

Practica 2 trazado de rayos en openmp

Ismael Fernández Herreruela



Ratón de computadora

Descripción generada automáticamente con confianza baja

**EJERCICIO 1**

Texto

Descripción generada automáticamente

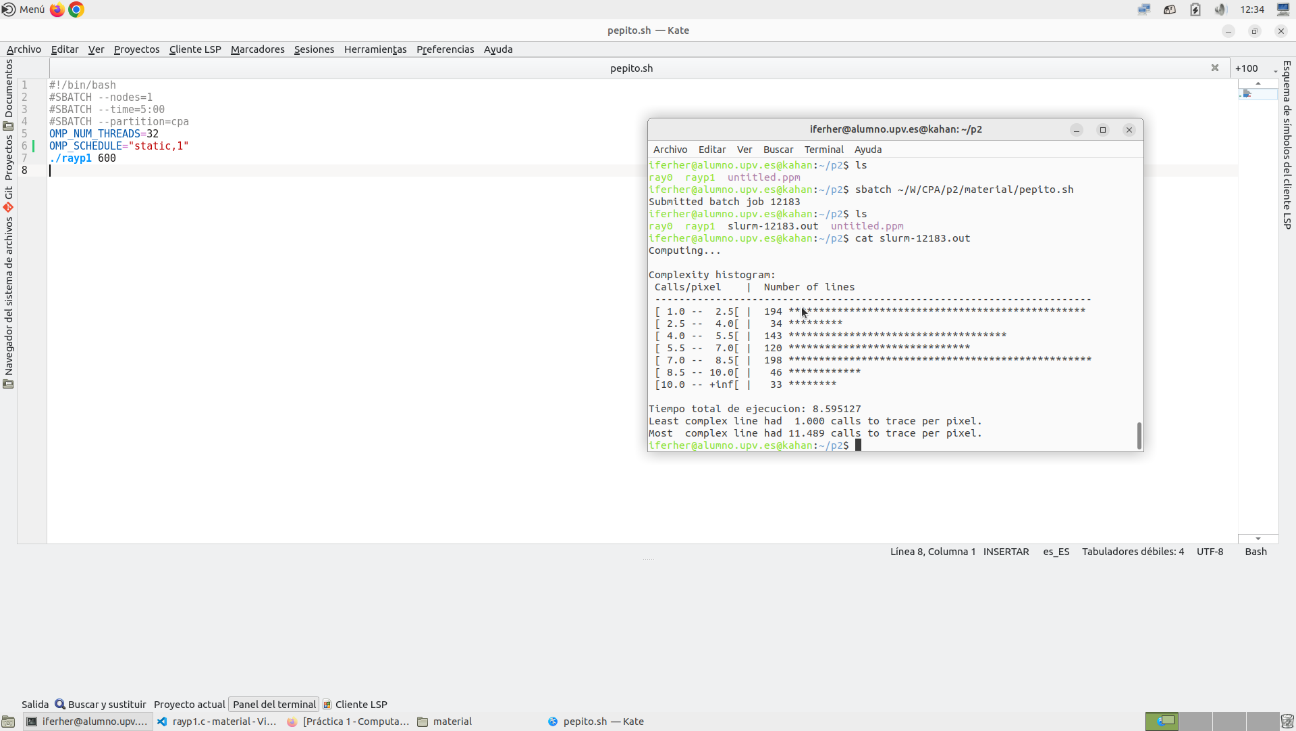
En la paralelización de este apartado, hay que privatizar las variables ***pixel, xx, yy, ncalls, raydir*** ya que son las que se van a modificar, y por otro lado hay que hacer un reduction de la variable ***ncalls\_line*** ya que tenemos un operador +=. En este caso no hay que hacer private la variable ***ncalls\_line*** porque en este for, la variable ya esta inicializada.

