universidad de león

Escuela de Ingenierías

Informe ejercicio 2

Computación GRID y Supercomputación

Alumno:

Sergio Chimeno Alegre

Grado en ingeniería informática

Índice

Comentarios

Para tomar el tiempo se realizarán 3 ejecuciones, y se tomará como tiempo la media de los tiempos.

La cantidad de hilos la especificaré con la variable de entorno OMP_NUM_THREADS

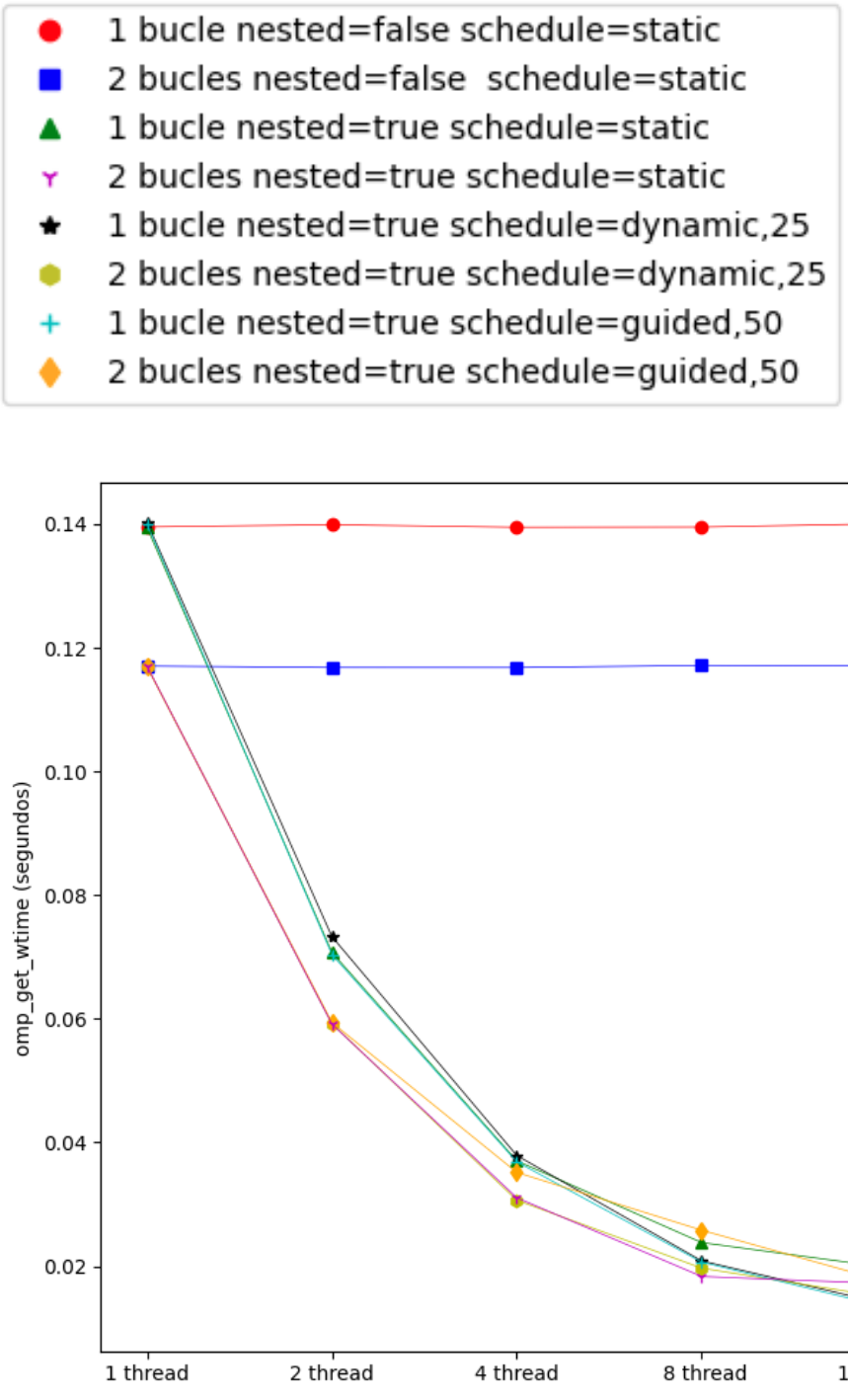
La clasula schedule está especificada como runtime, para establecerla con la variable de entorno OMP_SCHEDULE y no tener que recompilar cada vez que se quiera probar una nueva opción.

