Multiplicación de matrices de forma secuencial y paralela

Presentado por:
Santiago Sosa Herrera

Presentado a:
Ramiro Andrés Barrios Valencia

HPC: High Performance Computing
Ingeniería de Sistemas y Computación
Universidad Tecnológica de Pereira
2023

Tabla de contenido

Resumen

En este trabajo se aborda el problema de la multiplicación de matrices en C, utilizando diferentes técnicas para mejorar su rendimiento. Primero, se implementó la solución secuencial y se midió su tiempo de ejecución para matrices de diferentes tamaños. Luego, se utilizó la programación con hilos para paralelizar el cálculo y se comparó su desempeño con el método secuencial.

Además, se exploró la programación con procesos como otra alternativa de paralelización y se midió su tiempo de ejecución. También se aplicó la optimización del código mediante el uso de la consola de comandos para compilar el programa y se midió el tiempo de ejecución de esta versión optimizada.

Finalmente, se implementó la transposición de la matriz para reducir los accesos a memoria y mejorar el rendimiento. Se midió el tiempo de ejecución de esta solución y se comparó con las otras técnicas implementadas. Los resultados obtenidos indican que la multiplicación de matrices utilizando hilos y la optimización por consola son las técnicas más eficientes para este problema.

Introducción

La multiplicación de matrices es una operación fundamental en la computación científica y el procesamiento de datos. Esta operación se utiliza en una amplia gama de aplicaciones, incluyendo el análisis de datos, el aprendizaje automático, la simulación de sistemas físicos, entre otros. Sin embargo, la multiplicación de matrices puede ser computacionalmente costosa y presenta varios desafíos en cuanto a la complejidad del algoritmo y la optimización de código.

Uno de los principales problemas asociados con la multiplicación de matrices es su complejidad computacional, que crece de manera exponencial con el tamaño de las matrices. Esto puede limitar el uso de la multiplicación de matrices en aplicaciones en las que se requiere procesamiento rápido de grandes conjuntos de datos. Para hacer frente a este problema, se han desarrollado diversos enfoques y técnicas de optimización de código, como la utilización de algoritmos paralelos y la explotación de la memoria caché.

La programación paralela es un enfoque comúnmente utilizado para optimizar la multiplicación de matrices. Esto implica dividir la tarea de multiplicación de matrices en tareas más pequeñas que se ejecutan simultáneamente en diferentes hilos o procesos. Esta técnica puede mejorar significativamente el rendimiento y la eficiencia del código.

Además, también se han utilizado técnicas como la matriz transpuesta y la optimización de compilación para mejorar el rendimiento del código. La matriz transpuesta permite aprovechar la localidad de la memoria caché, reduciendo el tiempo de acceso a la memoria y mejorando así el rendimiento. La optimización de compilación, por su parte, implica la utilización de técnicas de optimización de código en el proceso de compilación, mejorando así el rendimiento y la eficiencia del código generado.

En este trabajo se presentan los resultados de un estudio en el que se compararon diferentes enfoques para la multiplicación de matrices en el lenguaje de programación C. Se evaluó el rendimiento del método secuencial, por hilos, por procesos, optimización por consola, y matriz transpuesta. Los resultados muestran que el enfoque de

programación paralela utilizando hilos y la matriz transpuesta son altamente eficientes en términos de tiempo de ejecución y rendimiento en comparación con otros enfoques.

Marco conceptual

High Performance Computing: High Performance Computing es un área de la informática que se dedica a crear sistemas informáticos y software que puedan manejar tareas de alta complejidad y procesamiento de grandes volúmenes de datos en un período de tiempo muy corto.

Multiplicación matricial: La multiplicación matricial es una operación que se realiza entre dos matrices y que arroja como resultado una tercera matriz. Esta operación es muy importante en distintos campos de la ciencia y la ingeniería, y se utiliza en problemas lineales y de optimización.

Complejidad Computacional: La complejidad computacional es una forma de medir los recursos que se necesitan para resolver un problema computacional. Sirve para determinar la dificultad de un problema y para comparar la eficiencia de distintos algoritmos.

Programación paralela: La programación paralela es una técnica de programación que se basa en dividir un problema en tareas más pequeñas y ejecutarlas en múltiples procesadores al mismo tiempo. Esta técnica se utiliza para mejorar la eficiencia de los sistemas informáticos y acelerar la resolución de problemas.

Hilos: Los hilos son una forma de dividir la ejecución de un programa en unidades más pequeñas y manejables, lo que permite que varias tareas se realicen simultáneamente. Cada hilo funciona de manera independiente, con su propia pila de memoria y compartiendo recursos con otros hilos dentro del mismo proceso. El uso de hilos puede mejorar significativamente la eficiencia y velocidad de ejecución de una aplicación.

Speedup: El speedup se utiliza para medir la mejora en la velocidad de ejecución de un programa cuando se ejecuta en un sistema de cómputo más rápido o en paralelo. Se mide en términos de una relación entre el tiempo que tarda un programa en ejecutarse en un sistema de cómputo determinado y el tiempo que tarda en ejecutarse en un sistema más rápido. Cuanto mayor sea el speedup, mayor será la mejora en el rendimiento.