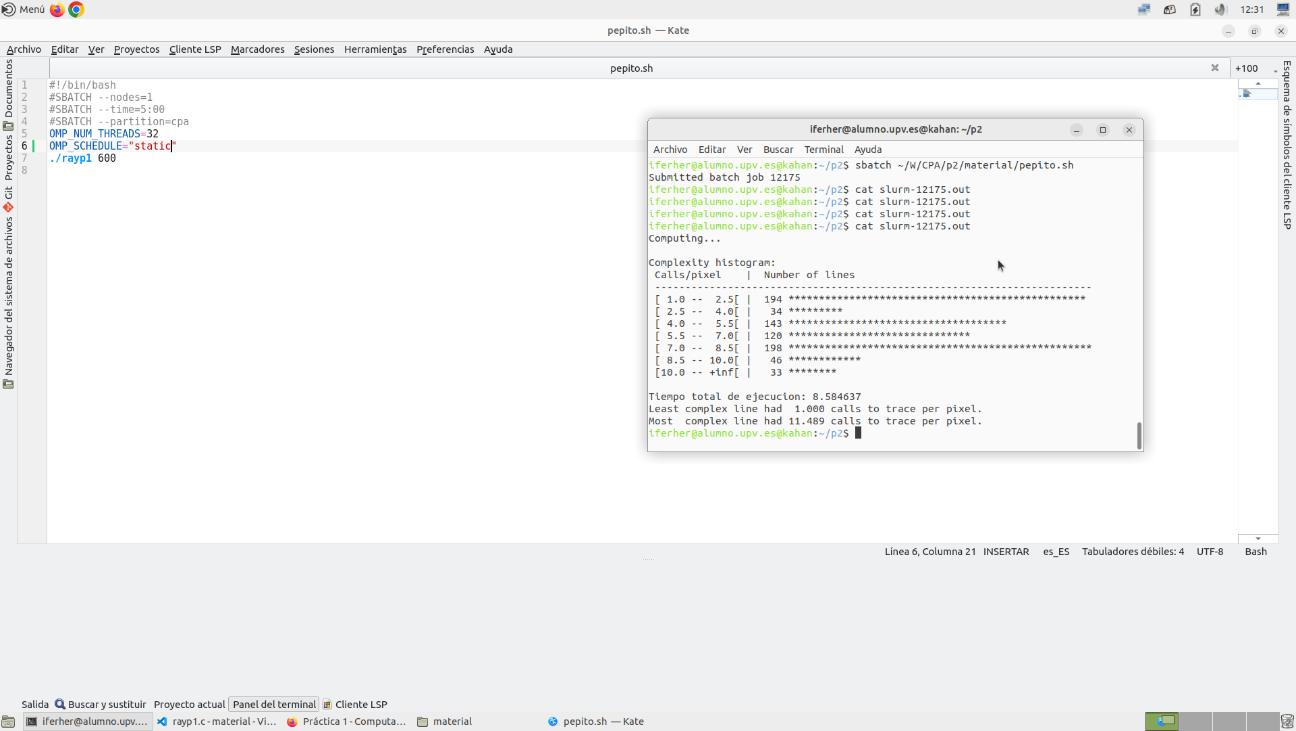
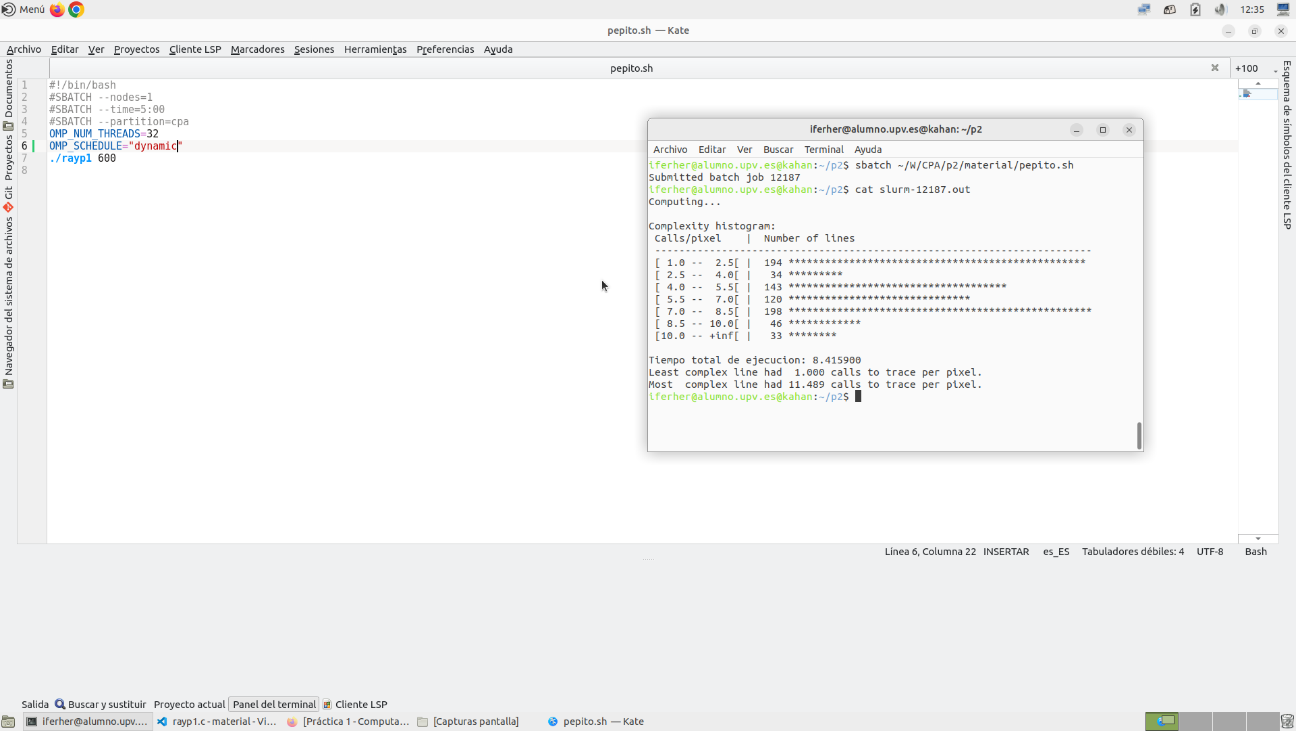
**EJERCICIO 2**

