**Práctica 2: Computación paralela con planetas**

Índice

[1. Ejercicio 1 2](#_Toc57751135)

[2. Ejercicio 2 2](#_Toc57751136)

[2.1 Planets2i – primer *bucle i* 2](#_Toc57751137)

[2.2 Planets2j 4](#_Toc57751138)

[2.3 Ambas versiones – último *bucle i* 5](#_Toc57751139)

[3. Ejercicio 3 6](#_Toc57751140)

[4. Ejercicio 4 7](#_Toc57751141)

[4.1 Ejecuciones en 8](#_Toc57751142)

[4.2 Planificaciones parciales - planets2i con distintas planificaciones para solo un bucle 9](#_Toc57751143)

[4.2.1 Planets2i con distinta planificación para el primer bucle 9](#_Toc57751144)

[4.2.2 Planets2i con distintas planificaciones para el segundo bucle 11](#_Toc57751145)

[4.3 Planificaciones para todo el programa 14](#_Toc57751146)

[4.3.1 Planets2i completamente planificado 14](#_Toc57751147)

[4.3.2 Planets2j completamente planificado 17](#_Toc57751148)

[4.4 Conclusiones 20](#_Toc57751149)

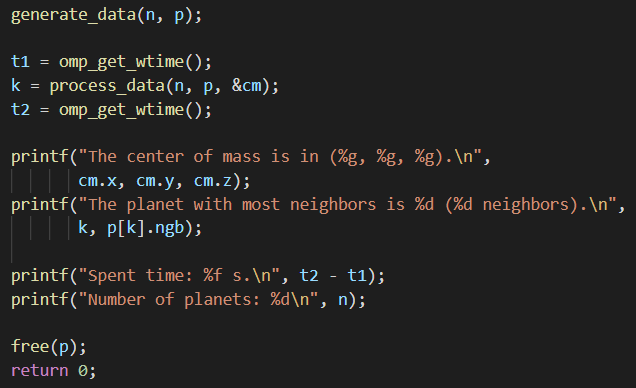
[5. Ejercicio 5 22](#_Toc57751150)

**Integrantes del equipo:**

* **David Arnal García**
* **Alejandro Sánchez Melero**

# ****Ejercicio 1****

En el inicio del programa principal (*main*) se han declarado dos variables, las variables , de tipo doble (*double)*. A continuación, se imprime por pantalla el tiempo de ejecución, siendo este la resta de así como el número de planetas, siendo este último el valor de una variable auxiliar que toma el valor de



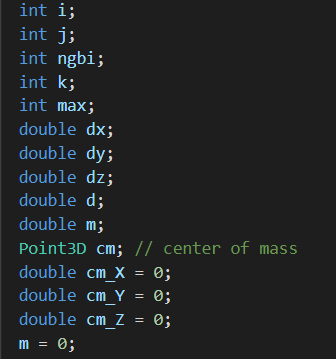
Ilustración

# Ejercicio 2

En este ejercicio, explicaremos cada bucle por separado y, por último, el que comparten ambas versiones. También es importante mencionar que ha sido posible realizar la paralelización de todos los bucles debido a que no hay ninguna dependencia de datos entre los bucles.

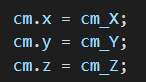
## Planets2i – primer *bucle i*

En este archivo se ha paralelizado el bucle externo y, por tanto, se consigue aquí la paralelización óptima. Al paralelizar el programa, se deben crear unas nuevas variables de tipo que se sustituirán por las variables del ya que, al paralelizar este programa, no puede acceder a las estructuras de datos, por lo que se sustituyen todos los accesos a la estructura por las variables anteriormente mencionadas. El fragmento de código empleado aquí es:



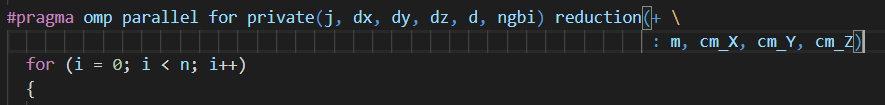
Ilustración

Posteriormente, se debe actualizar correctamente fuera del bucle la estructura con sus valores correspondientes. El código de este fragmento es:



Ilustración

Ahora mencionaremos las directivas para conseguir paralelización en el primer bucle que aparece en el programa. Se han incluido como privadas las variables ya que cada hilo toma sus propios valores en cada iteración. Además, las variables al contribuir al estado global del programa, son de tipo reducción



Ilustración

## Planets2j

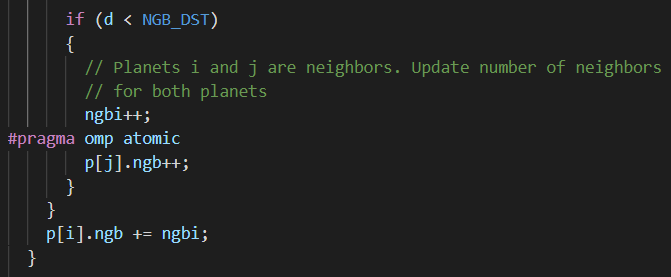
En esta versión del bucle se paraleliza el bucle interno, es decir, el bucle Esta versión será menos óptima que la anterior y, como se podrá observar, posteriormente, en otro ejercicio, se obtendrá una mayor complejidad temporal que en el bucle externo debido a la mayor sobrecarga por la activación y desactivación de hilos. En este bucle se han declarado como privadas las variables ya que, en cada iteración, cada hilo tendrá un valor propio, y no compartido.

Además, como intento de optimización de código y aseguramiento de un correcto valor para la variable , se ha incluido dentro de una cláusula reducción.



Ilustración

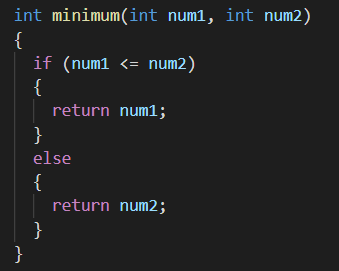
En esta versión, es importante mencionar que al realizar la instrucción no se producen condiciones de carrera, ya que cada hilo mantiene su propio valor para pero sí que es importante incluir una directiva al vector en la posición ya que todos los accesos y escrituras en las variables se producen de forma concurrente y , en este caso, no es una variable privada.



Ilustración

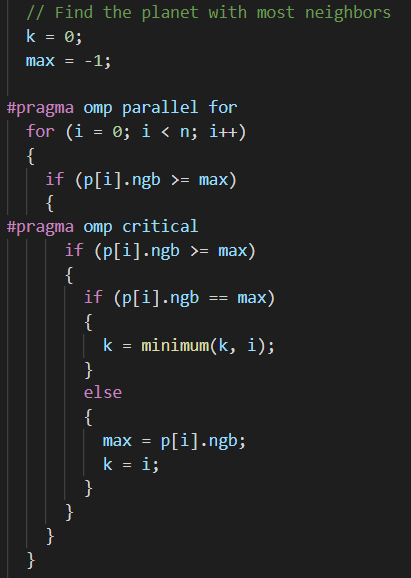
## Ambas versiones – último *bucle i*

El últimoque aparece en ambas versiones se puede paralelizar. En este bucle no se ha declarado ninguna variable como privada ni de tipo reducción, por lo que todas las variables son compartidas. Sin embargo, en este bucle sí que se produce una condición de carrera a la hora de acceder a la escritura en las variables por lo que se deberá declarar una sección crítica y duplicar la sentencia además de incorporar en ella el comparador ya que, si un valor de planetas vecinos coincide con el máximo, se deberá seleccionar el planeta de menor índice. Por este motivo, se introduce otro donde se calcula el mínimo entre los dos planetas con una función propia, llamada con el siguiente código:



Ilustración

Siendo el código completo del último el siguiente:



Ilustración

# Ejercicio 3

En el siguiente ejercicio mostraremos explícitamente distintas planificaciones para los programas. En este momento, comenzaremos analizando qué sucede si solo planificamos un solo bucle en . Si, por ejemplo, se planifica el primer bucle **no encontraríamos diferencias significativas de rendimiento entre las distintas planificaciones.** En otro caso, **si únicamente se planifica el segundo bucle , tampoco tendría por qué haber diferencias significativas** en cuanto a las prestaciones, tal y como se podrá comprobar en el siguiente ejercicio.