Marco Contextual

Características de la maquina

Las características del pc donde se realizaron las pruebas son los siguientes:

- Procesador AMD A8-7410 apu with amd Radeon r5 graphics x 4
- Memoria 6,7 GiB
- SO 64 bits
- Disco de 295 GB
- Sistema operativo Linux Mint 19
    - Gnome 3.28.2

Desarrollo

Para el inicio del trabajo se empezó desarrollando la multiplicación de matrices de forma normal, es decir secuencial, este código tenia que recibir por parámetro en consola el tamaño de las matrices, que eran cuadradas, y en el caso de hilos y procesos otro para el numero de estos, para el código de los otros programas básicamente se modificaba el secuencial para adaptarlos a cada caso, además, se le agrego a cada uno una forma de medir el tiempo en que demoraba el programa en ejecutar todo, con ayuda de la librería time.h.

Posteriormente con ayuda de un script se ejecutaban estos programas una cantidad de veces determinada para cada tamaño de las matrices calculadas, las cuales fueron 10, 100, 200, 400, 800, 1600 y 3.200.

En la multiplicación de matrices con hilos se hizo uso de la librería de C, pthread para el manejo de hilos.

Entendiendo como funciona la multiplicación con hilos básicamente se puede entender como funcionan los otros métodos, para cada hilo procesa un rango de filas de la matriz resultado C. El rango se calcula dividiendo el número total de filas por el número total de hilos. Cada hilo también recibe como argumentos, las matrices A y B, así como los tamaños de las matrices y el número total de hilos.

Después de crear los hilos, se espera a que terminen usando la función pthread_join(). Una vez que todos los hilos han terminado, se puede imprimir la matriz resultado, o terminar midiendo el tiempo que demoro en ejecutarse.

Cabe señalar que esta implementación divide las filas de la matriz C entre los hilos, pero también se podrían dividir las columnas o incluso bloques de la matriz C para obtener un mejor rendimiento. Además, para matrices muy grandes, se podría utilizar una técnica de división recursiva en la que cada hilo multiplica submatrices más pequeñas para aprovechar mejor el paralelismo.

Pruebas

1. Resultados de ejecutar el algoritmo en forma secuencial:

tiempo\matriz	10	100	200	400	600	800	1000	2000
Tiempo transcurrido:	0,000112	0,014843	0,098059	0,888076	5,552384	15,000685	31,377433	266,019738
Tiempo transcurrido:	0,000204	0,016888	0,093718	0,877490	5,719945	15,002777	31,093422	264,814952
Tiempo transcurrido:	0,000205	0,015730	0,094430	0,871610	5,388444	14,974775	31,121343	265,222653
Tiempo transcurrido:	0,000239	0,013825	0,095561	0,871919	5,442154	14,970025	31,072066	264,939481
Tiempo transcurrido:	0,000095	0,020637	0,094184	0,865209	5,373655	14,995896	31,147720	264,807805
Tiempo transcurrido:	0,000090	0,011772	0,094802	0,886910	5,498816	14,992217	31,262192	265,177446
Tiempo transcurrido:	0,000210	0,013281	0,094455	0,869835	5,413974	14,988915	31,231712	265,524536
Tiempo transcurrido:	0,000088	0,011850	0,095684	0,869758	5,405491	14,939653	31,977184	265,122534
Tiempo transcurrido:	0,000216	0,016110	0,094074	0,866782	5,402242	14,905383	31,795027	264,827617
Tiempo transcurrido:	0,000210	0,018044	0,094225	0,861536	5,395378	15,521809	31,814963	275,715047
Promedio	0,0001669	0,015298	0,0949192	0,8729125	5,4592483	15,0292135	31,3893062	266,2171809

2. Resultado de ejecutar el algoritmo con 2 hilos:

tiempo\matriz	10	100	200	400	600	800	1000	2000
Tiempo transcurrido:	0,000709	0,017464	0,125778	1,033544	6,423726	16,628052	36,349787	296,779504
Tiempo transcurrido:	0,000950	0,016912	0,132485	1,033160	6,197521	15,973700	32,726146	285,298079
Tiempo transcurrido:	0,000853	0,017015	0,128233	1,036969	6,186428	15,875732	32,515568	277,336486
Tiempo transcurrido:	0,000802	0,025096	0,124889	1,028659	6,782820	15,957374	32,440586	278,268634
Tiempo transcurrido:	0,001075	0,023157	0,129226	1,033115	6,172454	15,915244	32,519389	277,444121
Tiempo transcurrido:	0,001148	0,017002	0,136628	1,052246	6,187337	15,866347	32,533681	293,540716
Tiempo transcurrido:	0,001121	0,025109	0,127673	1,099258	6,234362	16,008939	32,432589	276,682155
Tiempo transcurrido:	0,000730	0,024534	0,125077	1,031679	6,172880	15,843013	32,445940	277,891267
Tiempo transcurrido:	0,000818	0,019286	0,124015	1,031721	6,192647	15,900849	32,456881	277,402064
Tiempo transcurrido:	0,001101	0,023799	0,124134	1,028148	6,154579	15,906563	32,433061	276,884794
Promedio	0,0009307	0,0209374	0,1278138	1,0408499	6,2704754	15,9875813	32,8853628	281,752782
speedup	0,17932739	0,73065424	0,74263655	0,83865358	0,8706275	0,94005549	0,95450692	0,944860878

