universidad de león

Escuela de Ingenierías
Industrial, Informática y Aeroespacial

GRADO EN INGENIERÍA INFORMÁTICA

INFORME MANDELBROT

Autor:
Guillermo Marcos García

1 de abril de 2022

Índice

Contenido

Índice 1

1. Schedule Static 2

1.1 Número de hilos (10) 2

1.2 Número de hilos (20) 2

1.3 Número de hilos (20) 3

1.4 Conclusiones 4

2. Schedule Dynamic 6

2.1 Número de hilos (10) 6

2.2 Número de hilos (20) 6

2.3 Número de hilos (20) 7

2.4 Conclusiones 8

3. Schedule Guided 9

3.1 Número de hilos (10) 9

3.2 Número de hilos (20) 9

3.3 Número de hilos (20) 10

3.4 Conclusiones 11

1.Schedule Static

Bajo la cláusula static, cada hilo obtiene aproximadamente el mismo número de iteraciones que cualquier otro hilo. Asumiendo que cada iteración requiere la misma cantidad de trabajo, todos los hilos deben finalizar aproximadamente al mismo tiempo.

1.1 Número de hilos (10)

Mediante la cláusula static y estableciendo el número de hilos a 10, realizaremos varias ejecuciones y observaremos los resultados.

Nº Ejecución	1	2	3
Tiempo	0.09707 s	0.08976 s	0.09858 s

A continuación, se verá la información en forma de gráfico.

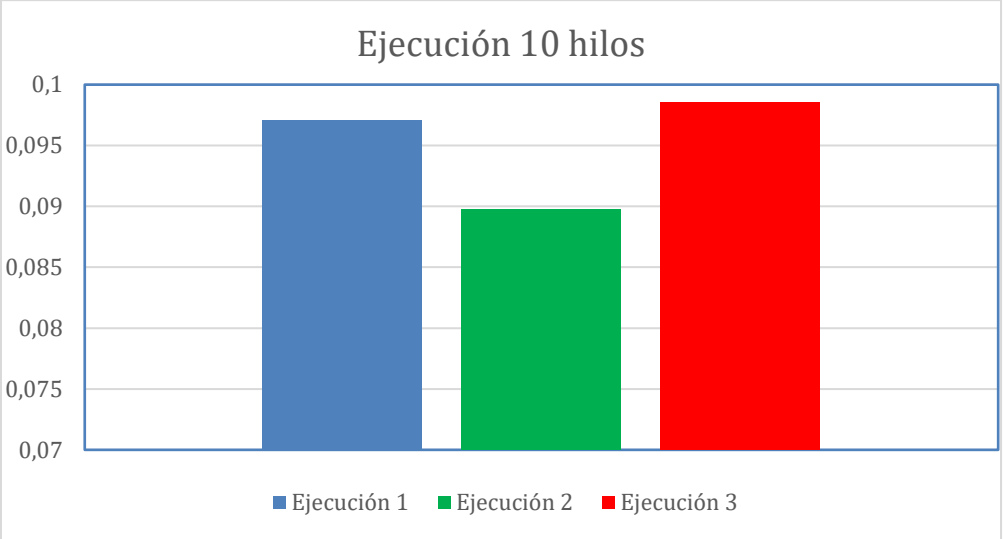

Gráfico 1.1. – Resultados de la ejecución con 10 hilos

La segunda ejecución ha sido la más rápida. Ahora probemos a cambiar el número de hilos.

1.2 Número de hilos (20)

Establecemos el número de hilos a veinte y observamos los resultados.

Nº Ejecución	1	2	3
Tiempo	0.08688 s	0.08906 s	0.09632 s

A continuación, se verá la información en forma de gráfico.

