Relatório Laboratório 07

Maria Eduarda Teixeira Costa e Sofia Gazolla da Costa Silva

Departamento de Informática e Estatística - Universidade Federal de Santa Catarina

Organização de Computadores I

Professor Marcelo Daniel Berejuck

Novembro 2025

1 Introdução

Nesse laboratório, utilizamos a implementação de duas matrizes 16x16 que realizamos em um laboratório anterior para analisar o desempenho de diferentes políticas de alocação na cache para ambas as matrizes.

2 Desenvolvimento

Para analisar corretamente os resultados, devemos entender como a cache é organizada. De uma maneira simples, a cache é composta por blocos, e cada bloco tem um número fixo de palavras. São justamente esses dois parâmetros, quantidade de blocos e o tamanho de cada bloco em words, que variamos na simulação.

Quando o processador precisa acessar um dado da memória principal, todo o bloco que contém esse endereço é copiado para a cache. Se futuramente o processador precisar de outro dado que está contido nesse mesmo bloco, temos um hit (acerto). Se o dado solicitado não estiver em nenhum bloco presente na cache, ocorre miss (falha) e precisamos buscar outro bloco na memória principal.

Isso nos leva ao conceito fundamental para interpretar os resultados da simulação, a localidade espacial. Se o programa acessa endereços que estão fisicamente próximos na

memória, é provável que eles estejam no mesmo bloco, permitindo que vários acessos gerem hits. Caso os endereços estejam longe, cada acesso cairá em um bloco diferente, impedindo o reaproveitamento do bloco anterior, resultando em vários misses seguidos.

Analisando os códigos que exploramos e os nossos resultados, o preenchimento da matriz por linhas apresenta um desempenho muito superior o que acontece porque os elementos de uma mesma linha ficam lado a lado na memória. Assim, ao acessar o primeiro elemento da linha, vários outros elementos da mesma linha são trazidos juntos no mesmo bloco, podendo ser reutilizados e gerando muitos hits.

Entretanto, o preenchimento por colunas acessa elementos que, apesar de estarem na mesma coluna da matriz, estão longe uns dos outros na memória, por conta de a memória organizar a matriz por linhas. Assim, quando o processador acessa o primeiro elemento, o próximo estará em um endereço distante. O bloco carregado para acessar o primeiro elemento quase nunca contém o segundo, levando a misses em quase todos os acessos e impedindo qualquer reutilização de dados presentes na cache.

3 Materiais e métodos

Para o desenvolvimento dessa atividade foram utilizados os dois códigos implementados no quarto laboratório da disciplina, na linguagem `Assembly` e a IDE `MARS`, na qual foi realizada toda a escrita, execução e depuração de todos os programas implementados. Além disso, foi utilizada a ferramenta `Data Cache Simulator`, também da IDE `MARS`.

4 Resultados

Configuração da Cache (Blocos x Words/Bloco)	HIt Rate (Matriz por linhas)	Hit Rate (Matriz por colunas)
8 blocos x 4 words	75%	0%
8 blocos x 8 words	88%	0%
16 blocos x 8 words	88%	0%
16 blocos x 16 words	94%	94%
16 blocos x 32 words	97%	97%

Figure 1: Tabela com os resultados

A tabela apresenta o hit rate que foi obtido em cada configuração da cache para o preenchimento por linhas e por colunas. Os resultados deixam evidente como a forma de preenchimento influencia diretamente o aproveitamento da cache.

Nas três primeiras configurações (8 blocos x 4 words, 8 blocos x 8 words e 16 blocos x 8 words) observamos que a matriz preenchida por linhas atinge taxas de hit de 75%, 88% e 88%, respectivamente. Esses valores já são relativamente altos porque o acesso por linha explora fortemente a localidade espacial, que acessa vários elementos consecutivos de um mesmo bloco carregado da memória.

Entretanto, nessas mesmas configurações, a matriz preenchida por colunas apresenta hit rate igual a 0%. Isso acontece porque, como mencionado anteriormente, no acesso por colunas, os elementos acessados estão distantes na memória. Cada acesso cai em um bloco completamente diferente, impedindo qualquer reaproveitamento e gerando miss em praticamente todos os acessos. Nesses cenários de blocos pequenos, nenhum bloco é reutilizado, e a cache deixa de oferecer qualquer benefício.

A situação muda nas últimas duas configurações, 16 blocos x 16 words e 16 blocos x 32 words, onde tanto o preenchimento por linhas quanto por colunas atingem taxas de hit elevadas (94% e 97% para ambos). Isso acontece porque o tamanho dos blocos passa a ser suficientemente grande para conter, dentro de um único carregamento, todos

os elementos necessários de uma mesma coluna. Assim até o preenchimento por colunas passa a apresentar localidade espacial já que os elementos acessados cabem no mesmo bloco. Resumidamente, o aumento significativo do tamanho do bloco elimina o problema dos saltos grandes entre linhas e permite que o acesso por colunas também se beneficie da cache.

