El uso de los pools de hilos en Java

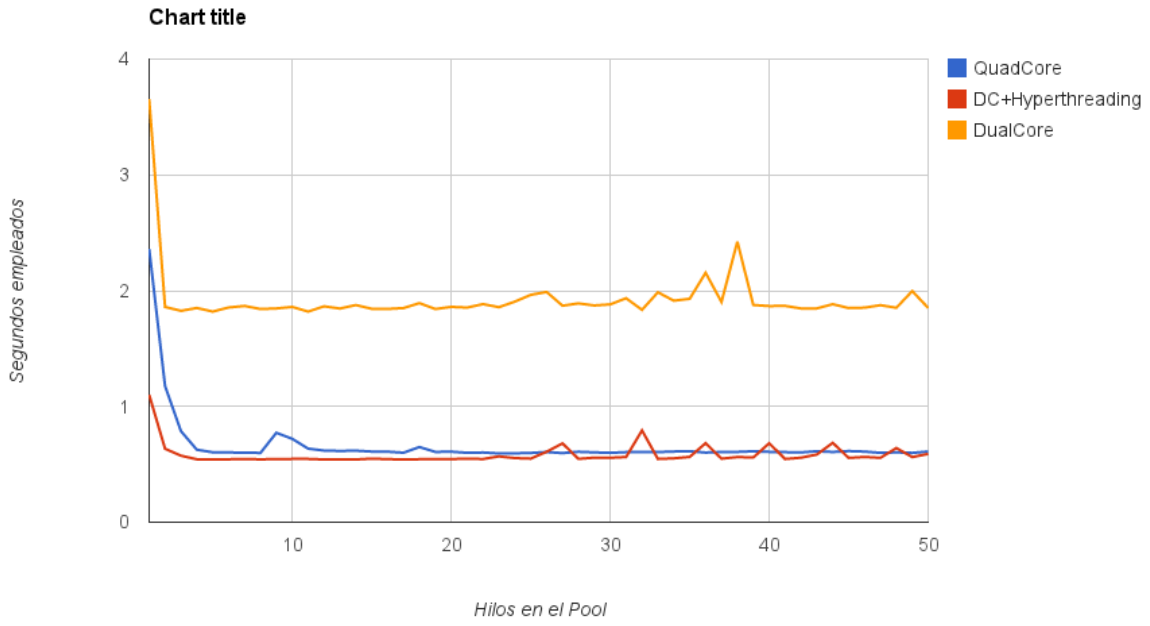
Práctica 5 - PC

Miembros del grupo:
Carlos Giraldo
Ricardo Pragnell

Ejercicio 1: La interfaz *ExecutorService*

Hemos implementado la TareaSimple siguiendo el esquema del enunciado. Para comprobar la eficacia del pool de hilos hemos testeado el programa en distintas máquinas con diferente número de cores.

Efectivamente, se nota que una vez alcanzado el número de cores del sistema la eficiencia del pool se estanca. Aunque siempre existe la posibilidad de que, con tantos cambios de contexto, al aumentar el número de hilos hayan iteraciones que resulten más costosas en tiempo.



Ejercicio 2: La interfaz *CompletionService*

Siguiendo el enunciado creamos un ***pool*** de 10 hilos de los que nos servimos para realizar 50 ***TareasLargas***. Estas tareas son tratadas en orden según acaban, el cual no es necesariamente el orden en el que se solicita su ejecución.

En caso de que una tarea no acabase, puesto que esperamos a que concluya con la instrucción ***poll*(..)**, en la que especificamos un timeout específico, eso no bloquearía el sistema y permite que se salga de la ejecución aunque hayan tareas sin acabar (permitiendo una finalización asíncrona).

Ejercicio 3: El marco *fork-join*

Comenzamos por descargar la librería htmlparser para ejecutar y observar el funcionamiento de **RastreadorWeb6**. Vemos que para parsear y visitar 1500 url únicas emplea un tiempo de **117,051730516** segundos (teniendo como origen la dirección ***http://www.codeofhonor.com/blog/***).

Después modificamos BuscadorenlacesAction para extender RecursiveAction, en su método **compute()** extrae las urls de la página actual y lanza nuevos RecursiveAction por medio de **invokeAll()**

RastreadorWeb7 funciona igual exceptuando el uso de un **ForkJoinPool** en vez de un executor service tradicional de java5.

Usando el ForkJoinPool la aplicación emplea **60,599071637** segundos en rastrear 1500 urls únicas desde la misma dirección.

Volvemos a realizar la operación desde otra página (***http://en.wikipedia.org/wiki/Oculus_Rift***). Obteniendo los resultados: **57,932198195** segundos para el **RastreadorWeb6** y **41,874069006** segundos para el **RastreadorWeb7**. Queda patente la eficiencia de este método.