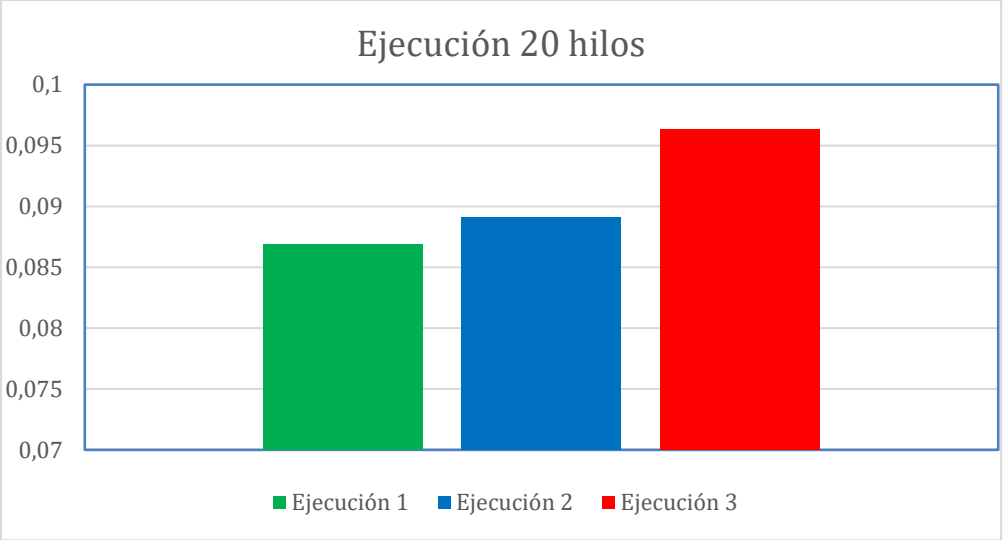

Gráfico 1.2. – Resultados de la ejecución con 20 hilos

En este caso la primera ejecución ha sido la más rápida. Aumentemos el número de hilos una última vez.

1.3 Número de hilos (20)

En este caso no vamos a variar el número de hilos, En este caso lo que va a variar es el tamaño del chunk o conjunto de datos que se envía al procesador En los casos anteriores hemos dejado el valor por defecto (1) y ahora vamos a establecerlo en cuatro. Esto significa que OpenMP va a dividir el número de iteraciones entre el tamaño del chunk y a cada hilo se le van a asignar grupos de ese tamaño de iteraciones. Por ejemplo, si el número de iteraciones es 64 y el tamaño del chunk es 4, a cada hilo se le asignarán 16 iteraciones en grupos de 4. Observemos los resultados.

Nº Ejecución	1	2	3
Tiempo	0.08812 s	0.08986 s	0.08753 s

A continuación, se verá la información en forma de gráfico.

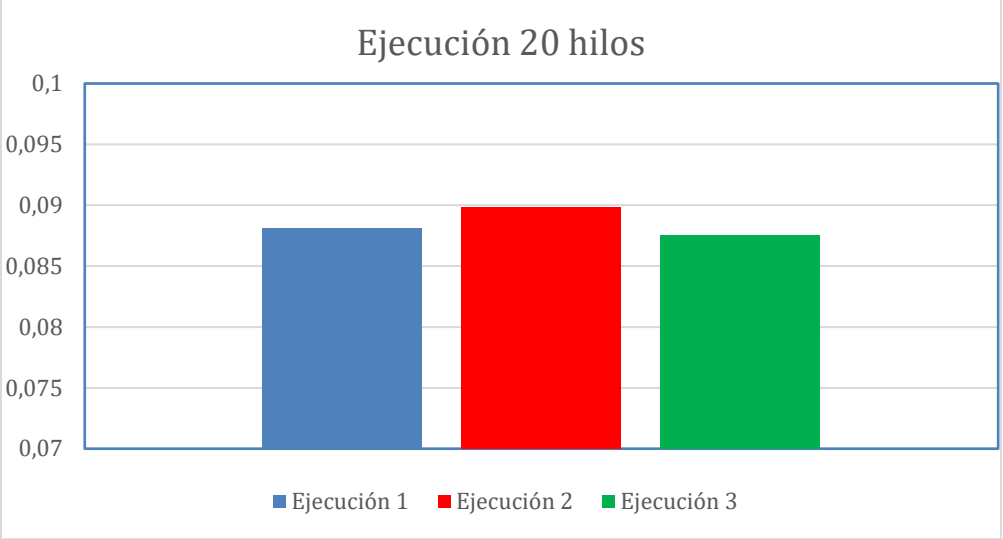

Gráfico 1.3. – Resultados de la ejecución con 30 hilos

En este caso la tercera ejecución ha sido la que ha tardado menos tiempo.

