# Ejercicio práctico: Evaluación de la relación Compilador-Arquitectura

Deseamos valorar los efectos que producen los mecanismos de optimización que realiza el compilador en la generación del ejecutable de un programa. En particular, como primera aproximación, se propone obtener distintas versiones ejecutables del clásico e histórico benchmark "whetstone.c" (primer ejemplo sencillo) mediante sucesivas compilaciones utilizando las opciones genéricas de optimización creciente O0, O1, O2, O3 y Os (para el gcc).

El programa fuente está disponible en la página web de la asignatura. Como compilador se utilizará el gcc, es decir, el compilador de c del gnu. *Recuérdese que debemos generar un ejecutable de 32 bits. Por este motivo, si estamos trabajando en un computador de 64 bits, por defecto generará un ejecutable de 64bits. Sabremos que estamos en un computador de 64, ejecutando un S.O. de 64 bits (referenciada como X86_64) utilizando el comando "uname –a". En ese caso tenemos que incluir "–m32" en la orden de compilación.*

Tener presente que realizar una optimización añadiendo la opción genérica "-O<n>" en nuestra sentencia de compilación es una forma de resumir un conjunto determinado de optimizaciones particulares, es decir, es una forma de aplicar un grupo de optimizaciones estandarizas. Si se tiene curiosidad, pueden ser consultadas de forma más precisa en el manual del compilador que se esté utilizando la lista de optimizadores concretos que se activan con cada nivel "-O<n>".

Tras este proceso obtendríamos un ejecutable para cada nivel de optimización:

- whetstone-O0 <Nivel mínimo de optimización> <Nivel por defecto><debug>
- whetstone-O1
- whetstone-O2 <Nivel de optimización habitual en la versión final>
- whetstone-O3 <Nivel máximo> <Otros compiladores tienen hasta –O5>
- whetstone-Os <Optimización en tamaño>

El programa whestone se ejecuta como "*path/whestone número*". El número que le proporcionamos determina la duración de la ejecución del benchmarck y debe tener un mínimo que dependerá de las prestaciones de nuestro procesador. Para cualquier duda puede verse el código fuente. En nuestro caso práctico, seleccionando unos márgenes de tiempo que deseamos que dure razonablemente el experimento, podemos determinar el número adecuado de forma heurística.

## Propuesta de ejercicio:

1. Ya que el programa en observación es un benchmark que nos proporciona un índice de prestaciones, el primer experimento sería realizar una observación de la ejecución del bench en modo nativo, con dos cargas diferentes por ejemplo:
    a. ./whetstone-version 100000 (mayor nº ➔ mayor carga)
    b. ./whetstone-version 500000
2. El segundo experimento sería realizar una observación de la ejecución del bench en simulado, con dos cargas diferentes por ejemplo(la carga debe ser menor para que acabe un tiempo abordable ):
    a. m2s ./ whetstone-version 100
    b. m2s ./ whetstone-version 500
3. El tercer experimento sería realizar una observación de la ejecución del bench en simulado con detalle, con dos cargas diferentes por ejemplo(la carga debe ser menor para que acabe un tiempo abordable):
    a. m2s --cpu-sim detailed <configuración> ./ whetstone-version 100
    b. m2s --cpu-sim detailed <configuración> ./ whetstone-version 500

En el primer caso, como se está evaluando un procesador real, hay que adoptar una serie de precauciones sencilla para que las medidas sean consistentes y coherentes. Por ejemplo, el computador debe estar lo más descargado posible para evitar las interferencias y, por supuesto todas las pruebas deben ser realizadas en el mismo computador en las mismas condiciones (cosa que no siempre es tan sencillo de asegurar en un procesador moderno).

Utilizar más de un computador distinto en el punto 1, o más de una configuración en los puntos 2 y 3 sería un ejemplo iniciativa dirigida a realizar una exploración del espacio de estudio en una dimensión más amplia, como comentamos en la exposición teórica en clase, pero que excedería el ámbito de este ejercicio.

## Observaciones experimentales:

Los valores observables serian los siguientes:

1. El tiempo de ejecución, medido en segundos y/o en ciclos (según corresponda en cada apartado).
2. El número de instrucciones/µOperaciones ejecutadas en cada caso (en el punto 1, sería posible utilizando los contadores hardware del procesador, pero esto excedería el ámbito del ejercicio).
3. En el último caso, debido a que realizamos una simulación detallada tendremos una información más completa de lo sucedido en el procesador. Aquí podemos realizar estudios más profundos. Se sugiere que el estudiante seleccione un parámetro que despierte su interés y que realice el análisis de cómo ha evolucionado su comportamiento.

## Visualización y análisis de resultados:

Los resultados obtenidos, normalmente desviados a ficheros, los filtraremos para obtener los valores de los parámetros que nos interesa analizar.

A menudo la forma más ilustrativa en realizar una representación gráfica de los resultados. Para esto resulta útil introducir los resultados en una "hoja de cálculo" que nos permita elegir de una forma cómoda la representación gráfica más adecuada.

## Epilogo:

Todo trabajo para resultar útil debe proporcionar alguna aportación o aprendizaje. El objetivo adicional de este ejercicio es reforzar la mecánica de experimentación práctica y evaluación razonada en general, y en particular introducir al estudiante a las técnicas que utilizará profusamente en las prácticas de laboratorio. Por este motivo, como punto final se propone al estudiante elaborar un informe resumido sobre las principales conclusiones que extrae de las observaciones realizadas.