Por otra parte, si hacemos un análisis más general, **no se encontrará ninguna planificación óptima** (es decir, mejor que las demás) **si se planifica el programa completamente**, es decir, ambos bucles , ya que la asignación de los hilos siempre se realizará desde el inicio de los bucles y no habrá problemas de rendimiento por ello, aunque podría ser que la planificación diera peores resultados debido a una peor gestión de hilos, ya que no son tareas cortas y sencillas, y podría no ser óptimo el tiempo de ejecución para este caso.

Sin embargo, si se planifica el programa completamente, sí habrá diferencias notables, siendo **la planificación la mejor de todas ellas**, y cuanto más grande el , mejores prestaciones para todas las planificaciones, ya que habrá menos desactivaciones y reactivaciones de hilos. **Además, la peor planificación, en este caso, sería la planificación dinámica**, que sufriría bastante en cuanto a prestaciones debido a las continuas activaciones y desactivaciones de los hilos que se realizan con esta planificación, y destacar que, a mayor las prestaciones mejorarían, aunque no a un nivel óptimo, debido a que aquí la planificación dinámica no es adecuada, en general.

Todo lo anterior comentado se analizará y se podrá observar detalladamente en el posterior ejercicio.

# Ejercicio 4

En este ejercicio, explicaremos cómo hemos lanzado los programas en , a continuación, analizaremos unas planificaciones parciales del programa y, finalmente, analizaremos los programas y completamente planificados.

## Ejecuciones en

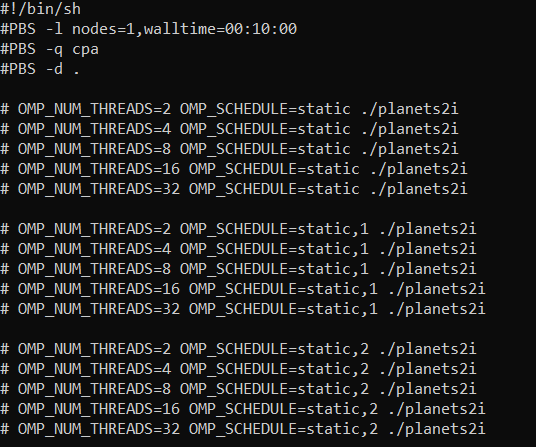
Antes de analizar las planificaciones gráficamente, vamos a explicar cómo lanzamos los programas en . Para esto, hemos creado unos archivos de extensión llamados y en los que hemos ido haciendo las ejecuciones de y , respectivamente.

En estos ficheros, se han definido las propiedades del trabajo sobre el sistema de colas, en el que elegimos trabajar con un único nodo de los seis disponibles en , ya que se trabaja en una máquina de memoria compartida donde todas las ejecuciones se realizan sobre el mismo nodo, y no se puede acceder a la memoria de otros nodos. Por tanto, decir, en este momento, que, se ha decidido ejecutarlo hasta con 32 hilos y no con más, ya que los programas se ejecutan solo en un nodo de que tiene 2 procesadores y cada uno de ellos 16 núcleos, por lo que un nodo únicamente admite hasta 32 hilos, y no tendría sentido ejecutarlo con más hilos.

Por otra parte, junto al número de nodos elegidos, incluimos un con un tiempo máximo de duración para nuestras ejecuciones; En nuestro caso, hemos decidido utilizar un tiempo límite de ejecución de 10 minutos.

Además, siendo de la asignatura CPA, escogemos trabajar en la cola de dicha asignatura con el comando y, por último, utilizamos la opción indicando que nuestro fichero estará en el mismo directorio desde donde se ejecutará .

Por último, en ambos ficheros, utilizamos diferentes líneas de ejecuciones para ejecutar ambos programas con diferentes números de hilos y planificaciones. Por ejemplo, para el archivo un fragmento de código de es:



Ilustración

Y el archivo continuaría con ejecuciones con las planificaciones y número de hilos que deseáramos. Lo mismo sucedería para el archivo , cambiando únicamente en el archivo por .

En la entrega, adjuntaremos los archivos y .