1.4 Conclusiones

Una vez obtenidos los resultados habiendo variado el número de hilos, podemos hacer un pequeño estudio sobre cuál es la opción óptima.

Nº Hilos	10	20	30
Mejor Tiempo	0.08976 s	0.08688 s	0.08753 s

A continuación, se verá la información en forma de gráfico.

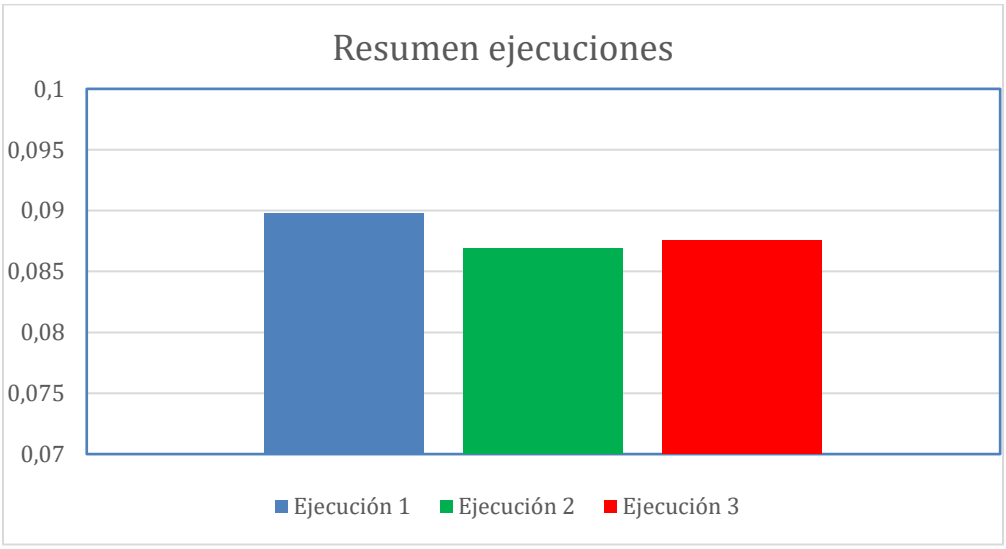

Gráfico 1.4. – Resultados de las mejores ejecuciones

El mejor resultado sale estableciendo el numero de hilos igual a veinte. Si bien las diferencias en los resultados son casi inapreciables, los resultados pueden dar a entender que a mayor número de hilos menor será el tiempo de ejecución será menor.

Pero esto no es del todo cierto ya que bajo la cláusula static, cuando un hilo termine su trabajo deberá esperar a que el resto realice su última iteración antes de poder continuar. Cuanto mayor sea el número de hilos, menor será el número de iteraciones para cada uno de ellos, pero esto no implica que el tiempo de ejecución sea menor.

2.Schedule Dynamic

La cláusula dynamic es apropiada para aquellos casos en los que cada iteración requiere una cantidad de trabajo variable o imprevisible. Al contrario que en static, los hilos no tienen por qué terminar al mismo tiempo.

A continuación, vamos a analizar los resultados que se obtienen bajo esta sentencia.

2.1 Número de hilos (10)

Establecemos el número de hilos a diez, realizamos varias ejecuciones y recogemos los resultados.

Nº Ejecución	1	2	3
Tiempo	0.09112 s	0.08683 s	0.08987 s

A continuación, se verá la información en forma de gráfico.

