Trabalho prático 4 – Estrutura de Dados

Aluno: Áquila Oliveira Souza

Introdução

Este relatório aborda um código que implementa um Tipo Abstrato de Dados (TAD) chamado "MAT," projetado para operações em matrizes bidimensionais. São discutidos pontos críticos de uso de memória, especialmente durante a inicialização de matrizes aleatórias e operações como cópia, soma, multiplicação e transposição. Métricas de desempenho obtidas com o Valgrind, incluindo taxas de falhas em cache, são apresentadas para destacar o impacto dessas operações na memória. O comando "cg_annotate cachegrind.out" é mencionado como ferramenta para análise detalhada do consumo de memória, permitindo identificar oportunidades de otimização.

Pontos Críticos em Relação à Memória:

O ponto mais crítico em relação à memória neste código é a inicialização de matrizes aleatórias na função **inicializaMatrizAleatoria**. Isso ocorre porque, dependendo do tamanho das matrizes **mat**, essa operação pode consumir uma quantidade significativa de memória devido ao uso de dois loops aninhados para preencher a matriz com valores aleatórios. Quanto maiores forem as dimensões da matriz, mais memória será necessária.

Além disso, as operações que envolvem cópia de matrizes (**copiaMatriz**), soma de matrizes (**somaMatrizes**), multiplicação de matrizes (**multiplicaMatrizes**) e transposição de matrizes (**transpoeMatriz**) podem consumir memória adicional dependendo do tamanho das matrizes envolvidas.

Coleta de dados

Com o comando **valgrind --tool=cachegrind ./bin/matop -m -x 5 -y 5**

==543== Cachegrind, a cache and branch-prediction profiler

==543== Copyright (C) 2002-2017, and GNU GPL'd, by Nicholas Nethercote et al.

==543== Using Valgrind-3.18.1 and LibVEX; rerun with -h for copyright info

==543== Command: ./bin/matop -m -x 5 -y 5

==543==

--543-- warning: L3 cache found, using its data for the LL simulation.

==543== error calling PR_SET_PTRACER, vgdb might block

	0	1	2	3	4
0	42.56	7.01	30.80	23.71	25.27

```
      1    97.84    18.96    87.36    70.83    88.44
      2   195.50    43.08   185.96   165.77   143.49
      3   118.97    19.34   102.75    80.23    38.18
      4   193.44   101.19   204.76   158.71   196.88
==543==
==543== I   refs:      228,686
==543== I1  misses:      1,598
==543== LLi misses:      1,531
==543== I1  miss rate:    0.70%
==543== LLi miss rate:    0.67%
==543==
==543== D   refs:       82,403  (56,561 rd   + 25,842 wr)
==543== D1  misses:      2,333  ( 1,682 rd   +    651 wr)
==543== LLd misses:      1,976  ( 1,371 rd   +    605 wr)
==543== D1  miss rate:     2.8% (   3.0%     +    2.5%  )
==543== LLd miss rate:     2.4% (   2.4%     +    2.3%  )
==543==
==543== LL refs:         3,931  ( 3,280 rd   +    651 wr)
==543== LL misses:       3,507  ( 2,902 rd   +    605 wr)
==543== LL miss rate:      1.1% (   1.0%     +    2.3%  )
```

- As linhas começando com **I refs:** se referem a referências de instruções. São métricas relacionadas ao acesso da CPU às instruções do programa.
- **I refs**: O número total de referências de instruções foi de 228,686.
- **I1 misses**: O número de falhas na cache L1 de instruções foi de 1,598.
- **LLi misses**: O número de falhas na cache unificada de instruções de último nível (LL) foi de 1,531.
- **I1 miss rate**: A taxa de falhas na cache L1 de instruções é de 0.70%.
- **LLi miss rate**: A taxa de falhas na cache LL de instruções é de 0.67%.
- As próximas linhas começando com **D refs:** se referem a referências de dados. São métricas relacionadas ao acesso da CPU aos dados do programa.
- **D refs**: O número total de referências de dados foi de 82,403.
- **D1 misses**: O número de falhas na cache L1 de dados foi de 2,333.
- **LLd misses**: O número de falhas na cache unificada de dados de último nível (LL) foi de 1,976.