3. Resultado de ejecutar el algoritmo con 4 hilos:

tiempo\matriz	10	100	200	400	600	800	1000	2000
Tiempo transcurrido:	0,001139	0,020479	0,129326	1,155768	6,867144	18,150768	37,041578	315,992051
Tiempo transcurrido:	0,001236	0,025422	0,130410	1,151716	6,815911	18,267974	37,040825	319,276810
Tiempo transcurrido:	0,000889	0,020659	0,131064	1,157434	6,873215	18,217650	37,028847	319,241700
Tiempo transcurrido:	0,001249	0,018387	0,131548	1,162087	6,897173	18,107880	37,046521	320,050825
Tiempo transcurrido:	0,001211	0,025689	0,126868	1,160085	6,825306	18,295248	37,342997	319,703308
Tiempo transcurrido:	0,001290	0,020450	0,132150	1,163357	6,858796	18,270532	37,319737	318,998588
Tiempo transcurrido:	0,001520	0,025146	0,127859	1,160428	6,788339	18,223719	37,281823	317,784114
Tiempo transcurrido:	0,001041	0,018001	0,136259	1,157861	6,751700	18,150283	36,847563	320,513042
Tiempo transcurrido:	0,001000	0,023841	0,126262	1,159708	6,821286	18,233298	36,955950	315,508297
Tiempo transcurrido:	0,001342	0,020126	0,134370	1,194696	6,830868	18,299677	37,155541	321,034431
Promedio	0,0011917	0,02182	0,1306116	1,162314	6,8329738	18,2217029	37,1061382	318,8103166
speedup	0,14005203	0,70109991	0,72672871	0,75101264	0,79895642	0,82479742	0,84593298	0,835033144

4. Resultado de ejecutar el algoritmo con 8 hilos:

tiempo\matriz	10	100	200	400	600	800	1000	2000
Tiempo transcurrido:	0,001921	0,021236	0,131148	1,159726	6,707492	17,657901	36,687143	336,523824
Tiempo transcurrido:	0,001764	0,020900	0,130653	1,156866	6,727012	17,725773	37,521637	336,654261
Tiempo transcurrido:	0,002022	0,022253	0,126962	1,169180	6,703998	17,957113	37,462457	339,010547
Tiempo transcurrido:	0,001822	0,024246	0,126576	1,165729	6,725124	17,971105	37,390412	334,210517
Tiempo transcurrido:	0,002223	0,024669	0,131865	1,195519	6,691962	17,707515	37,150439	336,688182
Tiempo transcurrido:	0,002081	0,021546	0,130335	1,162736	6,729389	17,793205	37,168015	338,005241
Tiempo transcurrido:	0,001703	0,018834	0,130116	1,153161	6,716366	18,213667	37,624068	339,827326
Tiempo transcurrido:	0,001600	0,019552	0,125509	1,167360	6,745344	18,101263	38,747480	338,924336
Tiempo transcurrido:	0,001416	0,015558	0,128929	1,183119	6,791207	18,239481	38,425846	345,331175
Tiempo transcurrido:	0,001655	0,020456	0,132578	1,173142	6,783369	18,462749	38,373886	340,082292
Promedio	0,0018207	0,020925	0,1294671	1,1686538	6,7321263	17,9829772	37,6551383	338,5257701
speedup	0,09166804	0,73108722	0,73315306	0,74693849	0,81092482	0,83574668	0,83359955	0,786401522

5. Resultado de ejecutar el algoritmo con 16 hilos:

tiempo\matriz	10	100	200	400	600	800	1000	2000
Tiempo transcurrido:	0,002068	0,018441	0,126692	1,171259	6,667770	18,239897	37,889030	376,134016
Tiempo transcurrido:	0,002745	0,018866	0,132047	1,283717	7,284910	19,372397	39,914063	378,641970
Tiempo transcurrido:	0,003684	0,019126	0,131929	1,318028	7,450903	19,794836	40,772982	374,314949
Tiempo transcurrido:	0,003019	0,017759	0,132142	1,278892	7,352293	19,384263	39,620225	360,780308
Tiempo transcurrido:	0,001371	0,017585	0,129356	1,318541	7,329991	19,346170	39,963410	357,885004
Tiempo transcurrido:	0,002295	0,018238	0,131213	1,239100	7,311337	19,084670	39,427284	352,198454
Tiempo transcurrido:	0,002035	0,017853	0,129853	1,232047	7,148703	19,227051	39,221607	349,855373
Tiempo transcurrido:	0,002170	0,018067	0,130741	1,244318	7,185803	19,217199	39,150757	350,124735
Tiempo transcurrido:	0,002694	0,017961	0,133043	1,241046	7,228599	19,072587	39,012785	350,288621
Tiempo transcurrido:	0,002720	0,018148	0,131104	1,238364	7,201756	19,163311	39,147825	350,430271
Promedio	0,0024801	0,0182044	0,130812	1,2565312	7,2162065	19,1902381	39,4119968	360,0653701
speedup	0,06729567	0,84034629	0,72561539	0,69470022	0,75652606	0,78316973	0,79644039	0,739357914