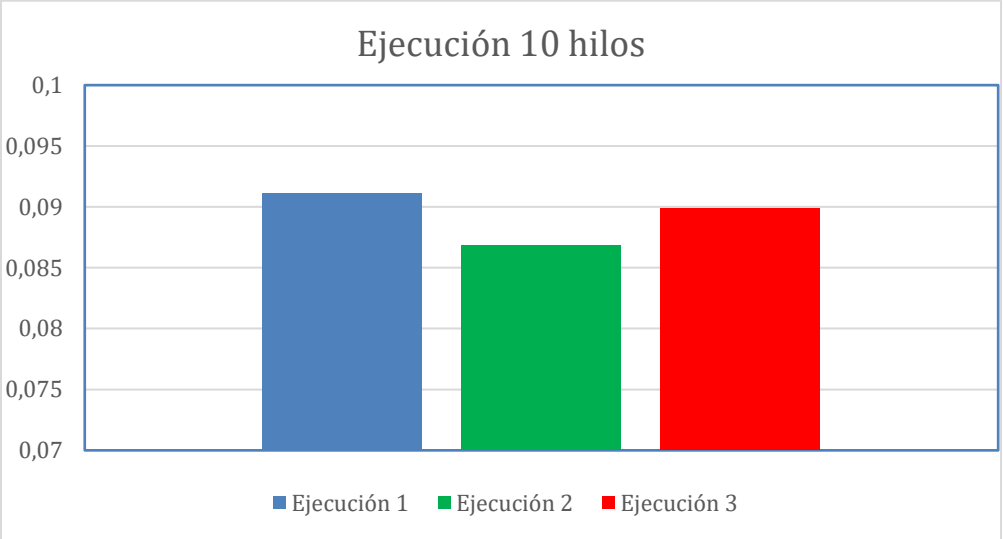

Gráfico 2.1. – Resultados de la ejecución con 10 hilos

La segunda ejecución ha sido la más rápida. Ahora probemos a cambiar el número de hilos.

2.2 Número de hilos (20)

Establecemos el número de hilos a veinte y observamos los resultados

Nº Ejecución	1	2	3
Tiempo	0.08759 s	0.08718 s	0.0874 s

.

A continuación, se verá la información en forma de gráfico.

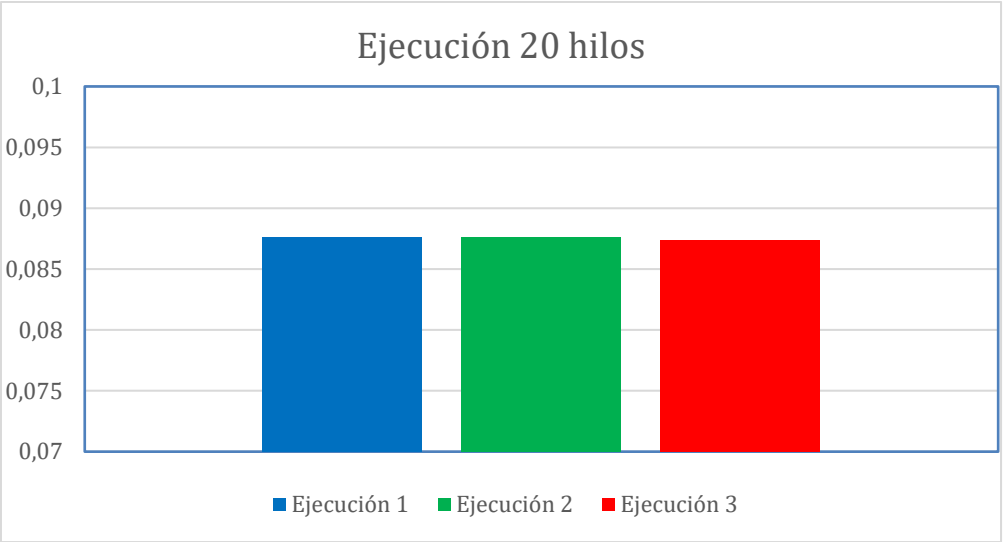

Gráfico 2.2. – Resultados de la ejecución con 20 hilos

En este caso la segunda opción vuelve a ser la óptima, pero las diferencias son apenas inapreciables. Aumentemos el número de hilos una última vez.

2.3 Número de hilos (20)

Ahora haremos como en el tercer caso de la cláusula static, vamos a mantener el número de hilos, pero variando el tamaño del chunk. Se le asignarán iteraciones a cada hilo de la misma forma que con static pero con la diferencia de que cuando un hilo termine su bloque va a pasar a ejecutar el siguiente sin esperar a que el resto hayan terminado sus respectivos bloques. Veamos los resultados.

