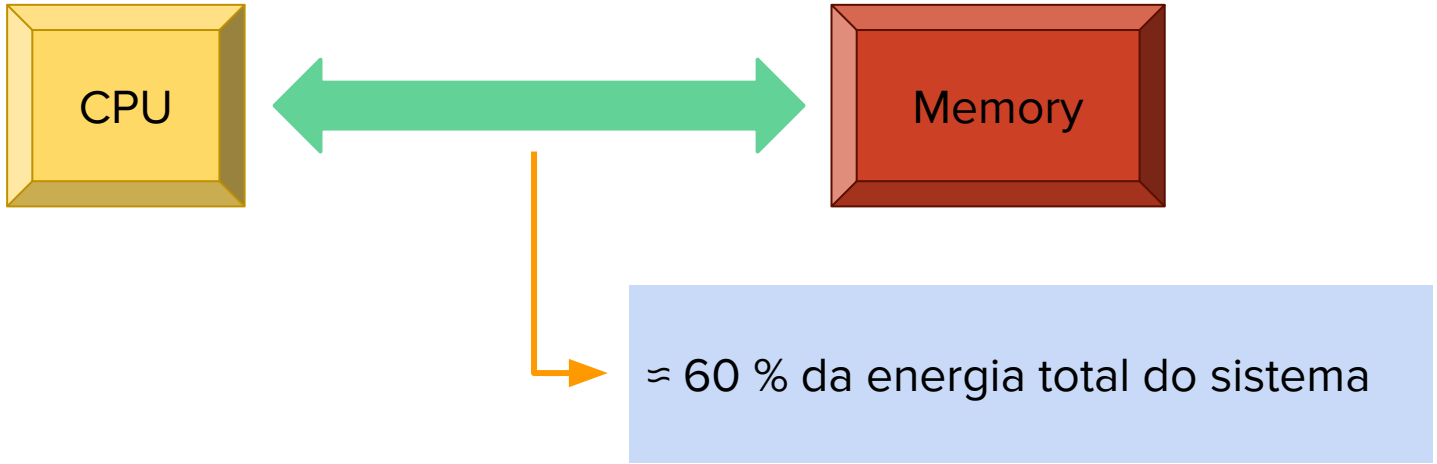
Conversão dinâmica de instruções para processamento vetorial em memória