6. Resultado de ejecutar el algoritmo con 32 hilos:

tiempo\matriz	10	100	200	400	600	800	1000	2000
Tiempo transcurrido:	0,004100	0,019823	0,132199	1,183707	7,016701	19,117129	39,227038	349,625937
Tiempo transcurrido:	0,003326	0,019927	0,132557	1,200787	7,020638	19,040254	39,217959	348,804531
Tiempo transcurrido:	0,004032	0,019306	0,132714	1,184925	6,981069	19,047399	39,260642	351,502683
Tiempo transcurrido:	0,003472	0,020203	0,133567	1,154502	6,854435	18,873613	39,173794	364,294019
Tiempo transcurrido:	0,003785	0,020419	0,131637	1,149742	6,761599	18,850749	39,222327	347,170011
Tiempo transcurrido:	0,002944	0,018824	0,131493	1,119006	6,963014	19,088307	39,035648	346,463762
Tiempo transcurrido:	0,004342	0,022055	0,133268	1,172042	6,948171	18,966107	39,187647	346,028272
Tiempo transcurrido:	0,004390	0,021174	0,133281	1,171294	6,942332	18,800211	39,021650	346,590376
Tiempo transcurrido:	0,004136	0,022167	0,132682	1,174626	6,958309	18,958815	39,121248	347,117564
Tiempo transcurrido:	0,004033	0,022238	0,133457	1,168929	6,854764	19,026562	39,105772	346,668133
Promedio	0,003856	0,0206136	0,1326855	1,167956	6,9301032	18,9769146	39,1573725	349,4265288
speedup	0,0432832	0,74213141	0,7153698	0,74738475	0,78775859	0,7919735	0,80161932	0,761868831

7. Resultado de ejecutar el algoritmo con 2 procesos:

tiempo\matriz	10	100	200	400	600	800	1000	2000
Tiempo transcurrido:	0,000370	0,001208	0,003511	0,012140	0,026325	0,045950	0,071824	0,278613
Tiempo transcurrido:	0,000374	0,001221	0,003497	0,011987	0,026471	0,046724	0,072314	0,295028
Tiempo transcurrido:	0,000340	0,001151	0,003345	0,012116	0,026547	0,048269	0,073111	0,280174
Tiempo transcurrido:	0,000328	0,001071	0,003394	0,011992	0,026498	0,046831	0,072418	0,276730
Tiempo transcurrido:	0,000363	0,001124	0,003363	0,012072	0,026383	0,046119	0,071762	0,277836
Tiempo transcurrido:	0,000335	0,001094	0,003315	0,012052	0,026350	0,046086	0,071602	0,276818
Tiempo transcurrido:	0,000327	0,001140	0,003355	0,011885	0,026356	0,046005	0,071510	0,276495
Tiempo transcurrido:	0,000412	0,001089	0,003384	0,012055	0,026397	0,048313	0,071406	0,273978
Tiempo transcurrido:	0,000335	0,001054	0,003284	0,012110	0,026355	0,046217	0,071593	0,273896
Tiempo transcurrido:	0,000345	0,001123	0,003380	0,012071	0,026347	0,046187	0,071684	0,276982
Promedio	0,0003529	0,0011275	0,0033828	0,012048	0,0264029	0,0466701	0,0719224	0,278655
speedup	0,47293851	13,568071	28,0593591	72,4528967	206,766995	322,030883	436,432964	955,364809

8. Resultado de ejecutar el algoritmo con 4 procesos:

tiempo\matriz	10	100	200	400	600	800	1000	2000
Tiempo transcurrido:	0,000477	0,001245	0,003565	0,012266	0,026441	0,046337	0,072306	0,278541
Tiempo transcurrido:	0,000467	0,001183	0,003455	0,012013	0,026699	0,046284	0,071867	0,277908
Tiempo transcurrido:	0,000462	0,001181	0,003543	0,012076	0,026385	0,046315	0,070853	0,279523
Tiempo transcurrido:	0,000455	0,001194	0,003503	0,012258	0,026724	0,046194	0,071759	0,286039
Tiempo transcurrido:	0,000480	0,001200	0,003456	0,012290	0,026916	0,046433	0,071670	0,277054
Tiempo transcurrido:	0,000504	0,001273	0,003500	0,012342	0,026706	0,046406	0,071741	0,277222
Tiempo transcurrido:	0,000508	0,001246	0,003405	0,012311	0,026394	0,046323	0,071957	0,280613
Tiempo transcurrido:	0,000548	0,001250	0,003594	0,012486	0,026541	0,046264	0,072010	0,277908
Tiempo transcurrido:	0,000471	0,001213	0,003639	0,012084	0,026573	0,046413	0,071817	0,286564
Tiempo transcurrido:	0,000521	0,001291	0,003478	0,012579	0,026554	0,046038	0,072047	0,278317
Promedio	0,0004893	0,0012276	0,0035138	0,0122705	0,0265933	0,0463007	0,0718027	0,2799689
speedup	0,34109953	12,4617139	27,013262	71,1391141	205,286606	324,600136	437,160527	950,881262

