Cache Blocking
Organização de Computadores I

Juliana Miranda Bosio, Vinícios Rosa Buzzi

[1]Departamento de Informática e Estatística – Universidade Federal de Santa Catarina
Caixa Postal 5094 – 88035-972 – Florianópolis – SC – Brazil

1. Objetivo

O objetivo deste relatório é analisar o desempenho do simulador de cache utilizando o MIPS, através da execução de dois programas que realizam a soma de matrizes, com e sem a técnica de cache blocking. O tamanho das matrizes e dos blocos (quando aplicável) é parametrizável.

2. Metodologia e Procedimento de Simulação

O procedimento de simulação para ambos os programas foi realizado seguindo os passos descritos abaixo, utilizando diferentes configurações de cache para analisar o desempenho de cada implementação.

Os testes foram realizados utilizando os seguintes critérios:

- **Ordem da Matriz:** 16
- **Tamanho do Bloco:** 8 e 4
- **Tamanho da Word:** 16, 8, 4

Além disso, para o Programa 2, foi utilizado um tamanho de bloco específico de 4 e 2, referente à técnica de cache blocking. A tabela a seguir resume as configurações de teste para ambos os programas:

Programa	Ordem da Matriz	Tamanho do Bloco	Tamanho da Word
Programa 1	16	8, 4	16, 8, 4
Programa 2	16	8, 4 (Cache Blocking específico: 4, 2)	16, 8, 4

Table 1. Configuração dos Testes

Para cada configuração, foram realizados testes de desempenho no simulador de cache, utilizando diferentes combinações de tamanho de bloco e tamanho de word, a fim de observar a variação na taxa de acertos da cache e identificar qual configuração apresentou melhor desempenho. No Programa 2, o tamanho do bloco especificado refere-se à técnica de cache blocking e não à configuração do simulador de cache.

3. Soma de Matrizes Sem Cache Blocking

Nesta seção, discutiremos a implementação da soma de matrizes sem utilizar a técnica de cache blocking. Este método realiza a soma de uma matriz com outra transposta diretamente, sem otimizações específicas para o uso eficiente da cache.

3.1. Implementação

A implementação da soma de matrizes sem cache blocking envolve os seguintes passos principais:

- **Leitura das Matrizes**: Carrega as matrizes da memória.
- **Soma dos Elementos**: Realiza a soma de cada elemento da matriz original com o correspondente elemento da matriz transposta.
- **Armazenamento do Resultado**: Armazena o resultado na matriz de saída.

Neste método, cada acesso à matriz ocorre de forma linear, sem dividir a matriz em blocos menores. Essa abordagem não aproveita a localidade espacial, o que pode levar a um número maior de faltas de cache (*cache misses*).

3.2. Conceito da Implementação

O conceito desta implementação é simples e direto:

- **Vantagens**:
  - **Facilidade de Implementação**: O código é mais fácil de entender e implementar, pois não envolve complexidade adicional para gerenciar blocos de memória.
  - **Linearidade**: O processamento é realizado de maneira sequencial, facilitando a depuração e compreensão.
- **Desvantagens**:
  - **Desempenho de Cache**: Não aproveita a localidade espacial da memória, resultando em um maior número de faltas de cache.
  - **Ineficiente para Matrizes Grandes**: Em matrizes maiores, o impacto de acessos à memória não otimizados pode ser significativo, reduzindo o desempenho global.

4. Soma de Matrizes com Cache Blocking

Nesta seção, discutiremos a implementação da soma de matrizes utilizando a técnica de cache blocking. Este método divide a matriz em blocos menores, otimizando o uso da cache e reduzindo o número de faltas de cache.

4.1. Implementação

A implementação da soma de matrizes com cache blocking envolve os seguintes passos principais:

- **Leitura das Matrizes em Blocos**: Carrega blocos menores de elementos da matriz para a cache.
- **Soma dos Elementos em Blocos**: Realiza a soma de cada elemento da matriz original com o correspondente elemento do bloco transposto.
- **Armazenamento do Resultado em Blocos**: Armazena o resultado na matriz de saída em blocos.