Texto

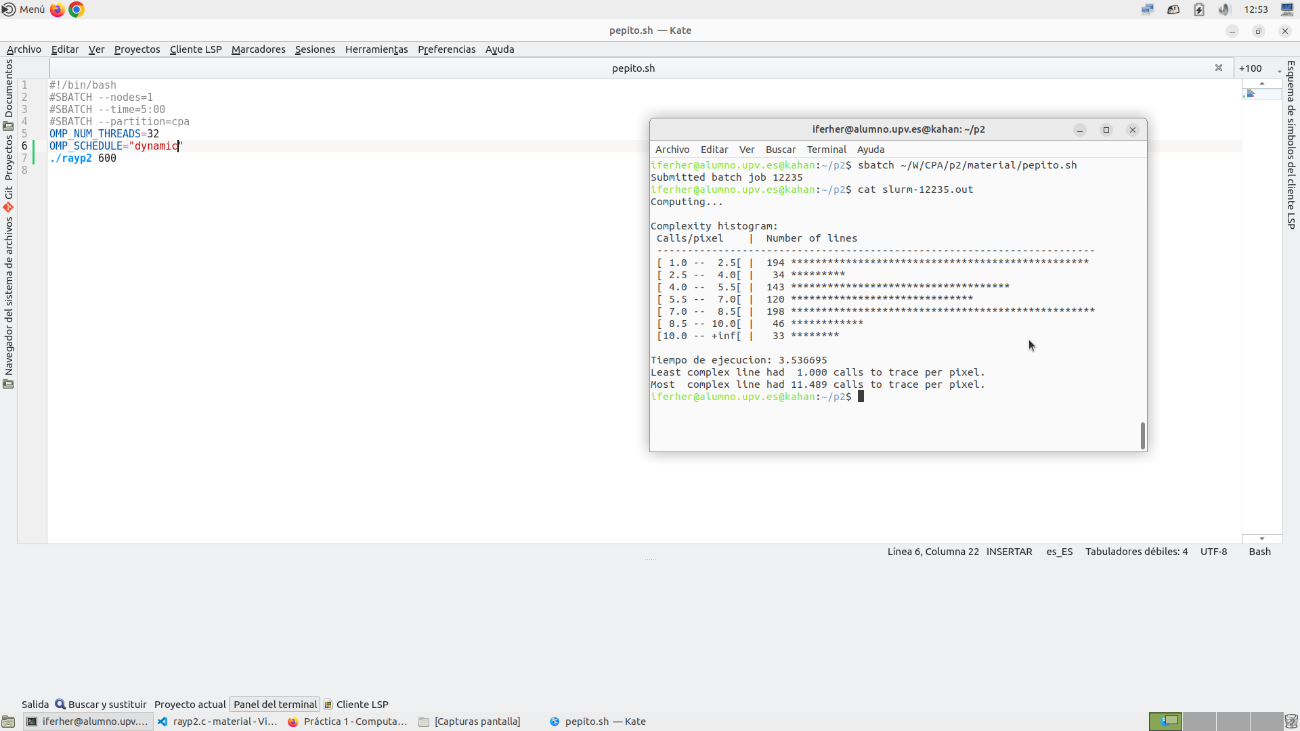
Descripción generada automáticamente

En este ejercicio teníamos que paralelizar el bucle externo, por lo que las cosas cambian un poco. Al estar en el bucle externo también entra la parte con los if. También podemos observar que ***ncalls\_line*** se inicializa en este bucle por lo que no hay que ponerlo como reduction esta vez. El único cambio significativo en este apartado es el ya comentado, ***ncalls\_line*** deja de ser reduction y se convierte en private, y también añadimos la variable ***x*** a private ya que se utiliza en el for interno.