9. Resultado de ejecutar el algoritmo con 8 procesos:

tiempo\matriz	10	100	200	400	600	800	1000	2000
Tiempo transcurrido:	0,000747	0,001527	0,004235	0,012737	0,027108	0,047338	0,073156	0,292343
Tiempo transcurrido:	0,000668	0,001375	0,003952	0,011025	0,027536	0,042291	0,064258	0,252196
Tiempo transcurrido:	0,000935	0,001923	0,004475	0,013226	0,029795	0,045497	0,066879	0,254428
Tiempo transcurrido:	0,000664	0,002031	0,004514	0,014159	0,027683	0,044390	0,069274	0,247959
Tiempo transcurrido:	0,000651	0,001374	0,004637	0,013177	0,027724	0,044811	0,067023	0,254650
Tiempo transcurrido:	0,000900	0,001966	0,004685	0,013515	0,027125	0,045295	0,065203	0,250138
Tiempo transcurrido:	0,000945	0,002524	0,003956	0,013472	0,029877	0,047368	0,067222	0,255497
Tiempo transcurrido:	0,000888	0,001477	0,003869	0,012534	0,025341	0,046815	0,065078	0,257173
Tiempo transcurrido:	0,000685	0,001461	0,004155	0,011629	0,028065	0,042704	0,067172	0,252121
Tiempo transcurrido:	0,000926	0,001753	0,004434	0,011664	0,023802	0,046038	0,068439	0,260033
Promedio	0,0008009	0,0017411	0,0042912	0,0127138	0,0274056	0,0452547	0,0673704	0,2576538
speedup	0,20839056	8,7863994	22,1195004	68,6586622	199,201926	332,10282	465,921327	1033,236

10. Resultado de ejecutar el algoritmo con 16 procesos:

tiempo\matriz	10	100	200	400	600	800	1000	2000
Tiempo transcurrido:	0,002054	0,003370	0,004502	0,013815	0,029029	0,046839	0,069901	0,267410
Tiempo transcurrido:	0,001149	0,003115	0,004573	0,014088	0,026906	0,046208	0,068070	0,265902
Tiempo transcurrido:	0,001148	0,001938	0,004568	0,013654	0,028686	0,045295	0,068481	0,267805
Tiempo transcurrido:	0,001136	0,002989	0,005267	0,016931	0,028096	0,047553	0,068452	0,266462
Tiempo transcurrido:	0,001161	0,002005	0,004924	0,013729	0,027903	0,047879	0,071617	0,267662
Tiempo transcurrido:	0,001263	0,002200	0,004971	0,016027	0,028692	0,048075	0,068996	0,266256
Tiempo transcurrido:	0,001169	0,003022	0,004932	0,014713	0,029545	0,045707	0,070188	0,267936
Tiempo transcurrido:	0,001139	0,001955	0,004393	0,015031	0,027828	0,050235	0,071072	0,267945
Tiempo transcurrido:	0,001343	0,002108	0,004609	0,016229	0,030607	0,048364	0,067432	0,266927
Tiempo transcurrido:	0,001161	0,002705	0,005209	0,014889	0,028102	0,043785	0,075668	0,276816
Promedio	0,0012723	0,0025407	0,0047948	0,0149106	0,0285394	0,046994	0,0699877	0,2681121
speedup	0,13117975	6,02117527	19,7962793	58,5430834	191,288124	319,811327	448,497467	992,932363