Nº Ejecución	1	2	3
Tiempo	0.08738 s	0.08759 s	0.08993 s

A continuación, se verá la información en forma de gráfico.

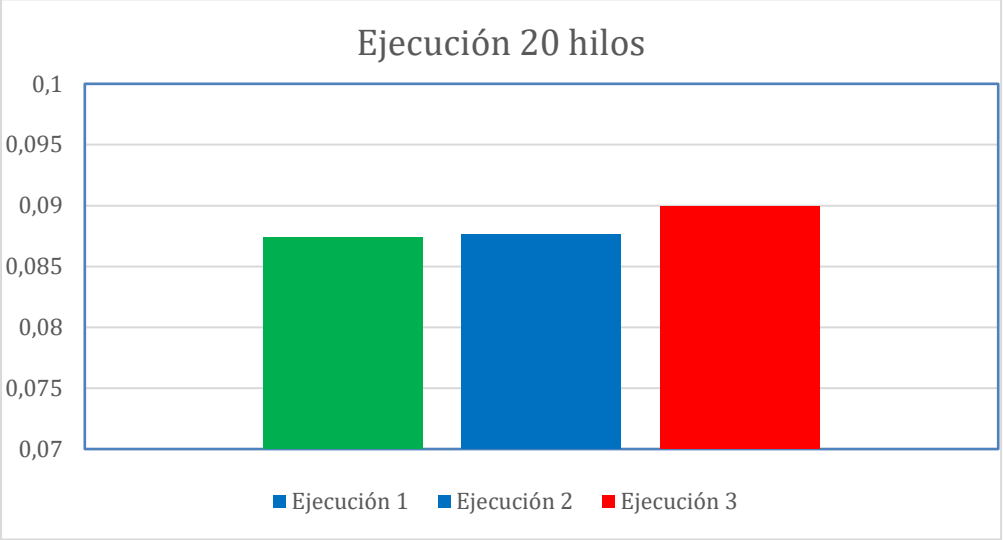

Gráfico 2.3. – Resultados de la ejecución con 30 hilos

La primera ejecución ha sido la más rápida.

2.4 Conclusiones

Una vez obtenidos los resultados habiendo variado el número de hilos, podemos hacer un pequeño estudio sobre cuál es la opción óptima.

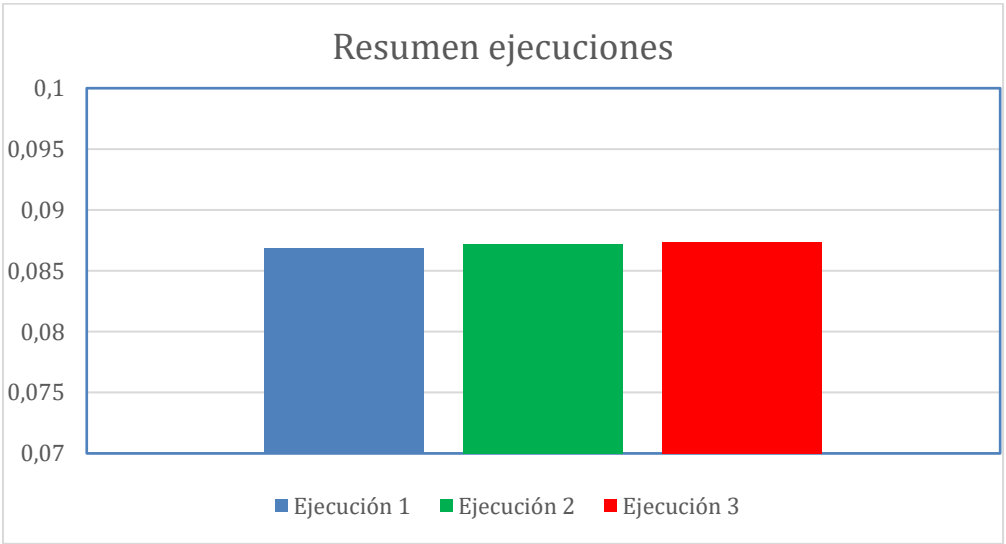

Gráfico 2.4. – Resultados de las mejores ejecuciones

El mejor resultado sale estableciendo el número de hilos igual a diez. Las diferencias entre los tiempos de ejecución son mínimas y cabe destacar que la cláusula dynamic se caracteriza por la propiedad de que ningún hilo va a quedarse esperando a que terminen el resto de los hilos. Cada uno seguirá su camino sin importar el estado en el que se