## Planificaciones parciales - planets2i con distintas planificaciones para solo un bucle

### Planets2i con distinta planificación para el primer bucle

Por ejemplo, en el histograma *Planets2i con distintas planificaciones para el primer bucle i*, la planificación del segundo bucle *i* se mantiene por defecto. Debajo de cada histograma se sacarán sus correspondientes aceleraciones y eficiencias. Además, **todos los tiempos se han tomado con un número de iteraciones .**

Ilustración 10

Para tomar las aceleraciones, se ha ejecutado y ha dado como **Se utilizará este valor para todas las aceleraciones.**

Ilustración

Como se ha podido observar en las ejecuciones anteriores, la planificación estática es la que mayor tiempo ha dado, por lo que es evidente que es la que tiene una menor aceleración en esta gráfica. Respecto a todas las planificaciones que hemos ejecutado, es la que se puede destacar, ya que las demás obtienen una aceleración muy parecida.

Ilustración

Por tanto, todas las eficiencias en esta gráfica son muy parecidas, excepto para la planificación que obtiene una eficiencia menor.

### Planets2i con distintas planificaciones para el segundo bucle

Ilustración 13

Si ejecutamos únicamente planificando el segundo bucle , no

encontraremos ninguna diferencia significativa respecto a planificaciones.

Ilustración

Por esto, todas las aceleraciones serán muy parecidas.

Ilustración 15

Así como sus eficiencias.

Por otra parte, se han ejecutado los programas completos con determinadas planificaciones, pudiendo observarse algunos casos particulares en los que el coste aumenta considerablemente.

## Planificaciones para todo el programa

### Planets2i completamente planificado

Ilustración 16

En este apartado comenzamos analizando qué sucede si se planifica el programa completamente con una planificación determinada. En este caso, el tiempo de ejecución únicamente es mayor en el caso de la planificación , siendo mayor debido a una mala gestión de la desactivación y reactivación de los hilos. En cuanto a las demás, no hay nada importante que decir, debido a que todas ellas tienen tiempos similares.

Ilustración

Por lo anterior mencionado, la planificación aquí es la que tiene una menor aceleración.

Ilustración

Y, de la misma forma, es la que cuenta con una menor eficiencia.

### Planets2j completamente planificado

Ilustración 19

Aquí se puede destacar, para empezar, el mal rendimiento de la planificación dinámica, debido a las continuas desactivaciones y reactivaciones de hilos que realiza, comparado con las planificaciones estáticas, por ejemplo.

Aunque la planificación obtendría un rendimiento parecido a la planificación estática con varios y con pocos hilos, se dispara cuando se ejecuta con muchos hilos, por ejemplo, con 32 hilos, por lo que no sería tan buena su gestión con las activaciones y desactivaciones de hilos.

Además, destacar aquí que cuanto mayor en cualquier planificación, mejor será el rendimiento, debido a que se harán menos desactivaciones y activaciones de hilos, siendo como mejor planificación, para este caso, la estática (con mayor ).

Ilustración

Por tanto, la mayor aceleración será para el con mayor (*i.e.* , 4), seguido del hasta 16 hilos, y de los demás con menor , para el caso general. Por otra parte, las planificaciones dinámicas serán la que peor rendimiento obtengan debido a lo anterior mencionado.

Ilustración

Y las eficiencias tendrán la misma proporción que las aceleraciones anteriores.

## Conclusiones

Como conclusión, podemos acabar comparando la mejor planificación para completamente planificado con la mejor planificación para completamente planificado.

Haciendo esto, saldrían las siguientes gráficas:

Ilustración

Ilustración

Ilustración

Para concluir, se puede observar, a través de las aceleraciones y las eficiencias, que es el programa más rápido para todos los casos (en comparación con ), teniendo sentido, ya que, cuanto más externo sea el bucle paralelizado, más eficiente debería ser y, con este último gráfico, observamos que con 2 y 4 hilos es donde más eficiencia hay, y cuantos más hilos se utilicen, más eficiencia se pierde en ambos programas.

# Ejercicio 5

En este ejercicio hemos creado varias variables auxiliares para poder mostrar las iteraciones procesadas por cada hilo, así como el centro de masa de los planetas que ha procesado cada hilo, además de una región paralela anterior al primer bucle .

A continuación, indicamos en la siguiente fotografía las variables auxiliares necesarias (están incluidas las variables a pesar de no tener relevancia).