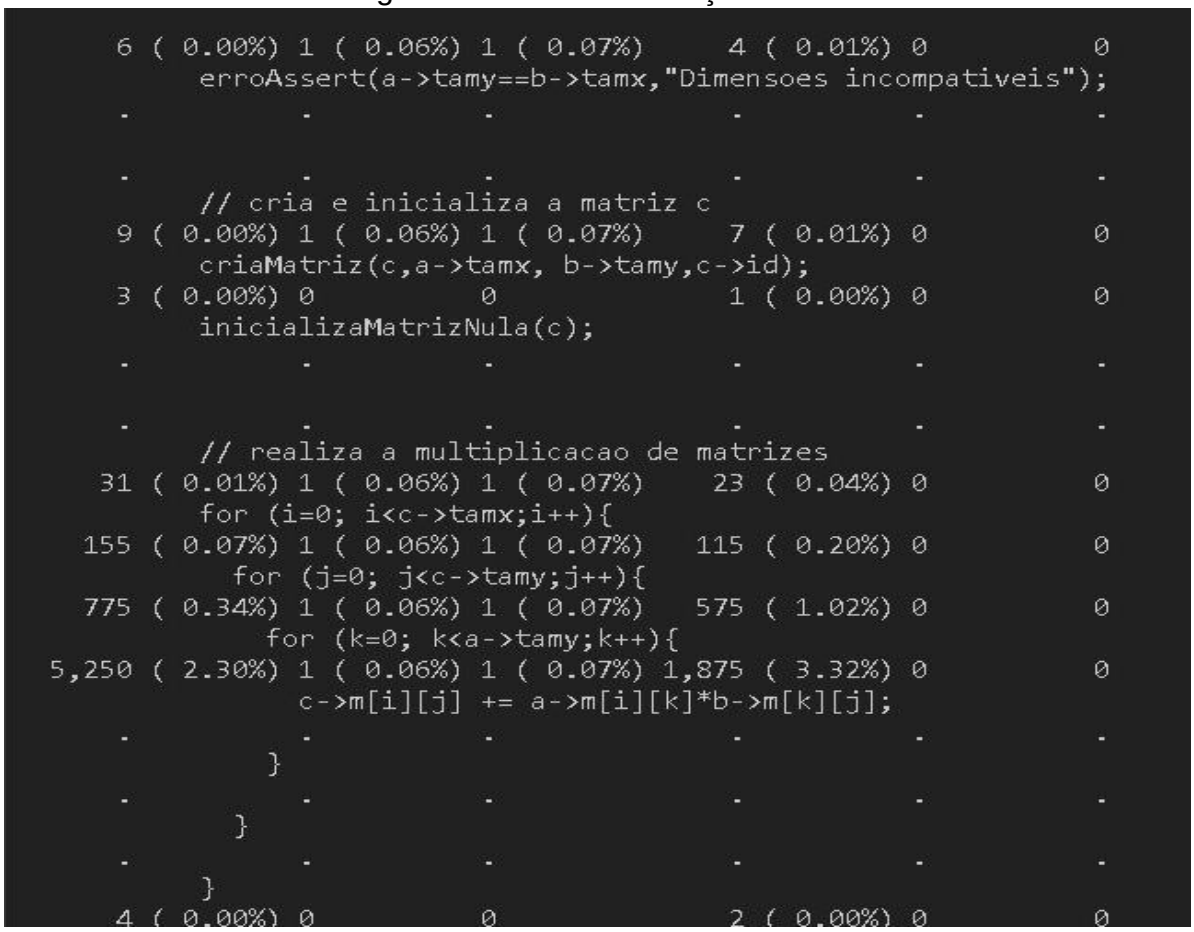
- **D1 miss rate**: A taxa de falhas na cache L1 de dados é de 2.8%.
- **LLd miss rate**: A taxa de falhas na cache LL de dados é de 2.4%.
- As últimas linhas começando com **LL refs:** e **LL misses:** se referem a referências de dados de último nível (LL) em geral.
- **LL refs**: O número total de referências de último nível foi de 3,931.
- **LL misses**: O número de falhas na cache unificada de dados de último nível (LL) foi de 3,507.
- **LL miss rate**: A taxa de falhas na cache LL de dados de último nível é de 1.1%.

Essas métricas ajudam a entender como o programa interage com a memória e a eficiência do cache durante a execução. Taxas mais baixas de falhas de cache são geralmente indicativas de um melhor desempenho, pois menos tempo é gasto buscando dados na memória principal.

Rodando o comando: **cg_annotate cachegrind.out.**

Com esse comando fica notável como as operações que foram destacadas no início do relatório destacam, como de fato, essas partes do programa consomem mais memoria nesse processo?

Como esse trecho do código mostra essa diferença



Conclusão

Em resumo, este relatório evidenciou os pontos críticos de uso de memória em um código que implementa um Tipo Abstrato de Dados (TAD) para operações em

matrizes bidimensionais. Ficou claro que a inicialização de matrizes aleatórias e operações como cópia, soma, multiplicação e transposição podem demandar uma quantidade considerável de memória, especialmente em matrizes grandes.