

2o. Trabalho: Simulação e Avaliação da Predição de Desvios

- Parte I: **Desenvolvimento de um simulador de técnicas de predição de desvios condicionais**
- Parte II: **Avaliação das técnicas de predição**
 - Realizar experimentos com o simulador, utilizando **traces** de execução de programas reais
 - Obter resultados de desempenho das técnicas de predição
 - Analisar resultados
 - Responder questionário

Parte I: Simulação da Predição de Desvios

- Desenvolver **simulador de predição de desvios condicionais**
- Simulador implementa **diferentes técnicas de predição**:
 - **Técnicas estáticas**:
 - Predição not-taken
 - Predição taken
 - Predição baseada na direção:
 - Se desvio é “para frente”, então predição é NT
 - Se desvio é “para trás”, então predição é T
 - **Técnicas dinâmicas**:
 - Predição 1-bit:
 - Usa BPB (Branch Prediction Buffer)
 - Predição 2-bits:
 - Usa BPB
 - Baseada no autômato visto em aula

Simulador e Interface de Execução

- **Simulador `simpred`:**
 - **Desenvolver:** em C, c++, java ou Python
- **Interface de execução do simulador:**
 - **Entradas:**
 - Arquivo de trace
 - nLinhasBPB: n^o de posições do Branch Prediction Buffer
 - **Saídas:** Impressas na tela
 - Taxa de acertos de cada técnica de predição
(na forma de porcentagem, arredondada para 2 casas decimais)
 - **Execução por linha de comando:**
 - `simpred arquivoTrace nLinhasBPB`
 - **NÃO modificar interface de execução do simulador**

Entrada do Simulador: Trace de Execução

- Cada trace corresponde à execução de um programa, com uma determinada entrada de dados
- Contém informações sobre todas as **instruções de desvio condicional** executadas pelo programa
- **Arquivo texto:**
 - Cada linha descreve uma instrução de desvio condicional executada:
 - Endereço da instrução de desvio
 - Endereço da instrução **alvo** da instrução de desvio
 - Ocorrido: desvio foi tomado (T) ou não (N)
- **Exemplo:**

22276	22440	N
116108	116260	T
116108	116260	N
116172	116208	T
88424	88408	T

Exemplo: Trace de Execução

- Programa executado:

- Valores iniciais dos registradores: $x1 = 4$ e $x2 = 2$

```
( 0)  loop:    lw      x3, 0      (x1)      # R3 = Mem[0 + R1]
( 4)                add    x4, x4, x3      # R4 = R4 + R3
( 8)                beq    x1, x2, fim_if   # se R1 == R2 desvia para fim_if
(12)                sw     x3, 400 (x1)     # Mem[400 + R1] = R3
(16)  fim_if:  addi    x1, x1, -1          # R1 = R1 - 1
(20)                bne    x1, x0, loop     # se R1 != 0 desvia para loop
```

- Trace de execução:

8	16	N
20	0	T
8	16	N
20	0	T
8	16	T
20	0	T
8	16	N
20	0	N

Simulador de Predição de Desvios Condicionais

- **Inicializa:**

- Recebe por linha de comando: arquivo de trace, $nLinhasBPB$
- Aloca BPB1 com $nLinhasBPB$ para **predição 1-bit**
- Aloca BPB2 com $nLinhasBPB$ para **predição 2-bits**
- Inicializa medidas de desempenho:

} Inicializa
BPB1 e BPB2
tudo em Φ

$nBranchesExecutados = 0$

$nAcertosNT = 0$

$nAcertosT = 0$

$nAcertosDireção = 0$

$nAcertos1bit = 0$

$nAcertos2bits = 0$

- **Laço principal:**

- ...