Puesto que con OMP_NESTED=FALSE solo se paraleliza el primer parallel, solo lo he probado con OMP_SCHEDULE=static, ya que en ese caso da lo mismo un schedule que otro.

Comparación de las distintas versiones

1 bucle nested=false schedule=static						
	1 thread	2 thread	4 thread	8 thread	16 thread	32 thread
omp_get_wtime	0.139543	0.13989	0.13948	0.139518	0.140048	0.140159
2 bucles nested=false schedule=static						
	1 thread	2 thread	4 thread	8 thread	16 thread	32 thread
omp_get_wtime	0.117042	0.116806	0.116799	0.117149	0.117049	0.118385
1 bucle nested=true schedule=static						
	1 thread	2 thread	4 thread	8 thread	16 thread	32 thread
omp_get_wtime	0.139447	0.070722	0.037031	0.023802	0.019782	0.017414
2 bucles nested=true schedule=static						
	1 thread	2 thread	4 thread	8 thread	16 thread	32 thread
omp_get_wtime	0.117019	0.059141	0.031056	0.018327	0.017162	0.01533
1 bucle nested=true schedule=dynamic,25						
	1 thread	2 thread	4 thread	8 thread	16 thread	32 thread
omp_get_wtime	0.140289	0.073291	0.037891	0.02085	0.013823	0.01265
2 bucles nested=true schedule=dynamic,25						
	1 thread	2 thread	4 thread	8 thread	16 thread	32 thread
omp_get_wtime	0.116946	0.059225	0.030658	0.019674	0.014895	0.015853
1 bucle nested=true schedule=guided,50						
	1 thread	2 thread	4 thread	8 thread	16 thread	32 thread
omp_get_wtime	0.14003	0.070389	0.036864	0.020652	0.013331	0.014324

2 bucles nested=true schedule=guided,50						
	1 thread	2 thread	4 thread	8 thread	16 thread	32 thread
omp_get_wtime	0.116923	0.059397	0.035154	0.025804	0.017466	0.015222



Las versiones con OMP_NESTED=FALSE tienen el mismo tiempo de ejecución independientemente del número de hilos. Esto es así al no poder ejecutar paralelamente las operaciones de matrices. No se aprecia tampoco el cambio de 1 hilo a 2, ya que, aunque la suma y la multiplicación puedan ejecutarse paralelamente, la suma toma un tiempo muy pequeño en comparación (hace 500*500 sumas), mientras que la multiplicación hace 500*500*500 sumas y multiplicaciones.

Vemos que en todos los casos, segun vamos aumentando el número de threads, el tiempo total disminuye.

De 1 a 8 threads el tiempo se divide por 2 cada vez que multiplicamos por 2 el número de hilos. De 8 a 16 threads varía según el caso. Y de 16 a 32 threads apenas hay diferencia, debido a que calendula solo me proveyó de 16 threads físicos.

Para pocos threads vemos que es mas eficiente la versión de 2 bucles que la de 1 bucle, pero a medida que se aumenta el número de threads, el tiempo parece converger.

La diferencia entre los valores de schedule es dificil de percibir, porque aunque hay alguna es mínima, dependiente del número de threads, y ademas se podría deber a la variabilidad.

Aunque no he tomado ejemplos, en caso de usar tamaños de bloque para schedule muy pequeños suele dar peores resultados debido a que se pierde la localidad en el acceso a memoria.