11. Resultado de ejecutar el algoritmo con “la matriz transpuesta” el cual consiste en multiplicar línea por línea

tiempo\matriz	10	100	200	400	600	800	1000	2000
Tiempo transcurrido:	0,001251	0,018245	0,139194	1,335608	7,625626	19,357562	39,900332	357,074447
Tiempo transcurrido:	0,001251	0,017623	0,136773	1,296270	7,431005	19,349726	39,813752	353,028457
Tiempo transcurrido:	0,000796	0,017762	0,135493	1,261941	7,249602	18,921681	39,296889	348,654123
Tiempo transcurrido:	0,001192	0,017662	0,142030	1,252363	7,271393	18,813237	39,198193	347,840245
Tiempo transcurrido:	0,001205	0,017858	0,135765	1,255282	7,315442	18,921431	39,434000	346,221380
Tiempo transcurrido:	0,000842	0,017961	0,135723	1,259195	7,278198	19,116688	39,268730	352,655130
Tiempo transcurrido:	0,001179	0,017892	0,135946	1,248427	7,295930	19,024492	39,138111	345,640279
Tiempo transcurrido:	0,000897	0,017879	0,135861	1,263945	7,300090	18,863366	39,278499	349,104166
Tiempo transcurrido:	0,000798	0,017710	0,138235	1,254051	7,326372	19,167805	39,526523	348,163095
Tiempo transcurrido:	0,001478	0,017722	0,135695	1,264202	7,349434	19,110710	39,434593	353,080798
Promedio	0,0010889	0,0178314	0,1370715	1,2691284	7,3443092	19,0646698	39,4289622	350,146212
speedup	0,15327395	0,85792478	0,69247947	0,68780472	0,7433304	0,78832803	0,7960977	0,7603029

12. Resultado de ejecutar el algoritmo con optimización por consola con O3 (para este se utilizó el programa basado en hilos)

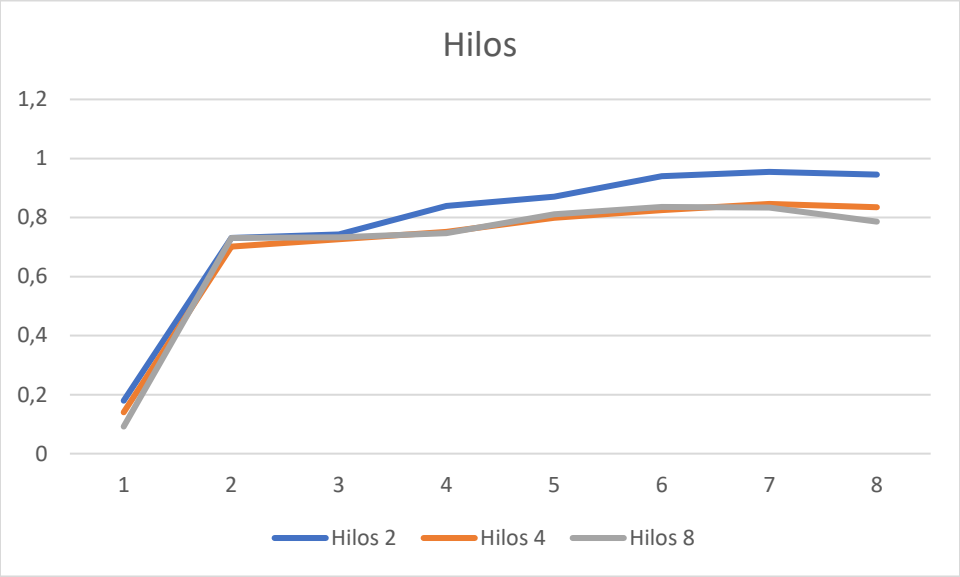
tiempo\matriz	10	100	200	400	600	800	1000	2000
Tiempo transcurrido:	0,000753	0,004525	0,029347	0,254745	2,622681	7,580328	17,078606	159,972599
Tiempo transcurrido:	0,001299	0,004452	0,028180	0,251122	2,602054	7,608626	17,213217	152,308526
Tiempo transcurrido:	0,001033	0,004405	0,028344	0,250584	2,638467	7,225962	16,204637	155,446091
Tiempo transcurrido:	0,001131	0,004920	0,029431	0,299703	2,876759	7,858464	17,515124	156,016079
Tiempo transcurrido:	0,000799	0,004411	0,027902	0,254227	2,598293	7,434792	17,096038	154,578270
Tiempo transcurrido:	0,001020	0,004485	0,027989	0,257431	2,578475	7,310787	16,529685	157,773493
Tiempo transcurrido:	0,001158	0,004414	0,028212	0,254085	2,633246	7,618838	16,607813	150,037200
Tiempo transcurrido:	0,001099	0,004504	0,028093	0,272297	2,696481	7,372919	16,211457	150,168715
Tiempo transcurrido:	0,000632	0,004333	0,027735	0,284278	2,733056	7,405200	16,231725	148,321383
Tiempo transcurrido:	0,001008	0,004641	0,028271	0,261401	2,662098	7,335154	16,315410	149,599843
Promedio	0,0009932	0,004509	0,0283504	0,2639873	2,664161	7,475107	16,7003712	153,42222
speedup	0,16804269	3,39277002	3,34807269	3,30664581	2,04914354	2,01056834	1,8795574	1,73519312