Rodrigo Machniewicz Sokulski
Orientador: Marco Antonio Zanata Alves

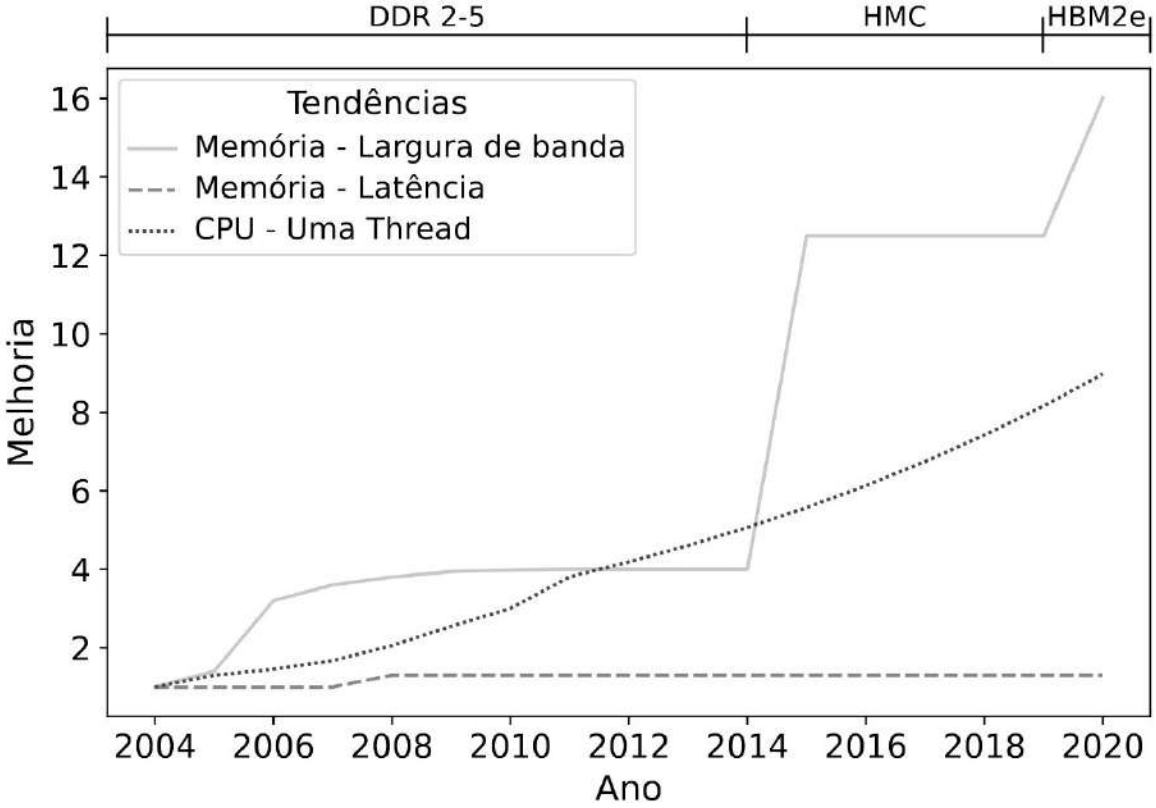
Motivação

Gargalo de von Neumann

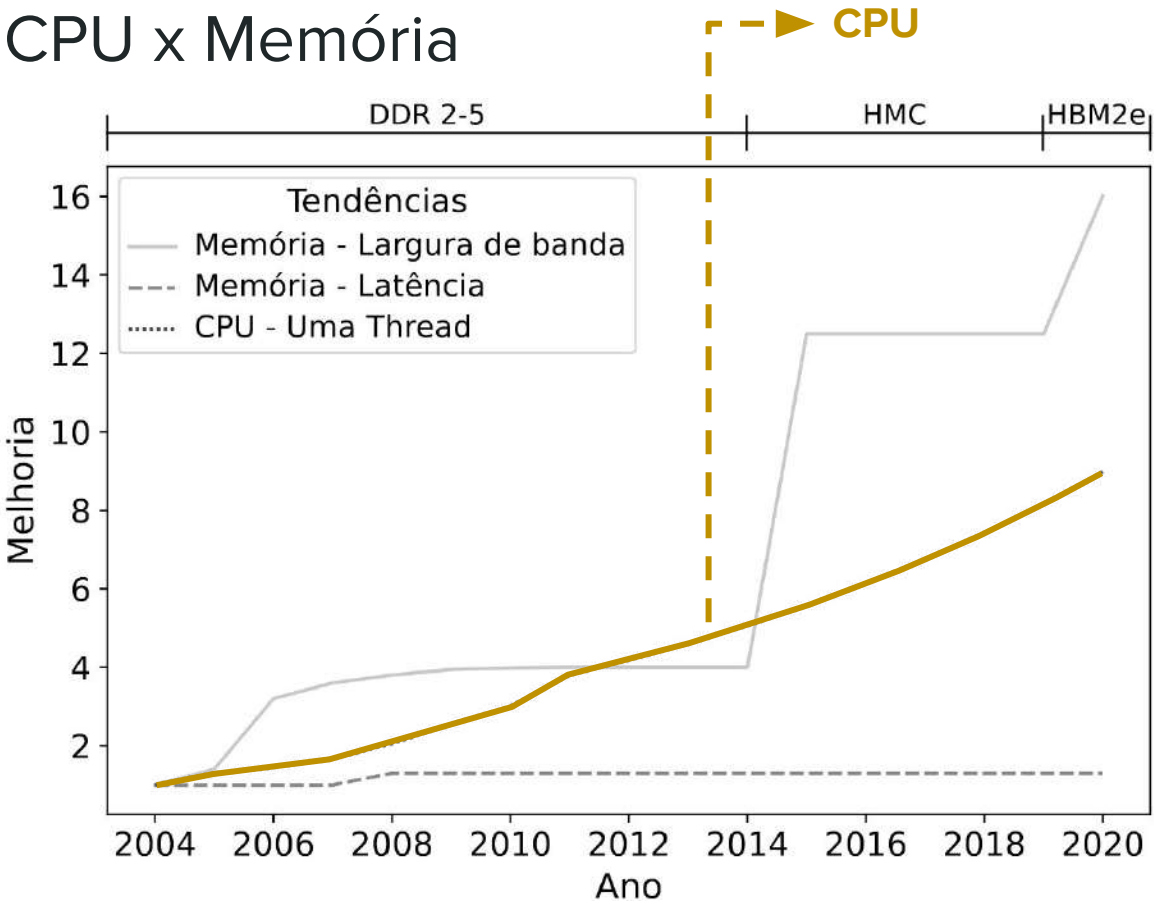


Boroumand et al. 2018 Google workloads for consumer devices: Mitigating data movement bottlenecks.
In Proceedings of the Twenty-Third International Conference on Architectural Support for Programming Languages and Operating Systems

