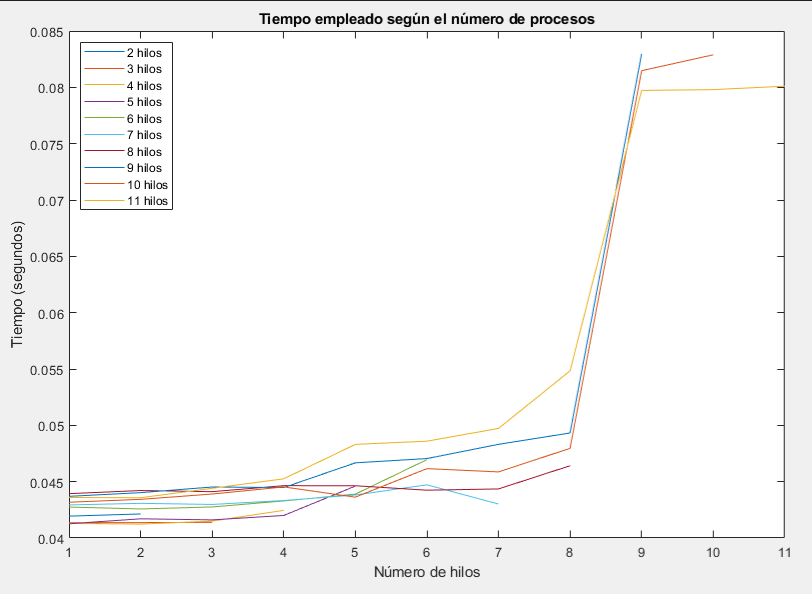
De la figura anterior se puede deducir que el procesador, únicamente, es capaz de paralelizar 8 procesos. Esto se puede comprobar analizando la figura 6, pues, de los 11 procesos realizados, 8 de ellos tienen un tiempo similar, mientras que los otros 3 restantes poseen el doble de tiempo. Es decir, 8 procesos se han realizado paralelamente, mientras que los 3 restantes se han realizado después de que alguno de los hilos del procesador haya terminado su respectiva tarea.

Figura 7.Gráfica que comprueba que, a partir de 8 hilos, el tiempo de ejecución aumenta.

Actividad práctica 3

Se escribe un programa que sume dos vectores de números en coma flotante:

* Se declara tres vectores de 100 elementos
* Se inicializa cada elemento del primero con el valor de su índice
* Se inicializa cada elemento del segundo con el doble del valor de su índice
* Finalmente, se suman los dos vectores en el tercero y comprueba el resultado

Figura 8. Comprobación de resultados. cada vector obtiene lo que debería.

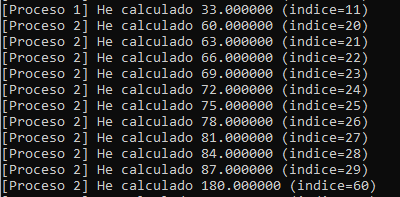
Se paraleliza el código de forma que haya cuatro hilos entre los que se repartan grupos de 10 iteraciones planificadas de forma dinámica.

Figura 9. Se comprueba que cada hilo ejecuta 10 iteraciones a la vez.

Se comprueba que cada proceso, en este caso se trata de 4 procesos, ejecuta 10 iteraciones, sin embargo, dado que se ha establecido que tenga un comportamiento dinámico, los hilos no esperarán a que termine uno para poder comenzar su tarea.

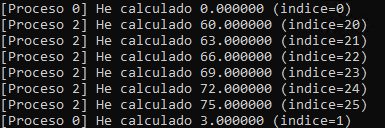


Figura 10. Se comprueba que los hilos no esperan a que otro termine.

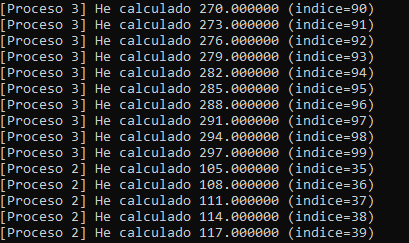
Además, se modifica el programa para que cada hilo muestre qué elementos del vector resultado ha calculado, comprobando las variaciones que se producen en diferentes ejecuciones. Esto puede ser comprobado mediante el índice del vector resultado que ha operado el hilo.

Figura 11. Se comprueban los índices que opera cada hilo.

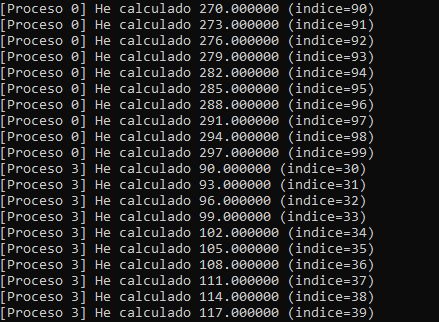


Figura 12. Se comprueba en otra ejecución que los índices e hilos varían.