Graficas comparativas

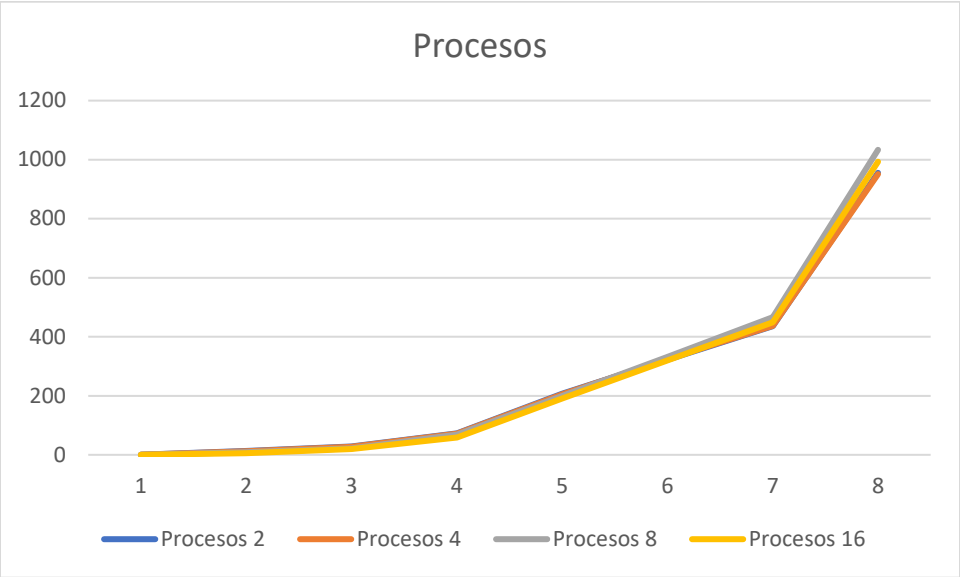
Para ver un poco mejor y poder ver las diferencias entre las implementaciones, las vemos comparadas en gráficos para poder diferenciarlo de otra manera, los valores del eje X representan el tamaño de la matriz siendo 8 el máximo de 2000 valores

Primero hacemos una comparativa del speedup de los hilos para ver cual tuvo mejor calificación entre los mejores hilos y escoger uno para las otras graficas comparativas:

En el grafico de hilos se puede ver que tuvo mejor resultado con 2 hilos



En el grafico con procesos podemos ver que el que tuvo mejor resultado, aunque es por muy poco fueron los 8 procesos



Hilos: 2

Procesos: 8

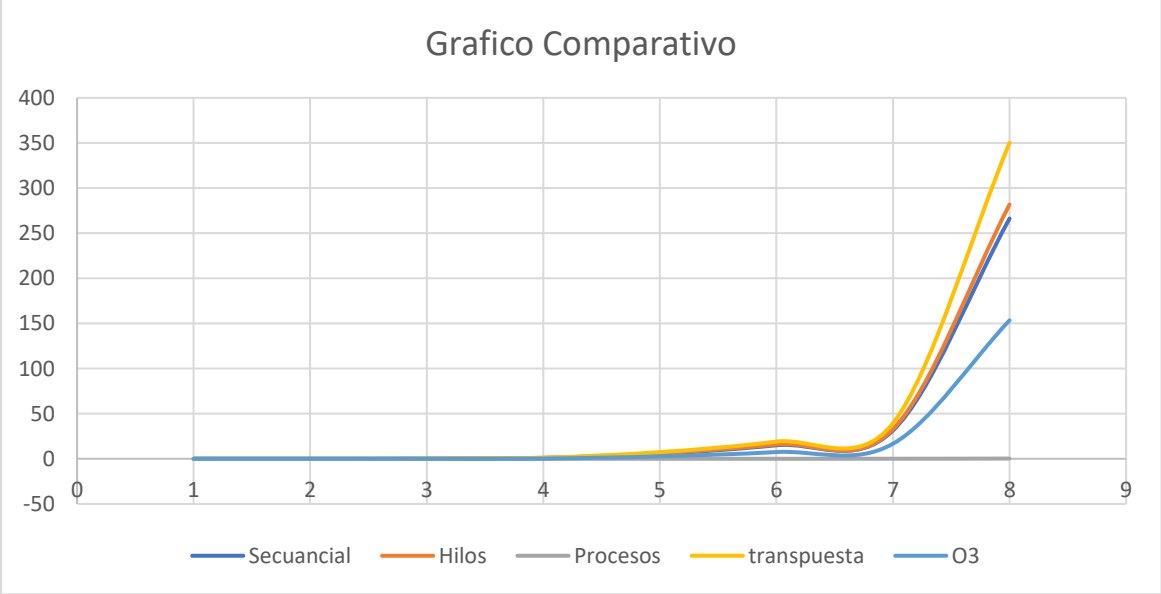
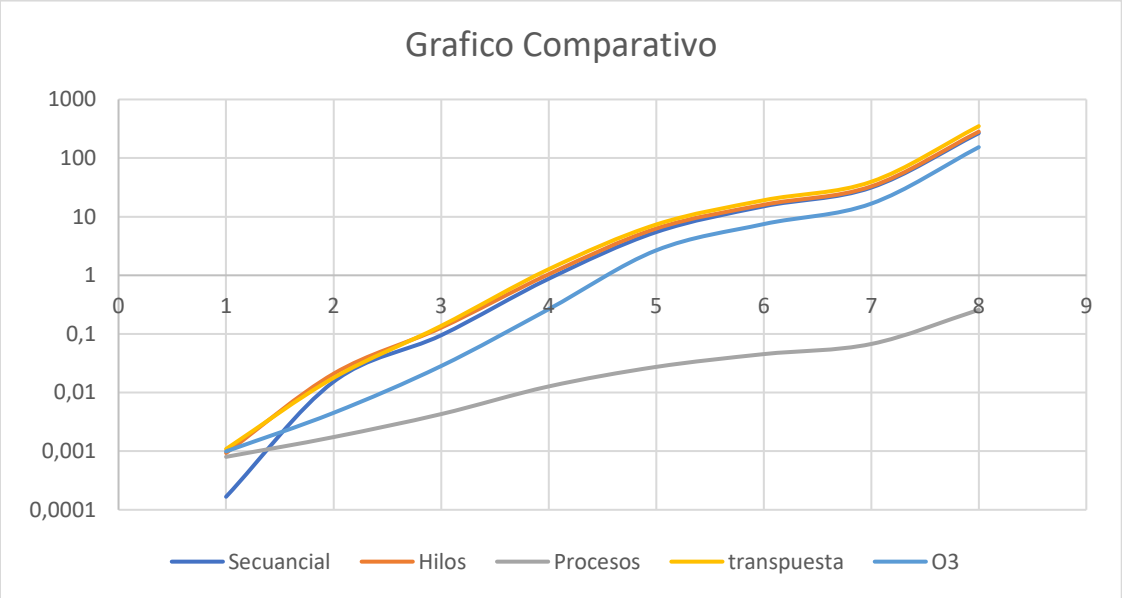


