Cálculo Simbólico
Práctica 2

José Ignacio Carmona Villegas
Juan Hernández García

Estructura de datos

Se han creado dos estructuras de datos muy simples (ComplexFieldMultiplications y ZpFieldMultiplications) que implementan las multiplicaciones escolar y fft para los tipos de datos nativos a la biblioteca JAS (Java Algebra System) de tipo polinomial.

Sendas estructuras implementan las multiplicaciones y aceptan dos polinomios como parámetros, devolviendo el polinomio resultante. Además, la estructura ZpFieldMultiplications da soporte a la base de datos de raíces primitivas de la unidad precalculadas (para acelerar los cálculos).

Las raíces precalculadas son leídas desde un fichero denominado “roots.txt” y almacenadas en una base de datos implementada como dos mapas hash anidados. Durante los cálculos, cada vez que se necesita una raíz se comprueba si está en la base de datos. Si está se utiliza directamente. Sólo si no está se calcula. En este caso se almacena en el fichero roots.txt para futuros usos. Este método de proceder hace que la biblioteca generada se acerque más a la filosofía que utilizan otras bibliotecas de cálculo simbólico.

Gracias a este sistema, se puede ejecutar simplemente el método de multiplicación deseado y medir el tiempo que tarda en ejecutarse para realizar el análisis de eficiencia deseado.

El gran problema que sufre la estructura ComplexFieldMultiplications es que JAS fuerza el uso (en la factoría) del tipo de dato BigComplex, que internamente tiene dos BigRational, lo cual reduce muchísimo la eficiencia de este código (y de hecho ha logrado hacer que el algoritmo FFT en C se comporte mucho peor que el escolar) debido a lo difícil de operar con números racionales.

Otro problema (inevitable) que sufre el cuerpo de los complejos, es el producido por la precisión finita con que se calculan las raíces primitivas de la unidad. Debido a que para calcular raíces primitivas de la unidad en el cuerpo de los complejos, se deben hacer una serie de cálculos que producen números reales, y un ordenador no puede más que aproximar un número real, se acaban obteniendo resultados que a veces provocan situaciones extrañas.

Por ejemplo, al multiplicar x^10 por x^2, el resultado debería ser x^12, pero debido a la situación ya mencionada, además, se obtendrían términos ax^11, bx^10, … con a,b,... tendientes a cero. El problema se agrava aún más al intentar imprimir los resultados, pues debido a que internamente los BigComplex que usa la librería para representar los coeficientes, se representan mediante BigRational, se acaba obteniendo algo así: a/bic/d (esta es la forma de JAS de imprimir el complejo a/b + (c/d)*i) siendo a,b,c,d números enteros extremadamente grandes, y siendo a/b y c/d tendientes a 0.