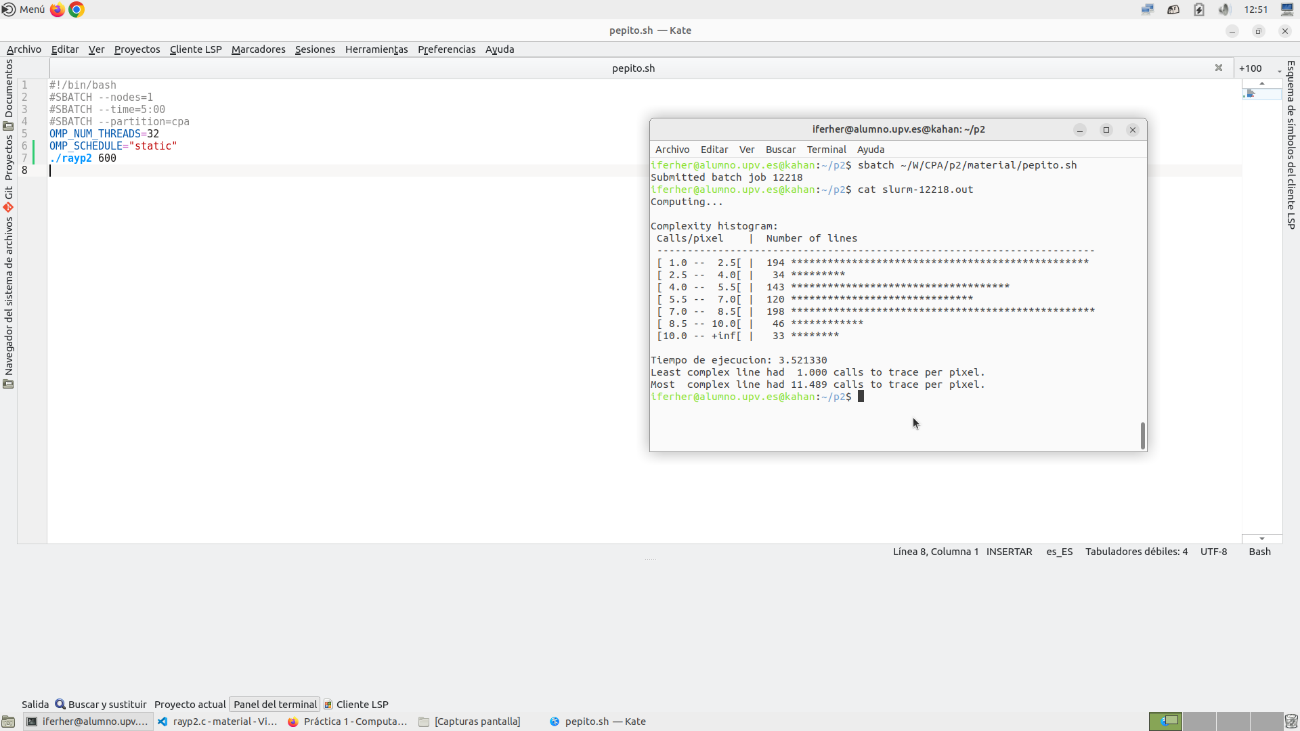
**EJERCICIO 3**

RAYP1:

En el caso de RAYP1 podemos observar que la opción que mejor resultado nos da es la de “Dynamic”, con 8,41s. En segundo lugar iría “static” con los chunks por defecto, con 8,58s. Y en ultimo lugar iría “static” con 1 chunk con un resultado de 8,59s. “Dynamic” nos ofrece unos mejores resultados debido a que las iteraciones del bucle se distribuyen dinámicamente entre los hilos en tiempo de ejecución. En lugar de asignar un conjunto fijo de iteraciones a cada hilo, asigna a cada hilo un lote de iteraciones, y cuando un hilo termina su lote, recibe otro lote hasta que se completen todas las iteraciones. Por otro lado, en “static” las iteraciones del bucle se dividen estáticamente entre los hilos antes de que comience la ejecución del bucle. Esto significa que cada hilo obtiene un número aproximadamente igual de iteraciones para procesar. La distribución es determinada en tiempo de compilación y se mantiene constante durante la ejecución del programa. Y en ultimo lugar “static,1” es similar a “static” pero garantiza que cada hilo reciba al menos una iteración para procesar.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamenteRAYP2:



En el caso de RAYP2, la opción con “static,1” es la mas rápida tardando 3,32s. En segundo lugar tendríamos a “static” con 3,52s y en ultimo lugar “dynamic” con 3,53s. La diferencia de rendimiento entre “dynamic” y “static” podría deberse a la distribución de carga desigual en la generación de rayos y en la llamada a la función trace() para calcular ncalls. Si la carga computacional varía significativamente entre diferentes regiones del espacio de píxeles, el uso de dynamic podría distribuir mejor la carga entre los hilos en comparación con static. Pero en el caso contrario podría hacer que fuese mas lento, que es lo que ha ocurrido esta vez. Dependiendo del código, a veces será mas optimo utilizar “dynamic” y otras será mas optimo utilizar “static”.