Grafico en Logaritmo base 10

Hilos: 2

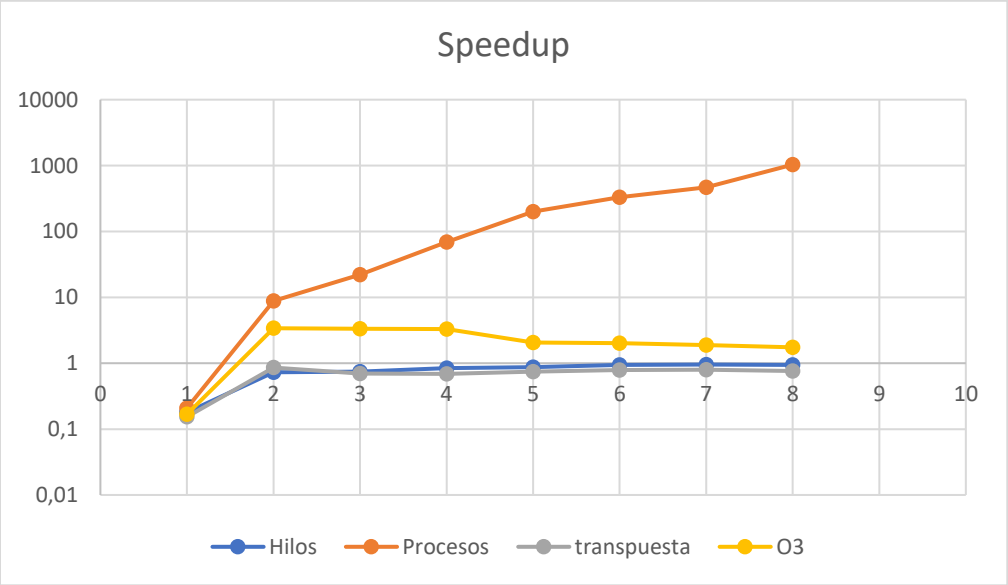
Procesos: 8



Mejora de velocidades con respecto a la secuencial

Hilos: 2

Procesos: 8



Conclusiones

1. Este trabajo podría decir que el uso de hilos podría no ser verdaderamente efectivo por que claramente podemos ver que es mucho peor que hacerlo secuencialmente, pero no podemos decir lo mismo de los procesos.

2. A medida que mas hilos se utilizaban mas se demoraba, por lo que puedo concluir que el sistema y el ambiente en el que se estaba probando quizás no eran los adecuados ya que en otros sistemas como el compilador online de C nos dice otra cosa con respecto a los tiempos, mostrando ser muchísimo mejor que la forma secuencial

3. Podemos ver que el sistema no respondía muy bien utilizando hilos como parte de su desarrollo ya que la matriz transpuesta al estar hecha en base al código de los hilos, también mostro deficiencia comparado con los otros, como por ejemplo con procesos.

4. No se puede establecer el porque la diferencia tan absurda entre los hilos y los procesos siendo ambos ambientes paralelos pero que necesitaba muchos mas recursos el de procesos, se puede intuir que el sistema en el que se estaba implementando no lo sabia usar de la mejor forma o tenía problemas para este.

Bibliografía

- HPC for Research (Sitio web de recursos)
- Introduction to Matrix Multiplication (Artículo)
- CUDA Programming (Sitio web de recursos)
- Understanding Speedup in Parallel Computing (Artículo)
- "Introduction to Multithreading in C" (artículo)