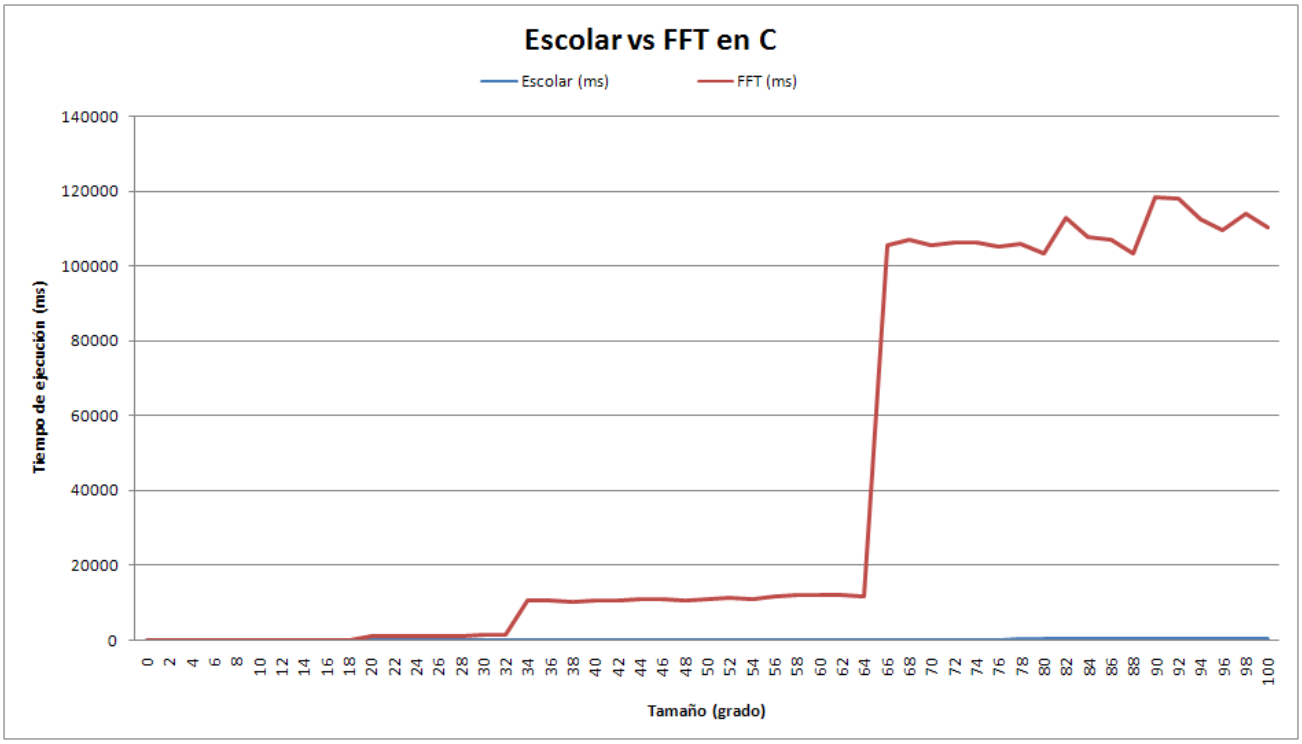
Obtencion de Tiempos

Para obtener los tiempos se ha creado una clase especial que permite:

- Comenzar a contabilizar tiempo.
- Pausar la cuenta del tiempo.
- Reanudar la cuenta de tiempo.
- Finalizar la cuenta del tiempo.
-

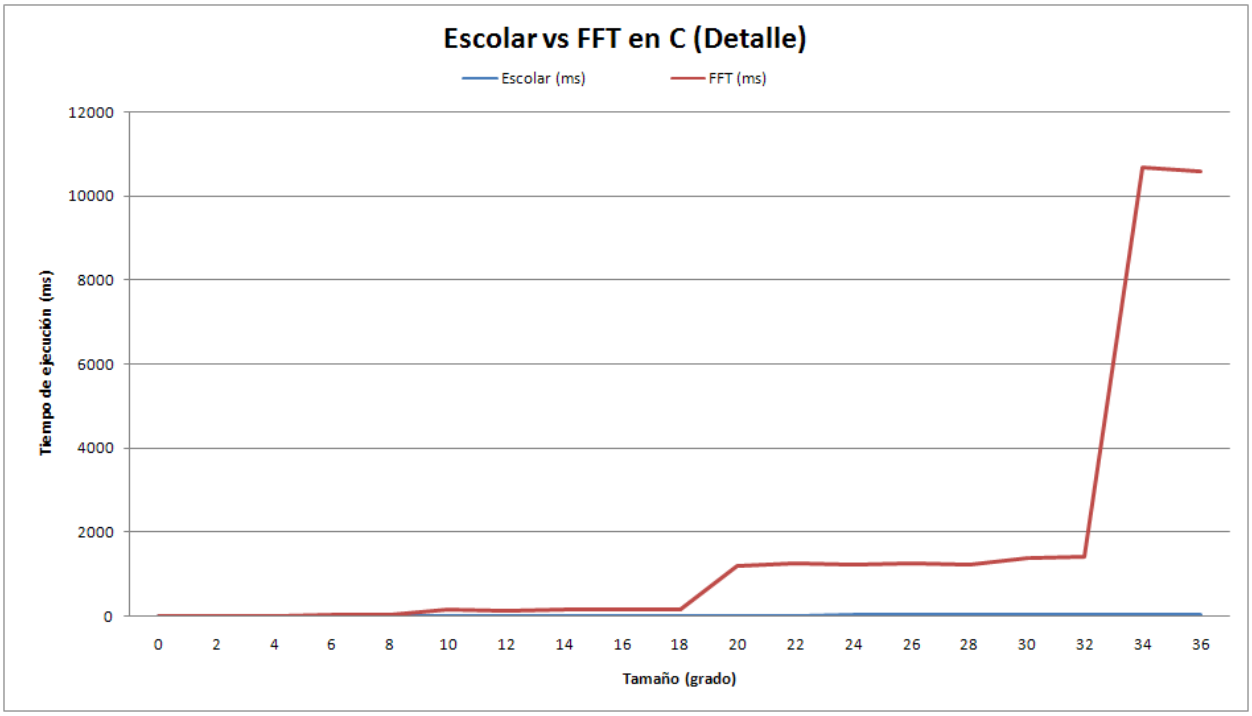
Haciendo uso de esta clase auxiliar, ubicada en el paquete “Util”, se han obtenido los tiempos siguientes:

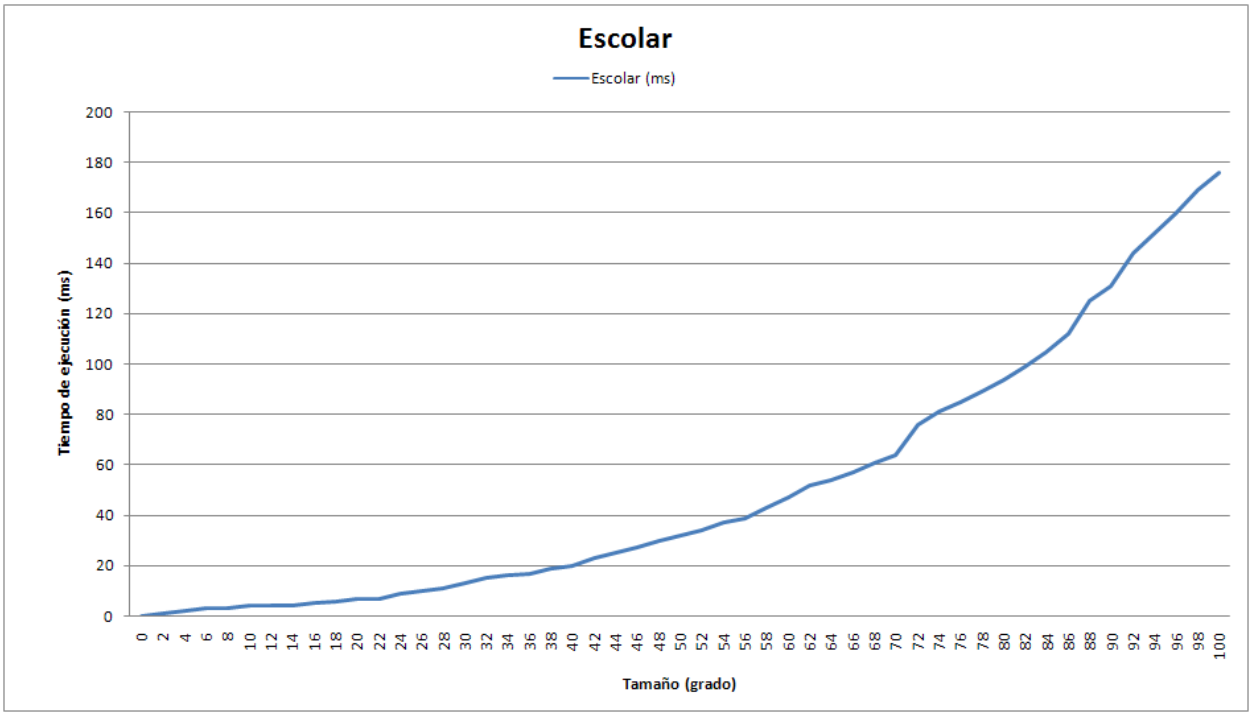
En C:

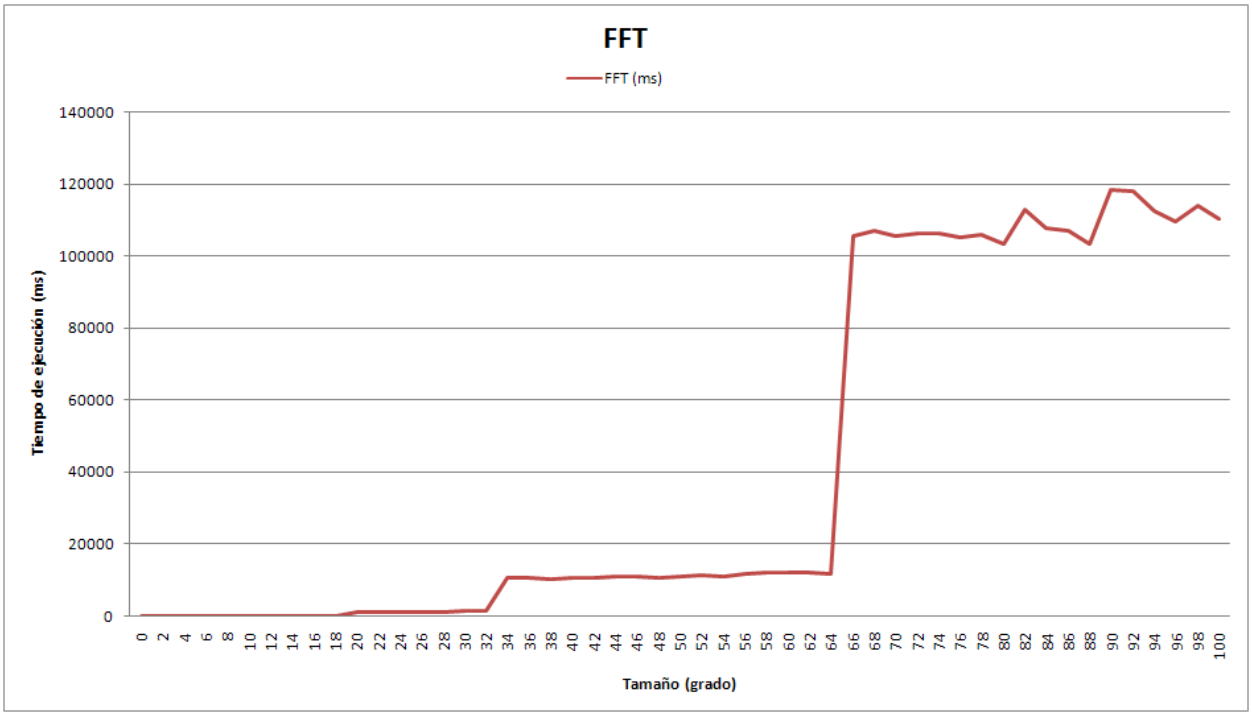


Como se ha comentado antes, la implementación interna de la biblioteca utilizada, a causa de los coeficientes utilizados, provoca que en el cuerpo de los complejos fft se comporte peor que el escolar.

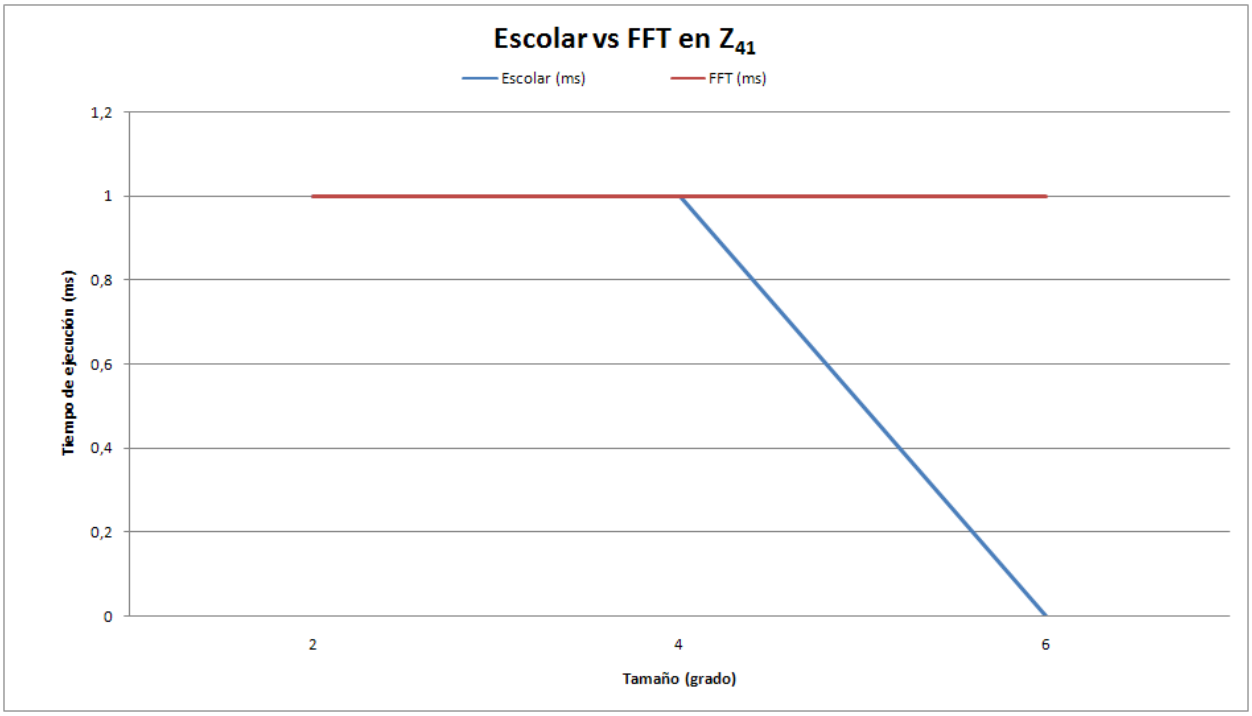
El algoritmo para Zp y C tiene básicamente la misma estructura, y para Zp el comportamiento es excelente, así que la pérdida de eficiencia en los complejos no se debe a un fallo algorítmico, sino a la implementación soporte utilizada.