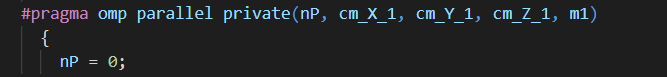
Imagen que contiene reloj, monitor, pantalla, cuarto

Descripción generada automáticamente

Ilustración

La variable es el contador de planetas que procesa cada hilo, las variables son las variables que contendrán el centro de masa en las 3 dimensiones para cada hilo, y la suma de las masas de todos los planetas que procesa un hilo.

Para poder desarrollar este ejercicio hemos creado una región paralela como hemos mencionado antes, la cual abarca todo el bucle y acaba antes de las asignaciones del último .



Ilustración

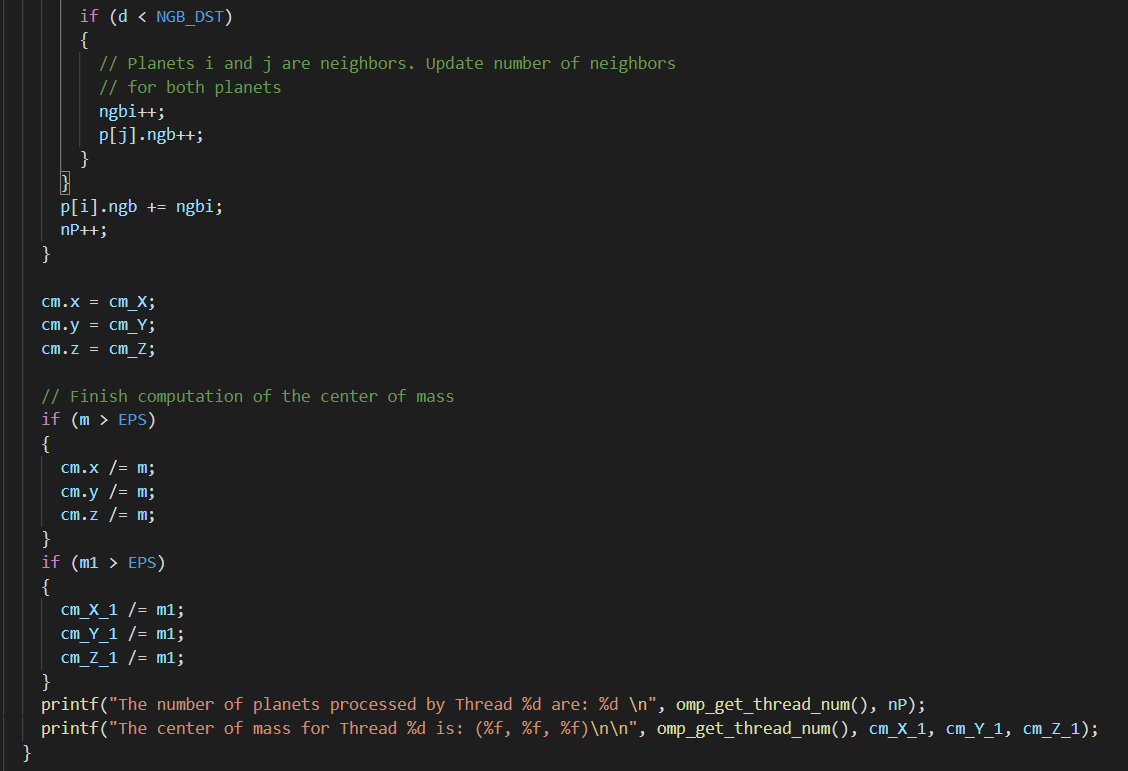
A continuación, mostramos cómo asignamos el valor a estas variables:

Texto

Descripción generada automáticamente

Ilustración

Todas las variables son privadas, ya que cada hilo mantiene un valor para cada variable, y ponemos la variable en cada iteración para que cada hilo comience con el valor asignado a 0, como mostramos en el siguiente fragmento de código:



Ilustración

En cada interacción del bucle se incrementa el contador de número de planetas para ese hilo. Luego hacemos la misma condición que para con y dividimos las variables que contienen el centro de masa en las tres dimensiones por la masa de los planetas. Por último, mostramos por pantalla el número de planetas procesados por cada hilo, así como el centro de masa de los planetas procesados por dicho hilo.