- **Finaliza:**

- Calcula, **para cada técnica de predição:**

$$taxaAcertos (\%) = \frac{nAcertos}{nBranchesExecutados} \times 100$$

- Libera BPB1 e BPB2

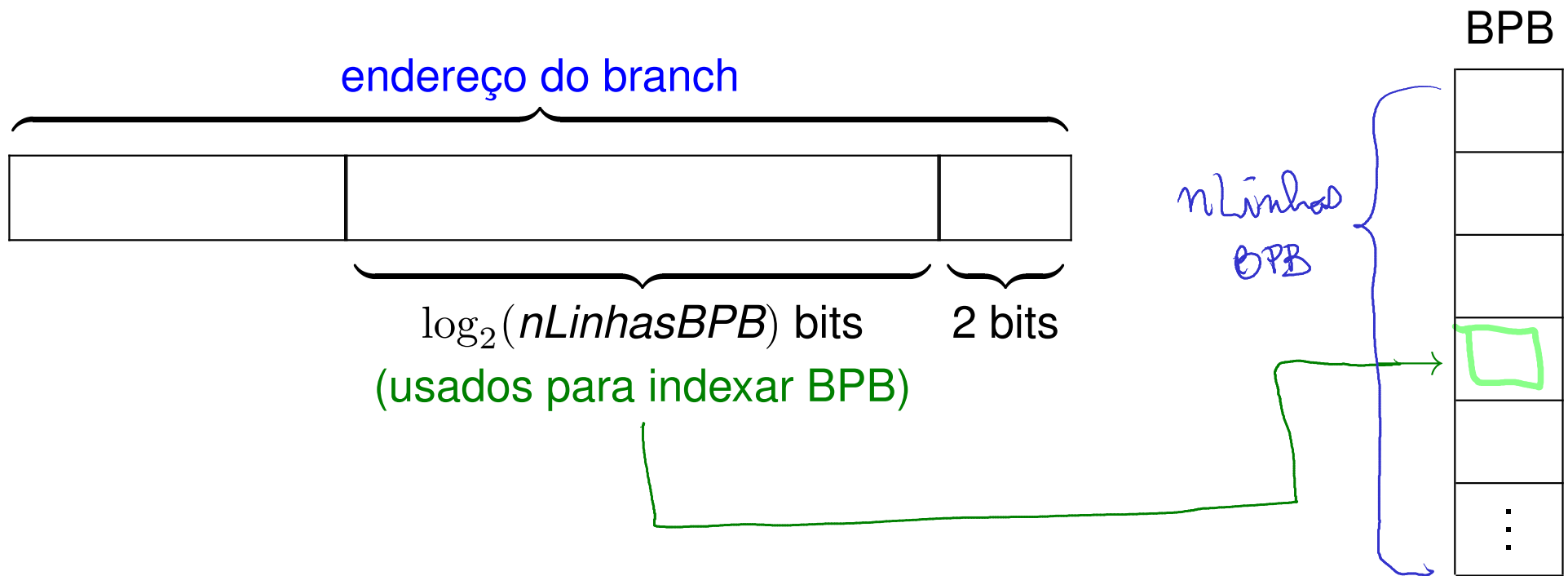
Simulador de Predição de Desvios Condicionais

- **Laço principal:**

- Enquanto não processou todos os desvios do trace:
 - Lê um desvio condicional do trace: (endereço do branch, endereço alvo, ocorrido)
 - Simula execução do desvio condicional, usando cada técnica de predição:
 - *nBranchesExecutados* ++
 - **Predição not-taken:**
Se *ocorrido* == 'N' então *nAcertosNT* ++
 - **Predição taken:**
...
 - **Predição baseada na direção:**
Determina predição, comparando *endereço do branch* e *endereço alvo*
Se predição == *ocorrido* então *nAcertosDireção* ++
 - **Predição 1-bit:**
Usa parte do *endereço do branch* para indexar BPB1 e obter predição
Se predição == *ocorrido* então *nAcertos1bit* ++
Se necessário, atualiza bits do BPB1, de acordo com técnica de predição
 - **Predição 2-bits:**
...

Branch Prediction Buffer (BPB)

- Usado para técnicas de predição 1-bit e predição 2-bits
- N^o de linhas do BPB ($nLinhasBPB$) definido por linha de comando
- Todos os bits do BPB inicializados em 0
- BPB é indexado usando parte (alguns bits) do endereço do branch:
 - Número de bits usados: $\log_2(nLinhasBPB)$
 - Bits usados: bits menos significativos, excluindo bits de posição 0 e 1



Resultados para Conferência

- **Trace:** `trace_rijndael_encoder_mi.txt`
- **Resultados:**
 - $nBranchesExecutados = 1.025.764$
 - $nAcertosNT = 397.764$
 - $nAcertosT = 628.000$
 - $nAcertosDireção = 708.828$
 - $nAcertos1bit = 782.480$ (para $nLinhasBPB = 16$)
 - $nAcertos2bits = 786.175$ (para $nLinhasBPB = 16$)
 - $nAcertos1bit = 904.983$ (para $nLinhasBPB = 1024$)
 - $nAcertos2bits = 964.353$ (para $nLinhasBPB = 1024$)

Parte II: Avaliação de Desempenho

- **Realizar experimentos:**
 - Executar simulador com traces fornecidos e opções indicadas
 - Coletar medidas de desempenho
- **Responder questionário com avaliação das técnicas de predição de desvios**
- **Atenção:**
 - **Depuração do programa \neq avaliação de desempenho:**
 - Traces fornecidos são grandes e serão realizadas diversas execuções
 - Testar e depurar simulador com traces menores
 - Após simulador estar correto,
executar com traces fornecidos para avaliação de desempenho

Entrega do Trabalho

- **Grupos:** 2 alunos ou 3 alunos
- **Data de entrega:** 29 junho, domingo
- **Submissão pelo AVA:**
 - Submeter um único arquivo **trab2.zip**, contendo:
 - Arquivos **fontes** que compõem o programa desenvolvido
 - Arquivo **PDF** do relatório com:
 - Nomes dos alunos integrantes do grupo
 - Instruções de como compilar o simulador desenvolvido
 - Questionário de avaliação de desempenho respondido
 - **NÃO submeter traces nem arquivo executável**
 - **Apenas um aluno do grupo deve submeter**