encuentre el resto. Esto significa que las iteraciones se han de asignar una a una a los hilos a medida que estos queden disponibles.

3.Schedule Guided

La cláusula guided es muy similar a dynamic. Ambas cumplen la característica de que los hilos siguen su recorrido sin esperar al resto. Sin embargo, existen unas pequeñas diferencias que iremos viendo con las ejecuciones.

3.1 Número de hilos (10)

Establecemos el número de hilos a diez, realizamos varias ejecuciones y recogemos los resultados.

Nº Ejecución	1	2	3
Tiempo	0,08776 s	0,08804 s	0,08785 s

A continuación, se verá la información en forma de gráfico.

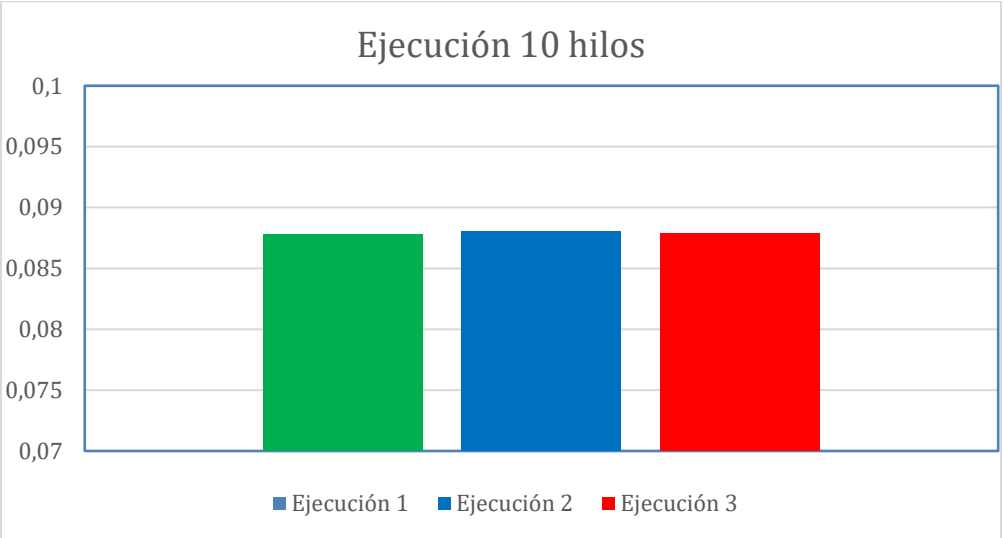

Gráfico 3.1. – Resultados de la ejecución con 10 hilos

La primera ejecución ha sido la más rápida. Aumentemos el número de hilos.

3.2 Número de hilos (20)

Establecemos el número de hilos a veinte y observamos los resultados

Nº Ejecución	1	2	3
Tiempo	0,08779 s	0,08763 s	0,08918 s

Veámoslo en forma de gráfico.