Principais conclusões após analisar os resultados:

- O padrão de acesso importa mais do que o tamanho da cache: com blocos pequenos, a diferença de desempenho entre o preenchimento por linhas e por colunas torna-se extremamente evidente.
- Blocos maiores suavizam o impacto do acesso desalinhado: quando o bloco possui espaço suficiente para acomodar vários elementos consecutivos na memória, até padrões de acesso pouco favoráveis (como o preenchimento por colunas) passam a apresentar bom desempenho.
- O acesso por linhas sempre aproveita melhor a estrutura da memória. Mesmo em configurações de cache pequenas, o acesso sequencial obtém altas taxas de hit, explorando naturalmente a localidade espacial.

5 Conclusão

A realização desse laboratório foi essencial para que pudéssemos aprimorar nosso entendimento sobre o funcionamento da memória cache e, em especial, sobre como o padrão de acesso aos dados influencia diretamente seu desempenho. Ao variar parâmetros como o número de blocos e o tamanho dos blocos, foi possível observar que a eficiência da cache depende menos da sua capacidade e mais da organização física da memória e à forma como o programa percorre os dados. As simulações evidenciaram claramente o papel da localidade espacial. Portanto, embora esse laboratório não tenha introduzido novos tópicos para nosso conhecimento prático em Assembly, ele foi importantíssimo para que pudessemos aprimorar nosso conhecimento teórico sobre o assunto de memórias cache.