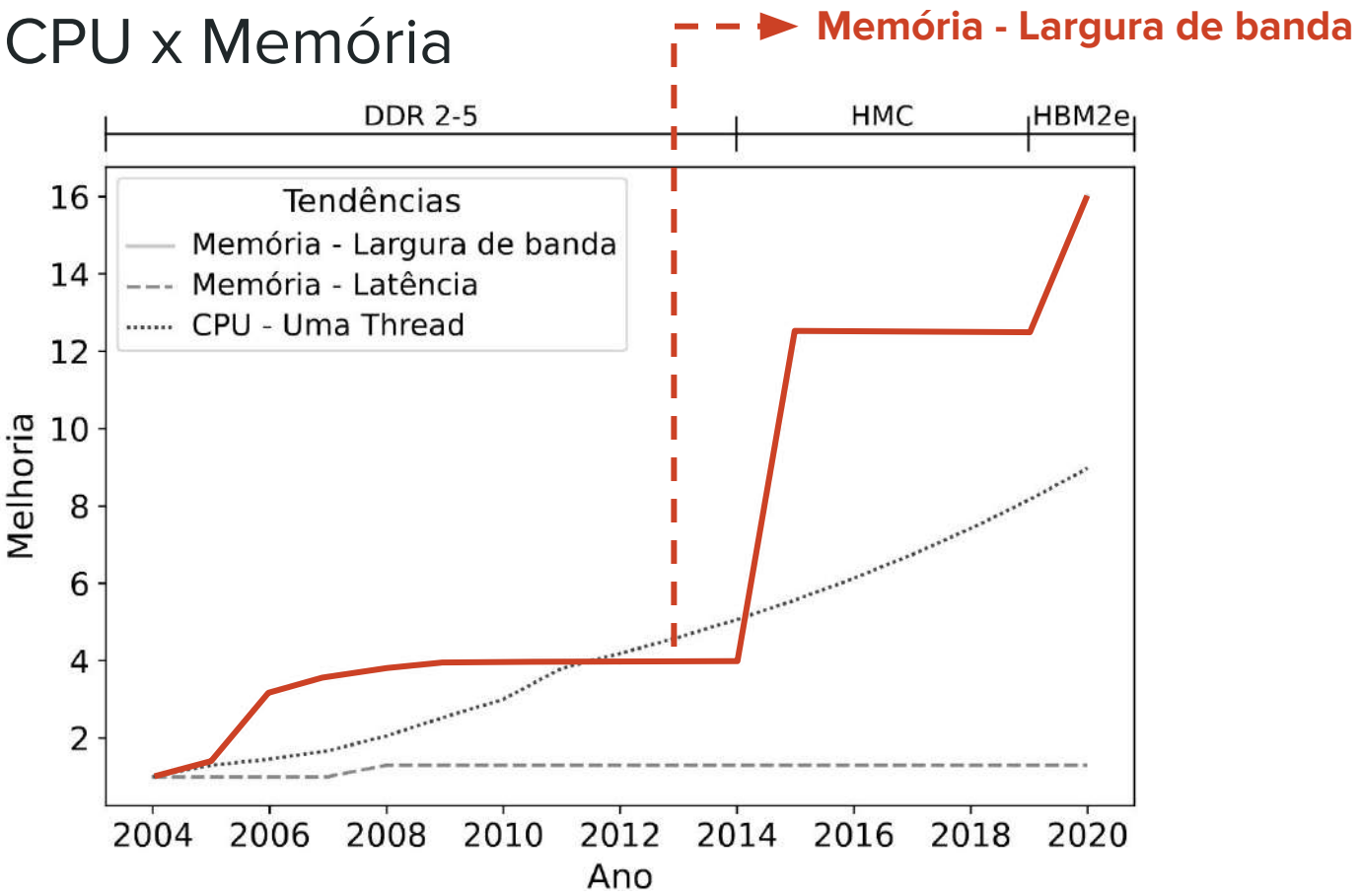
Melhorias CPU x Memória



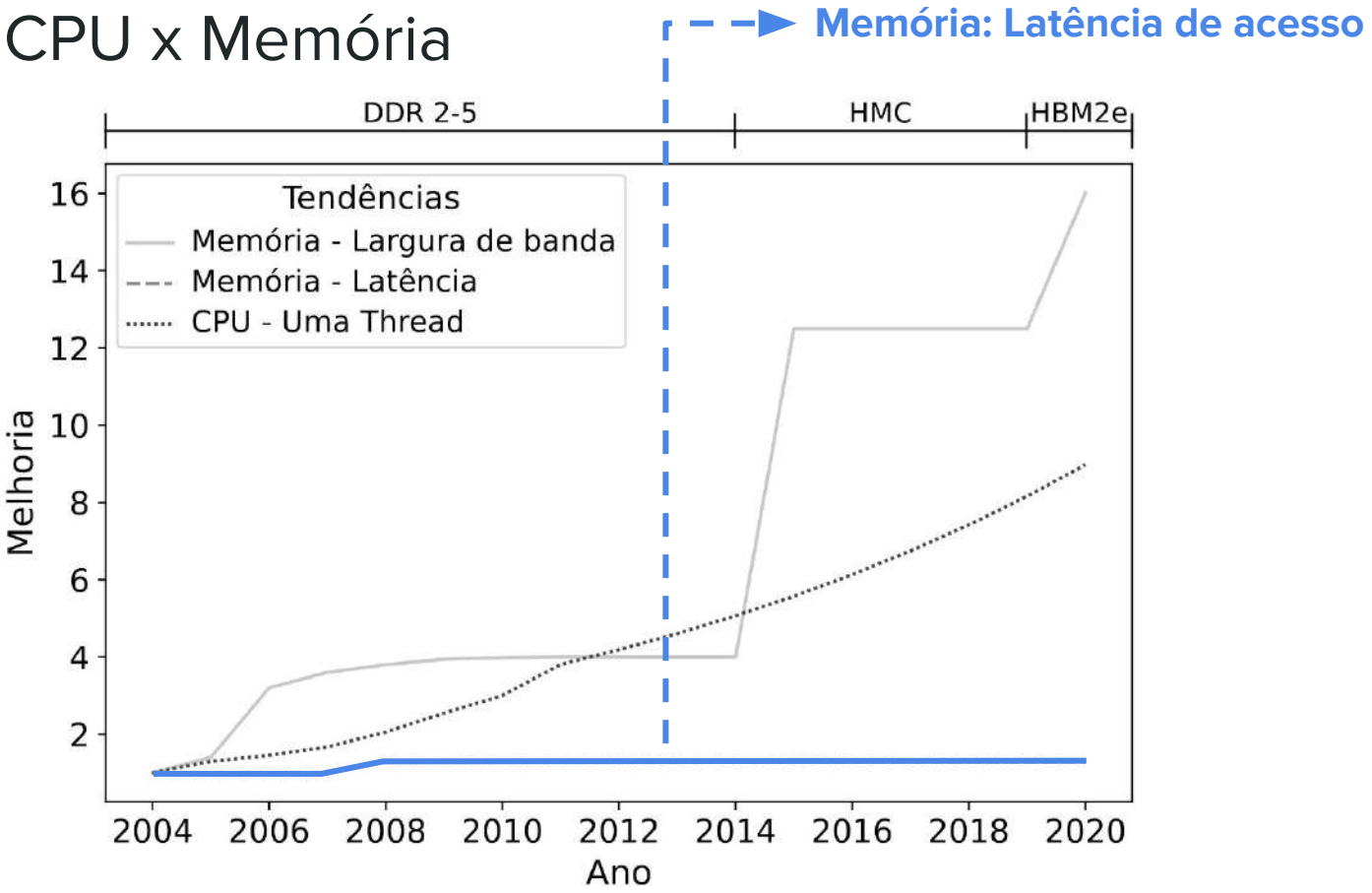