Nesta abordagem, a matriz é dividida em blocos menores (*submatrizes*), permitindo um melhor aproveitamento da localidade espacial da memória e reduzindo o número de faltas de cache.

4.2. Conceito da Implementação

O conceito desta implementação é focado na otimização do uso da cache:

- **Vantagens**:
  - **Desempenho de Cache**: Ao dividir a matriz em blocos menores, aproveita melhor a localidade espacial da memória, resultando em um menor número de faltas de cache.
  - **Eficiência para Matrizes Grandes**: Em matrizes maiores, o impacto da otimização de cache é mais significativo, melhorando o desempenho global.
- **Desvantagens**:
  - **Complexidade de Implementação**: O código é mais complexo e requer uma compreensão mais detalhada da gestão de memória.
  - **Gerenciamento de Blocos**: Dividir a matriz em blocos e gerenciar os endereços de memória pode ser mais complicado, exigindo um cuidado adicional para evitar erros.

5. Discussão dos Resultados

A tabela 2 apresenta o desempenho dos programas 1 e 2 com diferentes configurações de cache. A seguir, discutimos os motivos por trás desses resultados e fornecemos uma análise detalhada de cada programa.

Programa	N.º Blocos	Tamanho Blocos	Cache Blocking	Desempenho
1	4	16	inválido	72%
2	4	16	4	76%
2	4	16	2	73%
1	4	8	inválido	68%
2	4	8	4	66%
2	4	8	2	70%
1	4	4	inválido	61%
2	4	4	4	61%
2	4	4	2	57%
1	8	16	inválido	74%
2	8	16	4	83%
2	8	16	2	80%
1	8	8	inválido	71%
2	8	8	4	81%
2	8	8	2	76%
1	8	4	inválido	63%
2	8	4	4	64%
2	8	4	2	70%

Table 2. Desempenho dos Programas com Diferentes Configurações de Cache

5.1. Análise dos Resultados

Os resultados demonstram que o Programa 2, que utiliza a técnica de cache blocking, geralmente apresenta um desempenho melhor em comparação com o Programa 1, que não utiliza essa técnica. Aqui estão alguns pontos-chave observados:

- **Desempenho Geral:** O Programa 2 apresenta uma melhor taxa de acertos na cache devido ao uso eficiente da localidade espacial da memória. A divisão da matriz em blocos menores permite que a cache seja usada de forma mais eficiente, reduzindo o número de faltas de cache (*cache misses*).
- **Tamanho dos Blocos:** Notamos que blocos maiores, como 16 palavras, geralmente resultam em um desempenho melhor para ambos os programas. Isso ocorre porque blocos maiores permitem o carregamento de mais dados na cache de uma vez, o que é benéfico para o acesso sequencial típico das operações com matrizes.
- **Cache Blocking:** Para o Programa 2, a configuração de cache blocking com blocos de tamanho 4 mostrou-se mais eficiente do que blocos de tamanho 2, indicando que blocos intermediários são ideais para otimizar o desempenho sem aumentar excessivamente o número de acessos à memória.

5.2. Parecer sobre o Programa 1

O Programa 1 não utiliza a técnica de cache blocking. Embora seja mais simples de implementar, essa abordagem é menos eficiente em termos de desempenho de cache. Os resultados indicam que, sem cache blocking, a taxa de acertos na cache é significativamente menor. Em configurações de cache menores (por exemplo, blocos de 4 palavras), o desempenho cai ainda mais devido à maior frequência de faltas de cache.

