Trabalho 2 - Relatório
Universidade Federal do Matogrosso do Sul
Arquitetura de Computadores II
Simulação e Avaliação da Predição de Desvios

Alunos: Andrew P. Borges, Caio S. T. Pereira, Gabriel J. de Oliveira
Professora: Nahri Balesdent Moreano
UFMS - FACOM

2025/1

1 Introdução

Este trabalho tem como objetivo simular e avaliar técnicas de predição de desvios condicionais, fundamentais para otimizar o desempenho de processadores modernos. Utilizando o simulador simpred, foram implementadas abordagens estáticas e dinâmicas com diferentes estratégias. Experimentos foram conduzidos com arquivos de rastreamento reais, variando o número de linhas do Branch Prediction Buffer. Ao final, analisaram-se as taxas de acerto de cada técnica, evidenciando sua eficácia comparativa.

2 Linguagem

Python foi a linguagem escolhida pela sua simplicidade sintática, que torna o código mais legível e fácil de desenvolver e depurar além de minimizar as chances de conflitos visto que o código foi desenvolvido em ambiente Windows e, portanto, poderia haver diferenças de compilação caso a escolha fosse linguegem compilada e esta se desse em ambiente linux.

Comando de Execução:

python simpred.py arquivo-teste.txt nLinhasBPB

3 Experimentos

3.1 Experimento 1: Indexação do BPB

a) Para as técnicas de predição 1-bit e 2-bits, o número de linhas do Branch Prediction Buffer (nLinhasBPB) é sempre uma potência de 2.

Isso ocorre porque o índice utilizado para acessar o BPB é obtido diretamente a partir de alguns bits do endereço da instrução de desvio. Utilizando potências de 2, é possível calcular esse índice com uma simples máscara binária, garantindo acesso eficiente e sem colisões inesperadas.

b) Os traces utilizados foram gerados a partir da execução de programas em um simulador com as seguintes características:

- Todas as instruções têm tamanho de 32 bits;
- O processador trabalha com palavras de 32 bits;
- A memória principal é endereçada por bytes;
- Os endereços de memória são de 32 bits.

Para as técnicas de predição 1-bit e 2-bits, o BPB é indexado usando parte dos bits menos significativos do endereço da instrução de desvio, desconsiderando os bits de posição 0 e 1. O número de bits utilizados é $\log_2$(nLinhasBPB).

b.1) Por que os dois bits menos significativos do endereço da instrução de branch não são utilizados?

Porque todas as instruções são alinhadas a 4 bytes, o que significa que os dois bits menos significativos do endereço são sempre 0. Portanto, eles não contribuem para a diferenciação entre instruções e são ignorados para evitar redundância na indexação.

b.2) O que aconteceria se os bits utilizados para o índice iniciassem na posição 0?

Isso causaria colisões artificiais no acesso ao BPB, pois múltiplas instruções distintas com o mesmo valor nos bits superiores (mas variações irrelevantes nos bits 0 e 1) seriam mapeadas para entradas diferentes, desperdiçando espaço e prejudicando a acurácia da predição.

Experimento 2: Técnicas Estáticas

Deseja-se analisar o desempenho das três técnicas de predição estática implementadas: Not-Taken, Taken e Direção.

a) Para cada técnica de predição estática, deve-se calcular:

- A taxa de acertos para cada trace fornecido;
- A taxa de acertos média entre todos os traces, usando média aritmética simples.

As taxas devem ser apresentadas em porcentagem, com duas casas decimais.

b) Por que a técnica de predição taken em geral apresenta uma taxa de acertos melhor do que a técnica de predição not-taken?

Taxa de acertos (%)			
Trace	**Técnica de predição**		
	NT	T	Direção
trace_fft_mi	42,12	52,88	67,09
trace_gsm_decoder_me	43,90	56,10	65,81
trace_patricia_mi	42,67	57,33	70,12
Taxa de acertos média	55,69	55,43	67,67

Tabela 1: Comparação das taxas de acerto por técnica de predição estática

A técnica Taken tende a apresentar taxas de acertos superiores à Not-Taken porque, na maioria dos programas reais, os desvios condicionais costumam ser tomados com mais frequência do que não tomados, especialmente em loops. Isso favorece a predição "sempre tomada", elevando sua precisão estatística.

Experimento 3: Técnicas Dinâmicas

As técnicas de predição dinâmica usam o BPB, um buffer em hardware. Uma estimativa simples do custo do hardware do BPB é:

número total de bits do BPB = nLinhasBPB×número de bits em uma linha do BPB

Desejamos analisar as 2 técnicas de predição dinâmica implementadas, predição 1-bit e predição 2-bits, com o trace `trace_rijndael_encoder_mi.txt`.

a) Para cada técnica de predição dinâmica e cada valor de nLinhasBPB, calcule:

- A taxa de acertos
- O número total de bits do BPB

As taxas de acerto devem ser apresentadas na forma de porcentagem, arredondadas para 2 casas decimais.

nLinhasBPB	**Taxa de acertos (%)**	**Nº total de bits do BPB**
16	76,28	16
32	80,27	32
64	84,13	64
128	84,25	128
256	84,31	256
512	84,42	512
1024	88,23	1024
2048	88,23	2048

Tabela 2: Resultados para predição 1-bit

nLinhasBPB	Taxa de acertos (%)	Nº total de bits do BPB
16	80,47	32
32	88,02	64
64	91,94	128
128	92,05	256
256	92,08	512
512	92,14	1024
1024	94,04	2048
2048	94,04	4096

Tabela 3: Resultados para predição 2-bits

b) Por que as taxas de acerto melhoram à medida que o nLinhasBPB cresce?

À medida que o número de linhas do Branch Prediction Buffer (nLinhasBPB) cresce, a taxa de acerto melhora principalmente por reduzir colisões de índice e interferências entre instruções diferentes.

c) Considerando apenas esse trace, se desejamos obter uma taxa de acertos de aproximadamente 90%, com o menor custo possível do hardware, que técnica de predição devemos escolher e qual será o nº total de bits do BPB?

Utilizando predição de 2 bits no BPB atingimos 128 bits com 64 linhas e 91,94]% de acerto.

b) Considerando apenas esse trace, se desejamos ter um custo do hardware de no máximo nº total de bits do BPB = 2048, com a maior taxa de acertos possível, que técnica de predição devemos escolher e qual será a taxa de acertos?

Predição de 2 bits com taxa de 94,04% de acerto.

3.2 Experimento 4: Interferência

As técnicas de predição dinâmica que utilizam BPB estão sujeitas à ocorrência de interferências:

A cada instrução de branch executada, uma predição é feita, consultando o BPB. Se nessa consulta, a informação obtida é relacionada a outra instrução de branch, então ocorreu uma interferência.

a) Descreva detalhadamente (não é necessário implementar) quais modificações (em estruturas de dados e de controle) são necessárias no simulador `simpred` para que ele calcule a taxa de interferências (definida a seguir) de cada técnica de predição dinâmica.

$$\text{taxa de interferências} = \frac{\text{número de interferências}}{\text{número de branches executados}}$$

Seria necessário armazenar em um vetor o endereço da instrução associada à cada índice do BPB. Dessa forma quando uma nova instrução acessar esse índice, o simulador poderá verificar se o endereço que está lá é o mesmo ou é de outra instrução.

b) Ao fazer a simulação de uma técnica de predição dinâmica, com um determinado valor de `nLinhasBPB`, como a taxa de interferências pode ser usada para propor alguma mudança?

A taxa de interferências pode ser usada como parâmetro para medição do ganho na taxa de acertos conforme se aumenta a quantidade de linhas no BPB.