Melhorias CPU x Memória



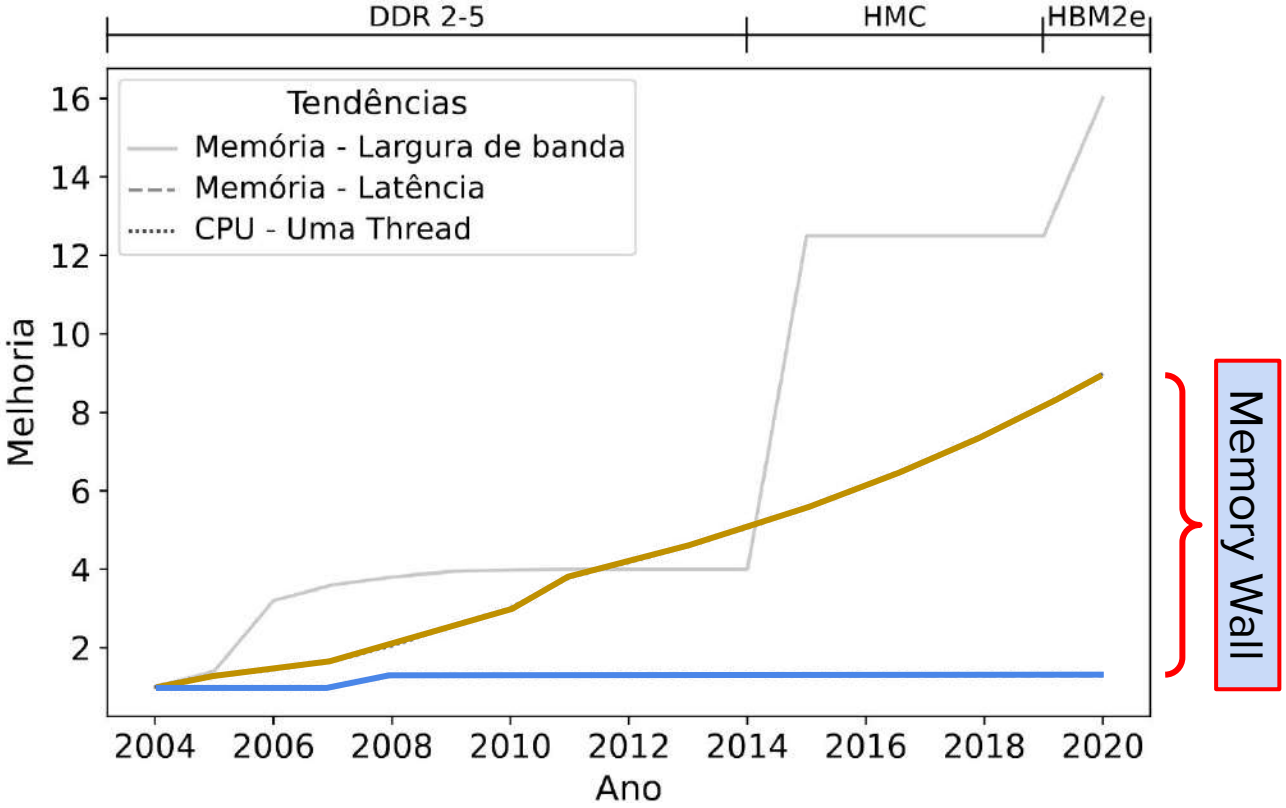
Melhorias CPU x Memória



Melhorias CPU x Memória



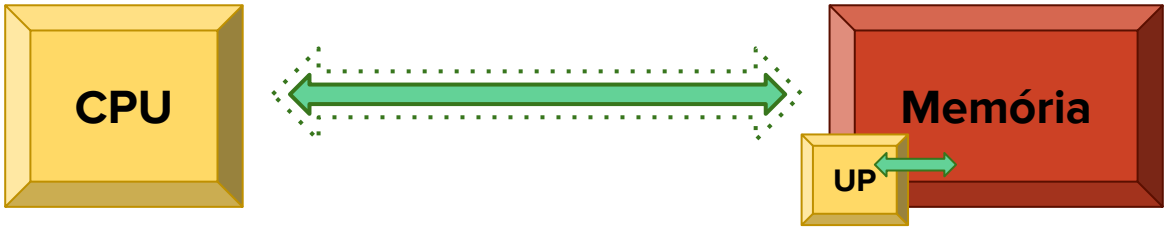
Melhorias CPU x Memória



Processing-In-Memory (PIM)