Escolar vs FFT en C (Detalle)
Escolar (ms)
FFT (ms)
Tiempo de ejecución (ms)
Tamaño (grado)


Escolar
Escolar (ms)
Tiempo de ejecución (ms)
Tamaño (grado)

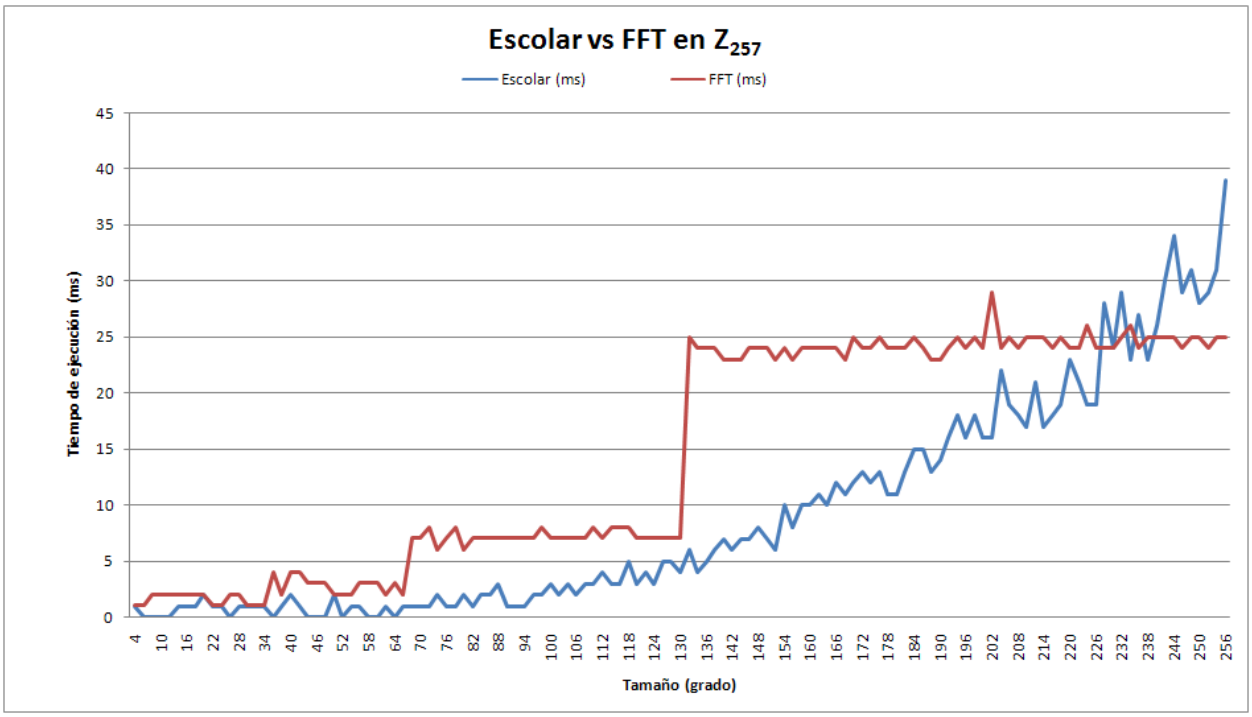


En Z_{41}:

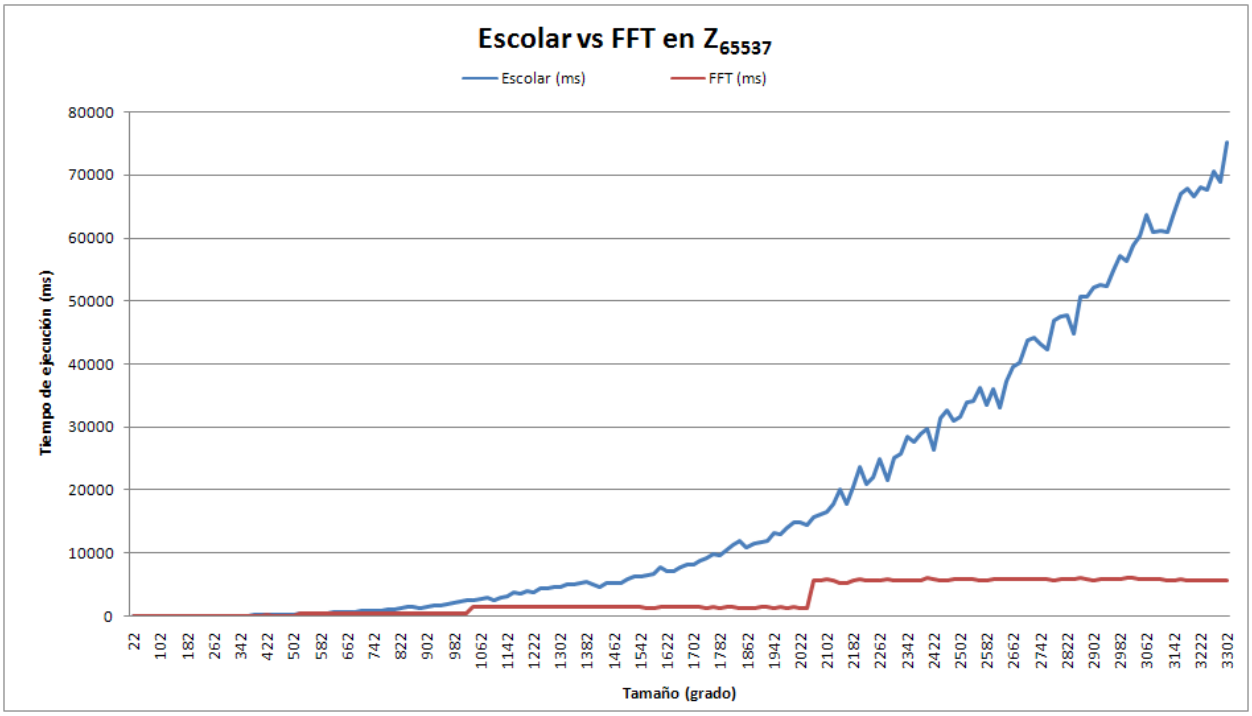


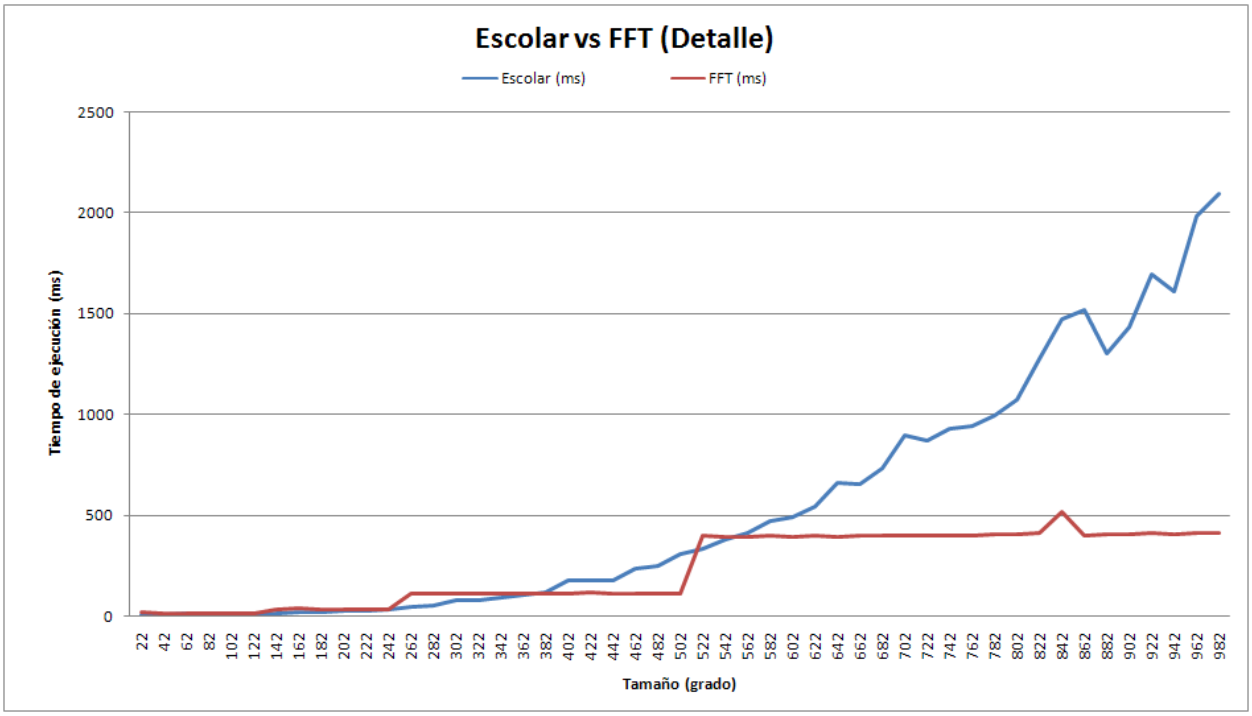
Apenas hay datos con los que trabajar. La gráfica no es representativa.

En Z_{257}:



En Z_{65537}:





Podemos ver como en Z41 y Z65537 el FFT se comporta como era de esperar. Eficiencia logarítmica y superando al escolar. Además, se convierte en superior muy pronto.

En Z65537 puede apreciarse que cerca de los 600 ya es claramente siempre mejor. Además, para el FFT apenas se producen picos, mostrando un comportamiento perfectamente determinista en cuanto al tiempo.

Manual de instalación y uso

Prerrequisitos:

- Tener instalado java jre7: http://www.oracle.com/technetwork/java/javase/downloads/java-se-jre-7-download-432155.html
- Descargar IDE eclipse Java: http://www.eclipse.org/downloads/moreinfo/java.php
- Asegurarse de que el path de donde toma el jre es correcto.
- Asegurarse de que las bibliotecas includas en "lib" se encuentran asociadas al buildpath del proyecto. En caso de que no, puede configurarse desde las propiedades el proyecto.

Datos de entrada:

- Para facilitar el uso y la comprobación de la correctitud, sin tener que establecer un formato para los datos de entrada, se ha optado por generar datos automáticamente a partir de las indicaciones dadas en el menú por el usuario. Internamente el programa **autovalida los resultados** comprobando si son iguales a los suministrados por la biblioteca profesional de cálculo simbólico.

Datos de salida:

- Las salidas se mostrarán por consola.

Proyecto: puede abrirse como un proyecto convencional. Adicionalmente puede descargarse desde el repositorio directamente.
https://github.com/Zalioth/SC_FFT

Ejecución

- Cuando se tengan los datos de entrada correctamente ubicados, bastará con colocarse encima del fichero Main.class, pulsar botón derecho/Run As/ Java Application.
- En ese momento en consola aparecerá un menú con la siguiente estructura:

  1 - C.
  2 - Z41.
  3 - Z257.
  4 - Z65537.
  5 - Salir.
  Elija una opción:

- Por consola podrá seleccionar el cuerpo.
- A continuación se le preguntará por el grado máximo de los polinomios.
- Se generarán **de forma aleatoria** dos polinomios y se les aplicará los algoritmos escolar, fft, y el suministrado por la biblioteca utilizada. Este último se utilizará para comprobar que los resultados del escolar y el fft (desarrollados por nosotros) devuelven el resultado esperado.