Informe: Matrix multiplication

Oscar Chujutalli

August 25, 2016

1 Comparación de tiempos

Se implementaron los algoritmos "3 nested loops" y "6 nested loops" los cuales solucionan la multiplicación de matrices. En el siguiente cuadro comparativo figuran los tiempos de ejecución:

elements	3 nested loops	6 nested loops
10	0,000017	0,000023
50	0,001912	0,002643
100	0,014157	0,021476
500	0,875914	1,28241
1000	12,6359	10.207
2000	-	73,435

Figure 1: Comparación de tiempos.

Al ingresar matrices de pequeñas dimensiones no podemos ver una gran diferencia entre las velocidades de los algoritmos. En la medida que se ingresen matrices con mayor número de elementos veremos que el algoritmo "6 nested loops" tiene un mejor desempeño.

2 Cache misses, acceso a memoria

Para evaluar los accesos a cache y memoria principal se utilizó valgrind.

```
==26285== Command: ./cache
==26285==
--26285-- warning: L3 cache found, using its data for the LL simulation.
0.26157
==26285==
==26285== I   refs:      54,935,313
==26285== I1  misses:         1,780
==26285== LLi misses:         1,666
==26285== I1  miss rate:       0.00%
==26285== LLi miss rate:       0.00%
==26285==
==26285== D   refs:      24,444,470  (23,067,799 rd   + 1,376,671 wr)
==26285== D1  misses:        84,110  (    80,348 rd   +     3,762 wr)
==26285== LLd misses:        11,171  (     8,349 rd   +     2,822 wr)
==26285== D1  miss rate:        0.3% (       0.3%     +       0.3%  )
==26285== LLd miss rate:        0.0% (       0.0%     +       0.2%  )
==26285==
==26285== LL refs:           85,890  (    82,128 rd   +     3,762 wr)
==26285== LL misses:         12,837  (    10,015 rd   +     2,822 wr)
==26285== LL miss rate:         0.0% (       0.0%     +       0.2%  )
oscar@oscar:Paralelos$
```

Figure 2: Evaluación valgrind para "3 nested loops".

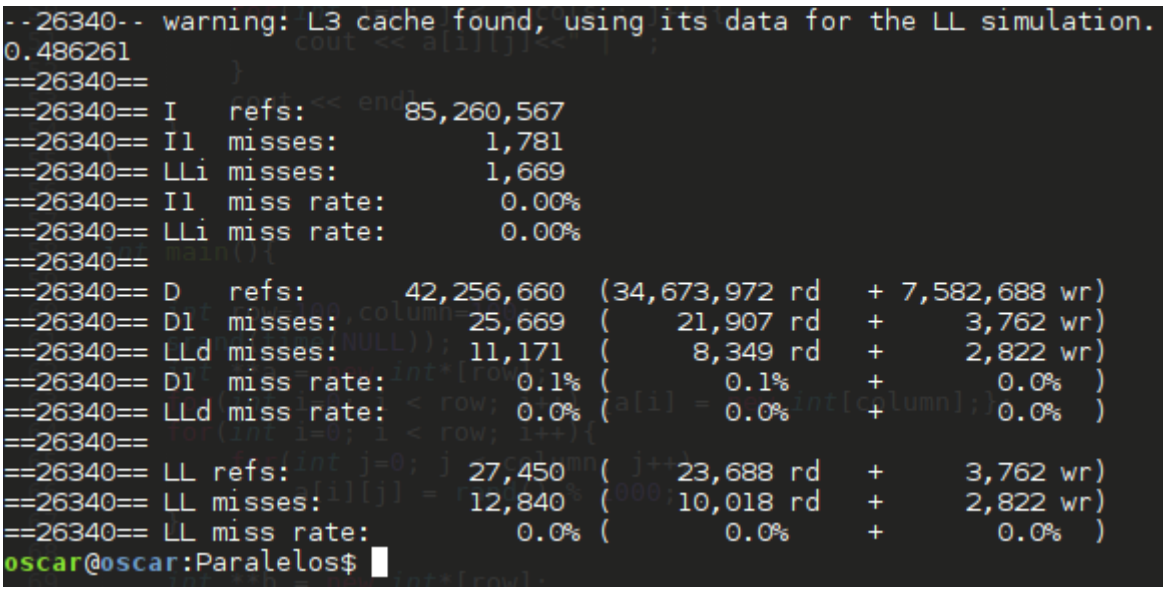

Figure 3: Evaluación valgrind para "6 nested loops".

En Figure2 y Figure3 podemos ver que la principal diferencia se encuentra en las líneas que indican los "misses" que se refieren a los cache misses, resultando más eficiente el algoritmo "6 nested loops", pues presenta cifras menores en dicha sección.

3 Conclusiones

- El algoritmo "3 nested loops" tiene un tiempo de ejecución de $3n^3$, su principal desventaja es el tamaño de las matrices de entrada.
- El algoritmo "6 nested loops" efectúa las mismas operaciones, con la diferencia que sub divide las matrices de entrada en bloques más pequeños, asegurando que éstos puedan ser almacenados en la memoria local y puedan ser operadas en el menor tiempo posible.
- El algoritmo "6 nested loops" demuestra mayor eficiencia, ya que por la evaluación de valgrind, presenta menor porcentaje de cache miss, lo cual indica que accede menos veces a memoria principal y mejora la localidad temporal.