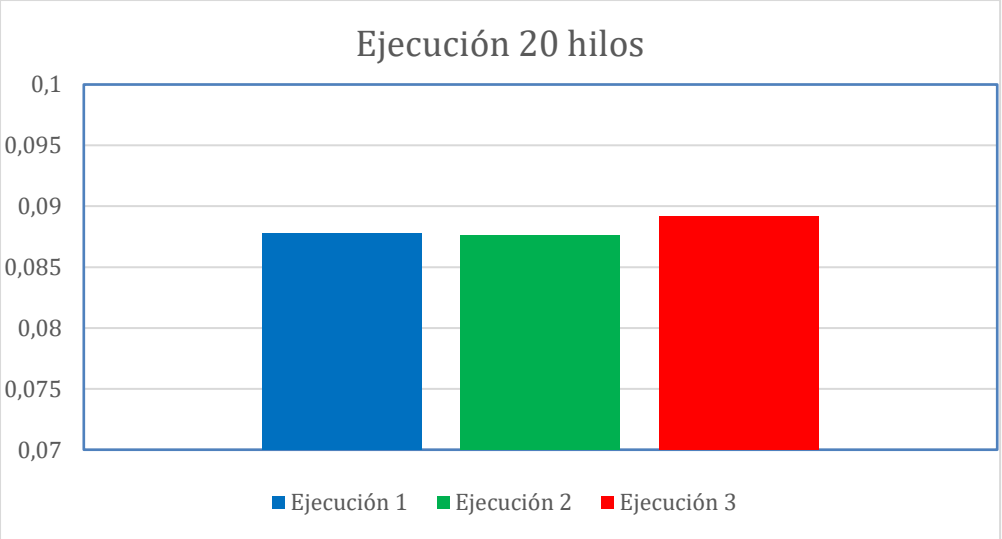

Gráfico 3.2. – Resultados de la ejecución con 20 hilos

La segunda ejecución ha sido la más rápida.

3.3 Número de hilos (20)

Establecemos el número de hilos a veinte y el tamaño del chunk a cuatro.

Nº Ejecución	1	2	3
Tiempo	0,08788 s	0,09378 s	0,09342 s

Veámoslo en forma de gráfico.

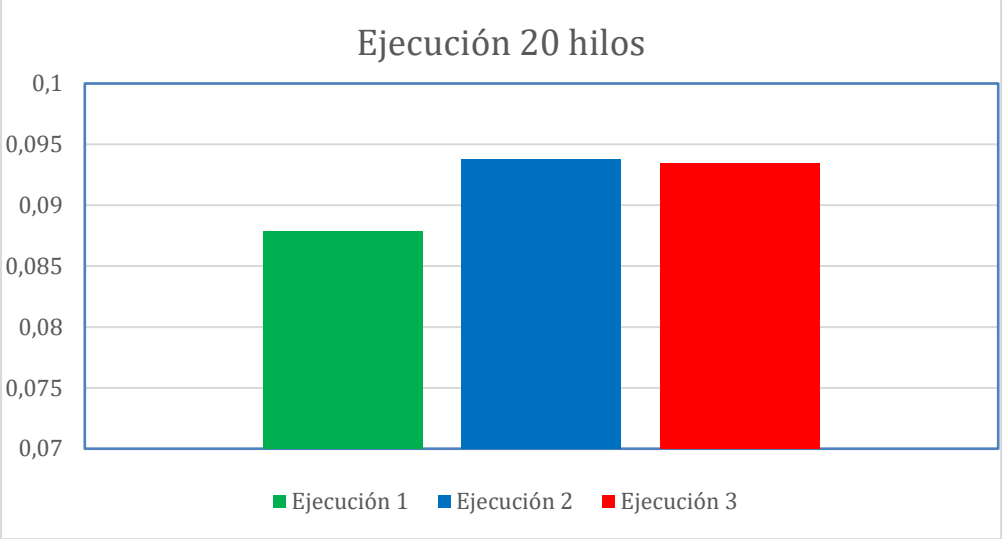

Gráfico 3.3. – Resultados de la ejecución con 20 hilos

La primera ejecución ha sido la más rápida.

3.4 Conclusiones

Los resultados son muy similares en la mayoría de los casos. Las diferencias entre las cláusulas dynamic y guided se basan en el tamaño del chunk. En la primera, el subproceso va realizando grupos de iteraciones del tamaño del chunk sin esperar al resto. Esto también se cumple bajo la sentencia guided, pero aquí el tamaño del chunk no es fijo. En este caso el tamaño del fragmento es proporcional al número de iteraciones sin asignar. Esto significa que en las últimas iteraciones puede darse el caso de que el bloque de iteraciones sea menor que el tamaño del fragmento.