5.3. Parecer sobre o Programa 2

O Programa 2 faz uso da técnica de cache blocking, o que se reflete em um melhor desempenho de cache. Ao dividir a matriz em blocos menores, o programa consegue melhor aproveitar a localidade espacial da memória, resultando em uma maior taxa de acertos. Observamos que o uso de blocos de cache blocking de tamanho 4 proporcionou um bom equilíbrio entre complexidade e desempenho, superando consistentemente o Programa 1.

Em resumo, a técnica de cache blocking empregada no Programa 2 provou ser eficaz para otimizar o desempenho de cache em operações de soma de matrizes. Essa abordagem é especialmente vantajosa em matrizes grandes, onde a localidade espacial pode ser melhor explorada para minimizar o número de acessos à memória principal.

6. Conclusão

Neste relatório, exploramos e analisamos duas abordagens distintas para a soma de matrizes: sem cache blocking e com cache blocking. Através dos experimentos realizados e das simulações no Data Cache Simulator, adquirimos vários aprendizados valiosos:

6.1. Aprendizados Gerais

- **Importância da Localidade Espacial**: A técnica de cache blocking demonstrou a importância de aproveitar a localidade espacial da memória para otimizar o desempenho dos programas, reduzindo significativamente o número de faltas de cache.

- **Eficiência em Matrizes Grandes**: Observamos que a abordagem com cache blocking é particularmente vantajosa para matrizes grandes, onde a estrutura de blocos permite um uso mais eficiente da cache.
- **Desempenho Relativo**: O Programa 2, que utiliza cache blocking, apresentou um desempenho superior em comparação com o Programa 1, confirmando a eficácia da técnica em melhorar a taxa de acertos na cache.

6.2. Dificuldades Enfrentadas

A implementação da técnica de cache blocking trouxe algumas dificuldades e desafios:

- **Complexidade da Implementação**: Implementar o cache blocking exige uma compreensão mais profunda da gestão de memória e do funcionamento da cache, tornando o código mais complexo e sujeito a erros.
- **Gerenciamento de Blocos**: Dividir a matriz em blocos e gerenciar os endereços de memória corretamente pode ser complicado, especialmente ao lidar com índices e offsets.
- **Afinamento de Parâmetros**: Encontrar o tamanho ideal dos blocos para maximizar o desempenho requer experimentação e análise, pois blocos muito pequenos ou muito grandes podem não oferecer os melhores resultados.

Em suma, este relatório destacou a relevância da técnica de cache blocking na otimização do uso da cache em operações de soma de matrizes. Apesar das dificuldades na implementação, os benefícios em termos de desempenho tornam essa técnica uma ferramenta poderosa para melhorar a eficiência dos programas que lidam com grandes volumes de dados. A compreensão e aplicação adequada dessas técnicas são essenciais para o desenvolvimento de algoritmos eficientes em ambientes de computação modernos.

7. Repositório da Atividade

Confira os códigos implementados acessando o $QRCode$ abaixo ou pelo link:

`https://github.com/buzziologia/UFSC/tree/main/OrganizacaoDeComputadores/Laboratorio07`

Figure 1. Repositório Github

Anexo

Neste anexo, estão apresentados os resultados obtidos durante as simulações da cache utilizando os códigos implementados no laboratório. Os resultados foram agrupados em três arquivos de imagem que representam as diferentes configurações testadas.

Figure 2. Simulação da Cache para o Programa 1 (Sem Cache Blocking) com todas as configurações.

A Figura 2 ilustra o desempenho da cache ao executar o Programa 1 (sem cache blocking) com todas as configurações de teste. Observa-se a variação na taxa de acertos da cache conforme as diferentes combinações de tamanho de bloco e número de blocos.