Con lo que se puede comprobar que cada hilo ha operado distintos elementos del vector en distintas ejecuciones del programa.

# Actividad práctica 4

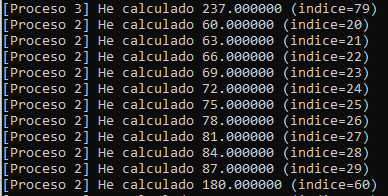
Se modifica el programa anterior para que la planificación se realice de forma estática y ejecutándolo varias veces para comprobar los cambios que se producen en su comportamiento.

Figura 13. Se comprueba que cada hilo realiza 10 iteraciones.

Cambiando el comportamiento de los hilos de dinámico (“dynamic”) a estático (“static”), se consigue que cada hilo realice sus 10 iteraciones sin interrupciones, por lo que si hay hilos que deseen realizar sus operaciones, éstos deben esperan a que el hilo actual termine las suyas.

Se añade al programa las instrucciones necesarias para medir el tiempo de ejecución de la suma de los dos vectores y comprobando las diferencias que se producen. Se añade al programa las instrucciones necesarias para medir el tiempo de ejecución de la suma de los dos vectores y comprobando las diferencias que se producen.

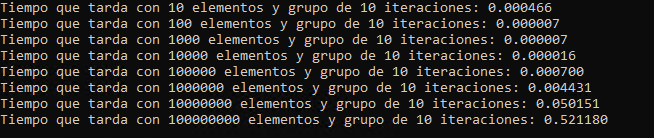
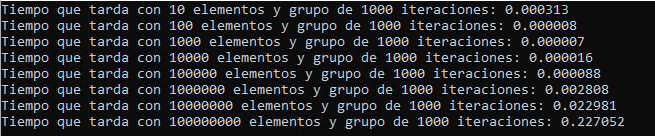
* Se utiliza varios tamaños de vector (100, 1000, 10000…)
* Se utiliza varios tamaños de grupo de iteraciones (10, 100…)

Figura 14. Comprobación empleando conjuntos de 1000 iteraciones.

Figura 15. Comprobación empleando conjuntos de 10 iteraciones.

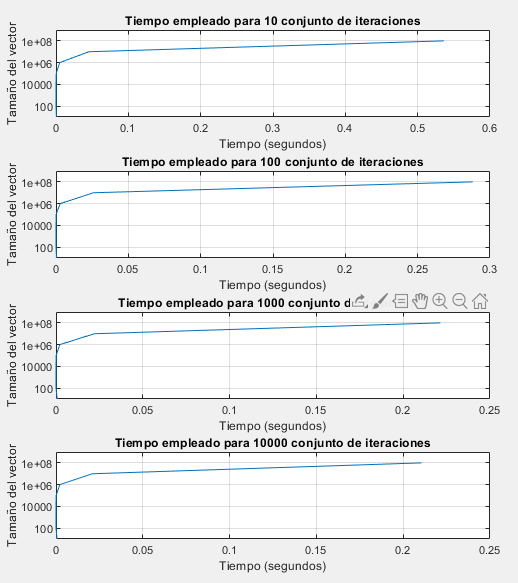
 Finalmente, se dibuja una gráfica que detalle las diferencias encontradas. Dado que existen dos variables independientes, el tamaño del vector y del conjunto de iteraciones, se ha realizado un total de ocho gráficas. Se ha decidido que el tamaño del vector dentro de cada gráfica, mientras el tamaño del conjunto de iteraciones, únicamente, varía al realizar una nueva gráfica. Además, para que las gráficas se visualicen mejor, se ha invertido la gráfica, situando el tiempo invertido en el eje “X” y el tamaño del vector en el eje “Y”.

Figura 16. Gráficas variando el tamaño del vector y del conjunto de iteraciones (1).

Como se puede apreciar en la figura anterior, a medida que aumenta el número del conjunto de iteraciones, el tiempo empleado disminuye. Esto es lógico pues, al disponer únicamente de 10 hilos y tener establecido el comportamiento de los hilos en estático (“static”), si se divide todo el vector en trozos muy pequeños, cada hilo tendrá que realizar más tareas, generando así más tiempo de espera entre hilos, ya que cada hilo en cola deberá esperar a que termine otro para poder realizar sus tareas encomendadas.

