Instrucciones de uso

El proyecto incluye varios paquetes, tres de ellos corresponden al código que permite ejecutar tareas a nivel de core (java.lang, net.openhft.affinity, net.openhft.affinity.impl) y los otros son los que hacen la carga de CPU y RAM.

Toca importar varios .jar al proyecto antes de poder ejecutarlo. Estos se encuentran en la carpeta /data/dependencias

La clase Simulator hace el llamado a las otras clases para comenzar el simulador. Aquí se debe especificar las rutas de varios archivos.

La clase que genera carga en RAM se le pasa como parámetro la ruta de un archivo .txt y la ruta de donde se encuentra el archivo RamTest.jar (este se encuentra en la carpeta data).

El archivo .txt tiene la siguiente estructura:

* tiempo que dura la carga,porcentaje de carga

La clase que genera carga de CPU solo recibe como parámetro la ruta de archivo .txt. Este archivo tiene la siguiente estructura:

tiempo que dura la carga, porcentaje de carga core1, porcentaje de carga core2, …, porcentaje de carga core-n

donde n es el número de cores que tiene el computador donde se va a ejecutar la prueba.

La implementación está hecha para tratar de simular el uso core a core pero no se garantiza que se alcance una buena precisión, por ende, se recomienda hasta el momento establecer todos los porcentajes de carga de cada core al mismo valor.

Adicionalmente, existe una clase que pasa de un archivo de Open Hardware Monitor a los dos archivos correspondientes de carga de CPU y RAM. Sin embargo, este asume varias cosas. Las mediciones del log se hacen cada segundo y que se está en una máquina que tiene 4 cores físicos.

El programa se comporta mejor cuando se cumplen estas condiciones:

1. La carga que se va a hacer en CPU no es muy grande. La incertidumbre en la medición crece proporcional al tamaño de la carga que se va a hacer.
2. El cambio de RAM no es frecuente. Si la RAM cambia a menudo entonces pasan dos cosas:
   1. El garbage collector de java no alcanza a reclamar la memoria que se ha asignado previamente y los procesos comienzan a consumir mucha más RAM de la estipulada.
   2. Los procesos que lanza el programa para ocupar la RAM comienzan a generar una carga en la CPU (alrededor de 0.5% de carga por proceso y dependiendo de la maquina se lanzan muchos de estos).

Por ende, si utilizan la clase que pasa de un archivo de open hardware a los .txt correspondientes, revísenlos y hagan los cambios pertinentes (si pueden) para mejorar la ejecución del programa.