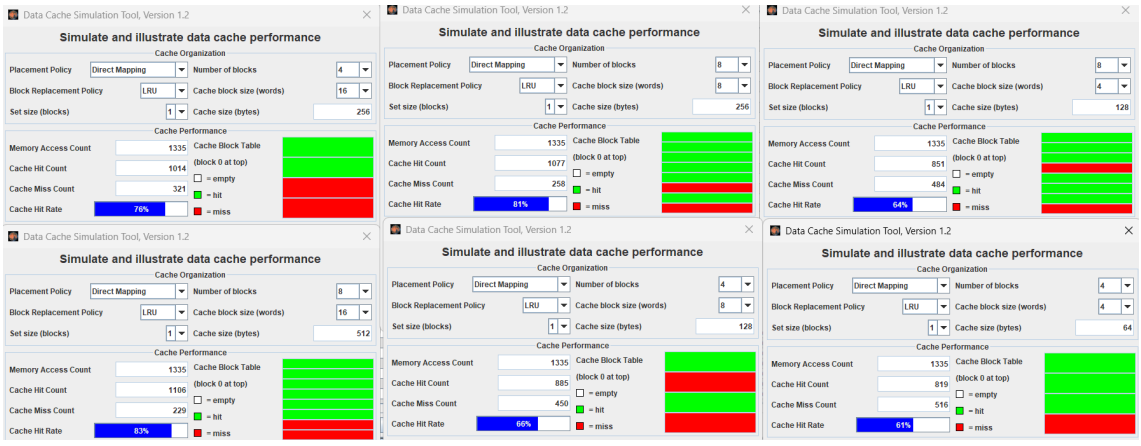

Figure 3. Simulação da Cache para o Programa 2 (Com Cache Blocking) com Bloco de Tamanho 4.

A Figura 3 mostra o desempenho da cache ao executar o Programa 2 (com cache blocking) utilizando blocos de tamanho 4. A técnica de cache blocking melhora a taxa de acertos ao aproveitar melhor a localidade espacial da memória, especialmente em configurações com blocos maiores.

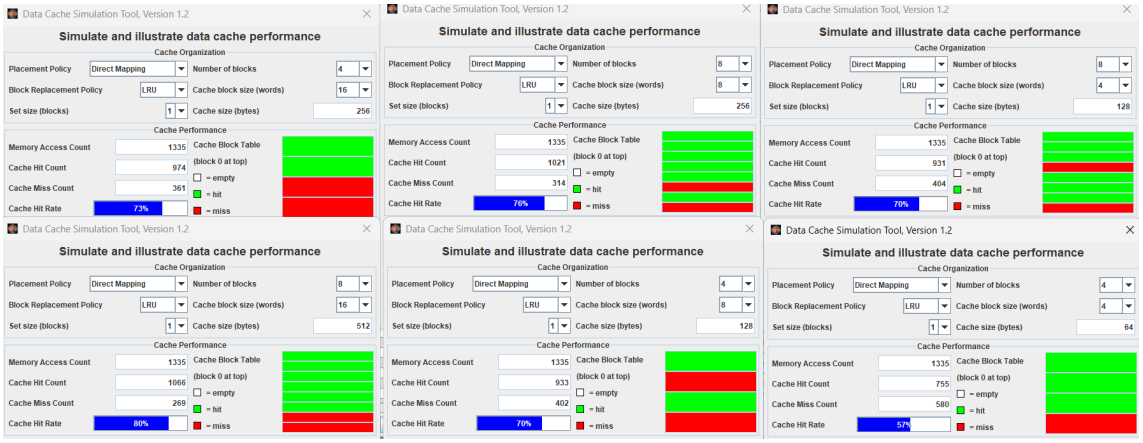

Figure 4. Simulação da Cache para o Programa 2 (Com Cache Blocking) com Bloco de Tamanho 2.

A Figura 4 apresenta o desempenho da cache para o Programa 2 utilizando blocos de tamanho 2. Apesar da redução no tamanho dos blocos, o cache blocking ainda demonstra melhorias na taxa de acertos em comparação com o Programa 1.

Cada uma das figuras destaca a eficácia da técnica de cache blocking em diferentes cenários, reforçando a importância de otimizar o uso da cache para melhorar o desempenho de programas que manipulam grandes volumes de dados.