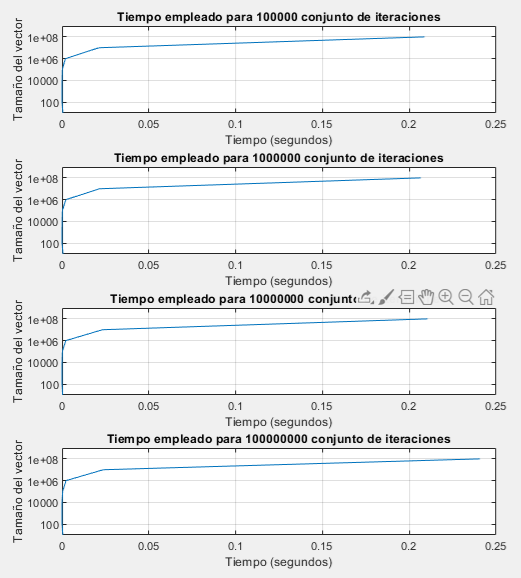
 Esta segunda gráfica es una continuación a la figura 16, en la que se sigue aumentando el número del conjunto de iteraciones llegando, inclusive, a establecer un número de conjunto de iteraciones igual al tamaño del vector. En esta gráfica se comprueba que, mientras el conjunto de iteraciones asciende, el tiempo empleado también lo hace.

Figura 17. Gráficas variando el tamaño del vector y del conjunto de iteraciones (2).

# Actividad práctica 5

Se escribe un programa que suma y multiplica, de forma separada, los elementos de dos vectores de números en coma flotante:

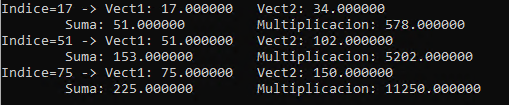
* Se declara cuatro vectores de 100 elementos
* Se inicializa cada elemento del primero con el valor de su índice
* Se inicializa cada elemento del segundo con el doble del valor de su índice
* Se suma los dos vectores en el tercero y se comprueba el resultado
* Se multiplica los dos vectores en el cuarto y se comprueba el resultado

Figura 18. Se comprueba que los vectores están bien creados.

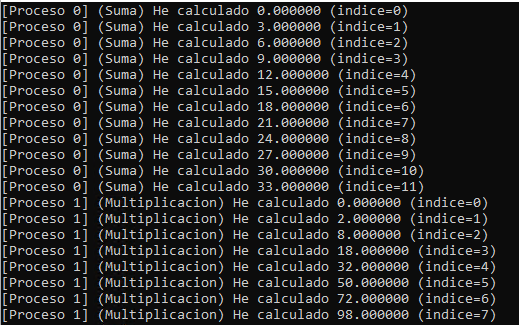
Se paraleliza el código de forma que haya dos secciones; la primera sección realizará la suma y la segunda sección realizará la multiplicación. Además, se modifica el programa para que cada hilo muestre qué elementos de cada vector resultado ha calculado y se comprueba las variaciones que se producen en diferentes ejecuciones.

Figura 19. Se comprueba que cada hilo ejecuta o la suma o la multiplicación.

Se puede apreciar que en la figura anterior el hilo principal, (el 0), es el encargado de realizar la suma mientras que el hilo 2 es el que realiza la multiplicación.

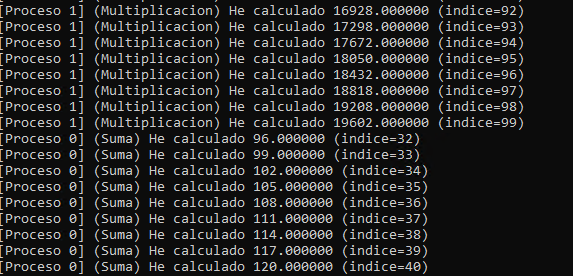
Tal como se aprecia en la figura 19 y 20, el hilo encargado de realizar la suma, en todas las ejecuciones que se han realizado, ha sido siempre el hilo 0, mientras que la el hilo 1 realiza la multiplicación. Además, dado que sólo existen dos secciones en el programa, el número de hilos máximo que se usarán será también de dos.

Figura 20. Otra comprobación de la suma y la multiplicación.

# Actividad práctica 6

¿Cómo se comporta el programa anterior si hay menos hilos que secciones?

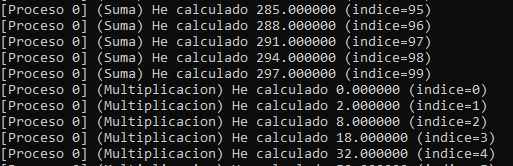
Puesto que cada sección es ejecutada, únicamente, por un hilo, si existen menos hilos que secciones, existirán hilos que realicen más de una sección. Por ejemplo, en el caso anterior, en el que el número de secciones es igual a 2, si sólo existiese un hilo (hilo 0), éste sería el encargado de realizar tanto la suma como la multiplicación.

Figura 21. Se comprueba que el hilo principal realiza ambas operaciones.

¿Cómo se comporta el programa anterior si hay más hilos que secciones?

Tal como se comentó en la pregunta anterior, debido a que cada sección es ejecutada, únicamente, por un hilo, si se indica un número mayor de hilos que supere a las secciones, el programa ignora los hilos sobrantes. Por tanto, el número de hilos debería ser menor o igual al número de secciones.

Se modifica el programa para que haga un uso adecuado de cuatro hilos.

**CAPTURA**