6 Imagens dos Resultados

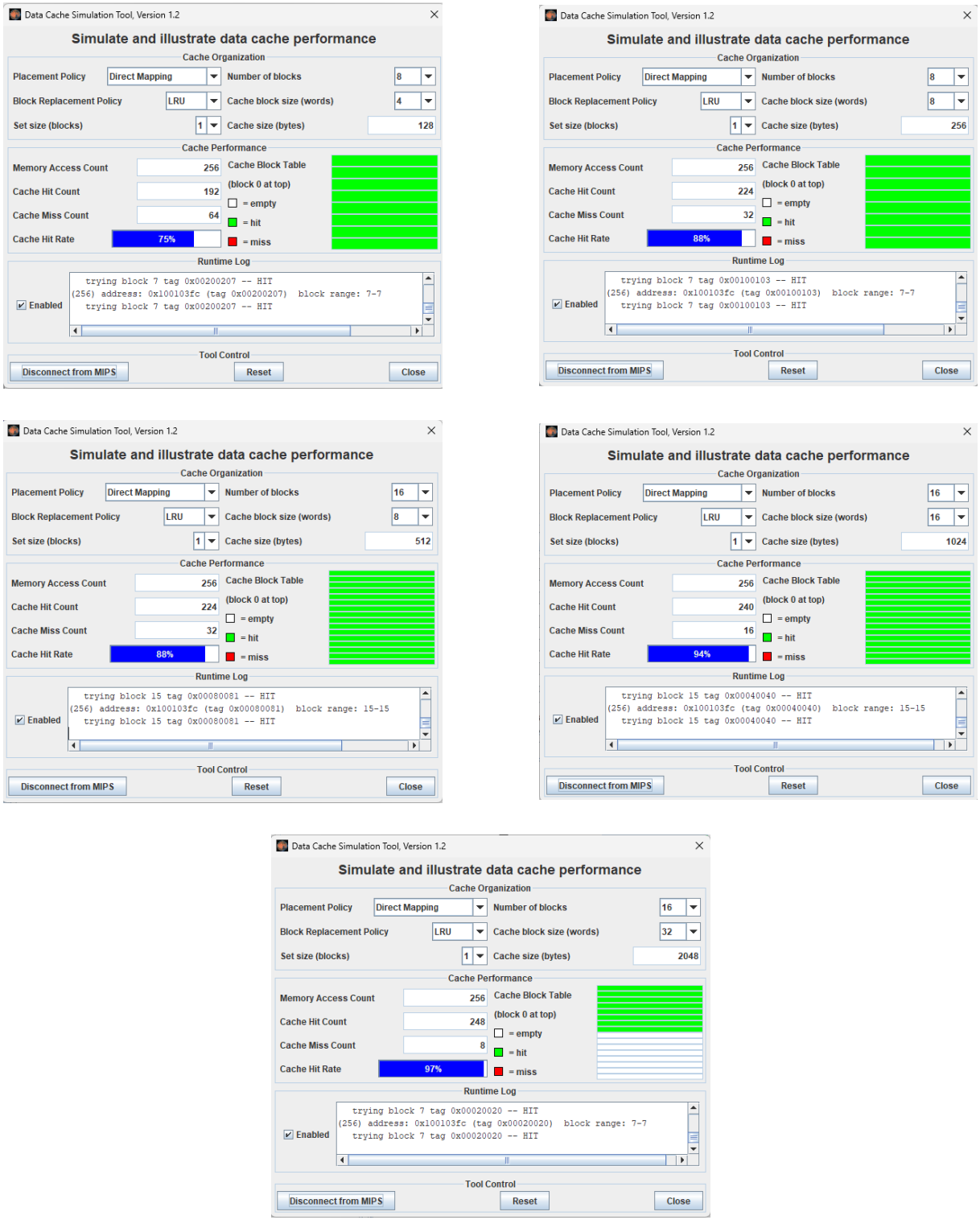

Figure 2: Resultados para a matriz preenchida por linhas

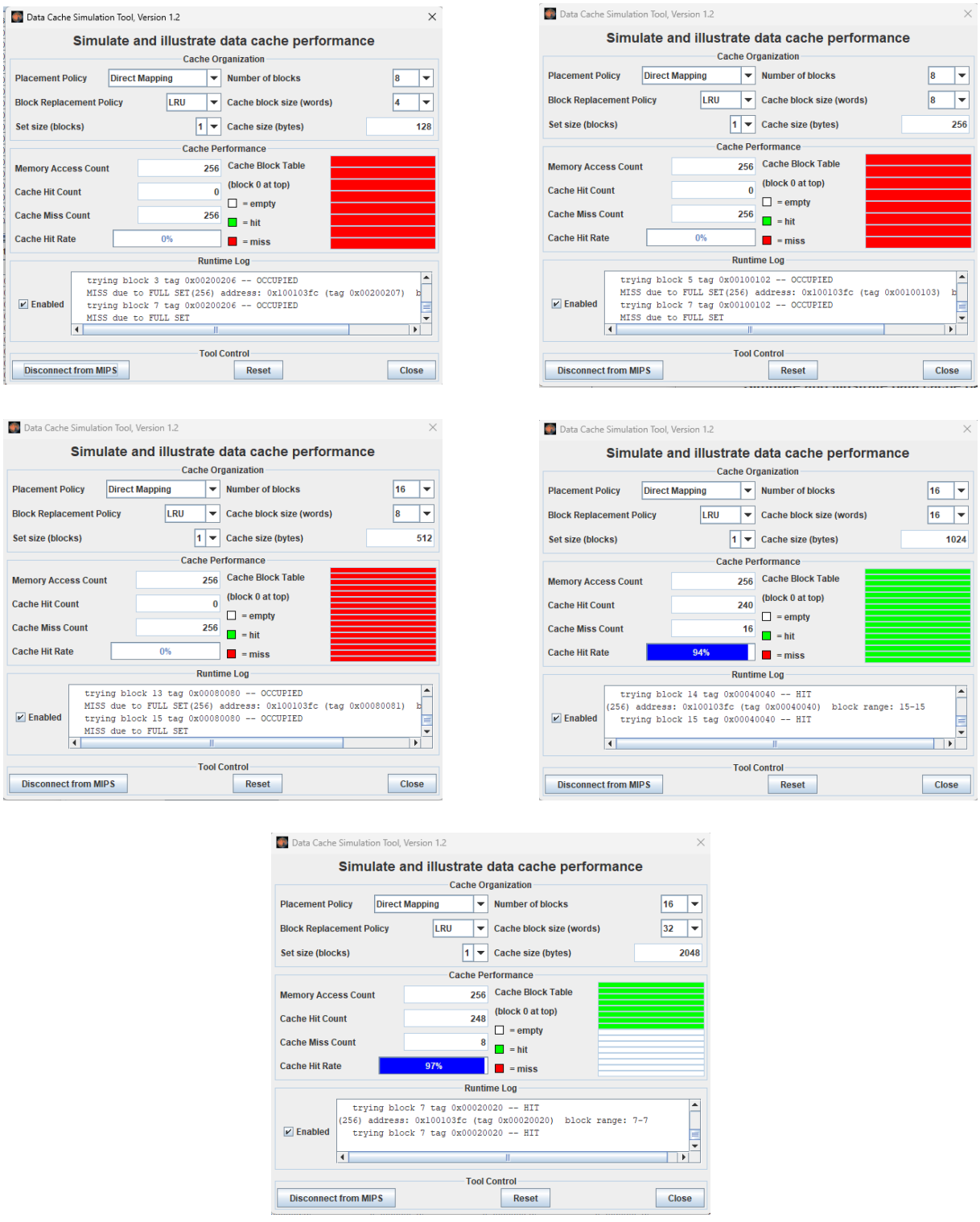

Figure 3: Resultados para a matriz preenchida por colunas