Insere unidades de processamento (UP) mais próximas ou dentro do módulo de memória

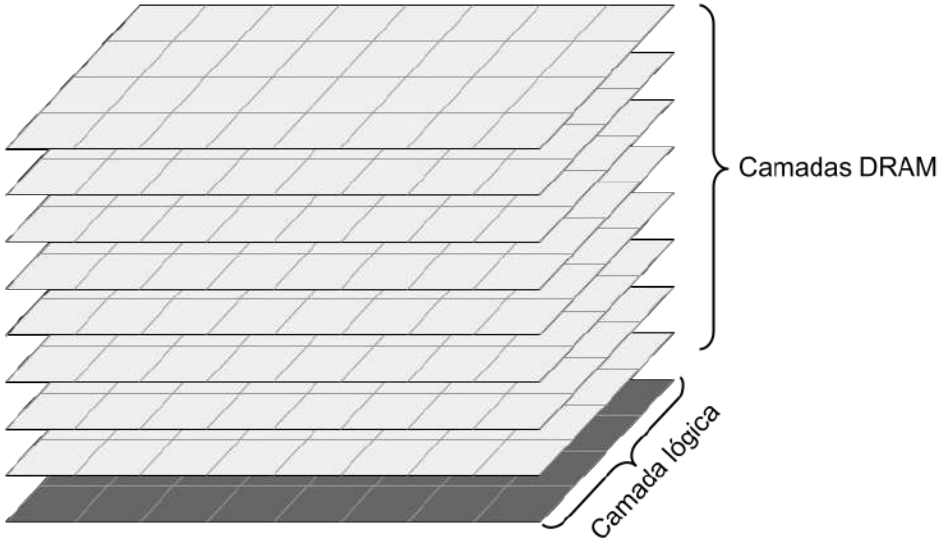
- Reduz o tráfego de dados entre a memória e o processador



Memórias 3D

Memória 3D (HMC/HBM):

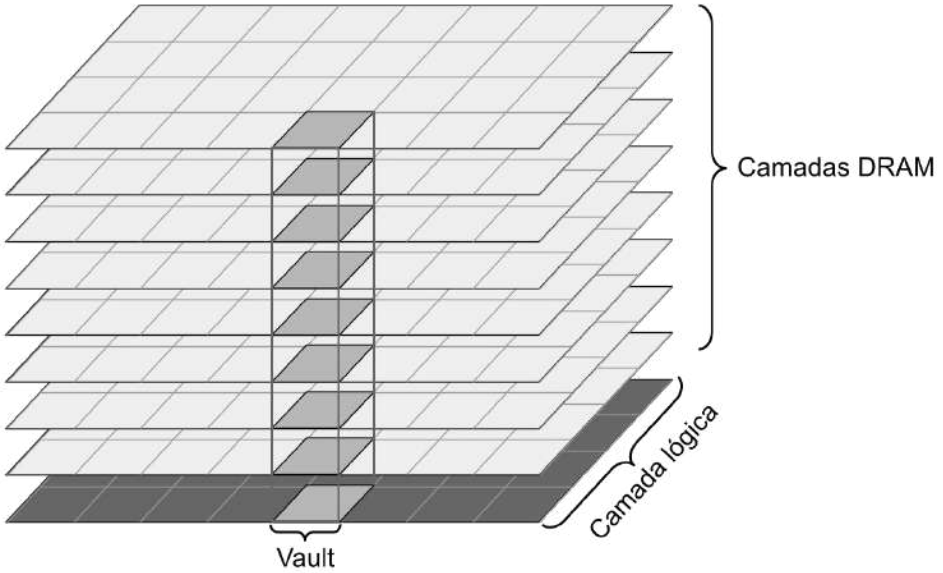
- Camadas de DRAM
- Camada lógica



Memórias 3D

Memória 3D (HMC/HBM):

- Camadas de DRAM
- Camada lógica
- Organizadas *vaults*
- Vários *banks* por *vault*

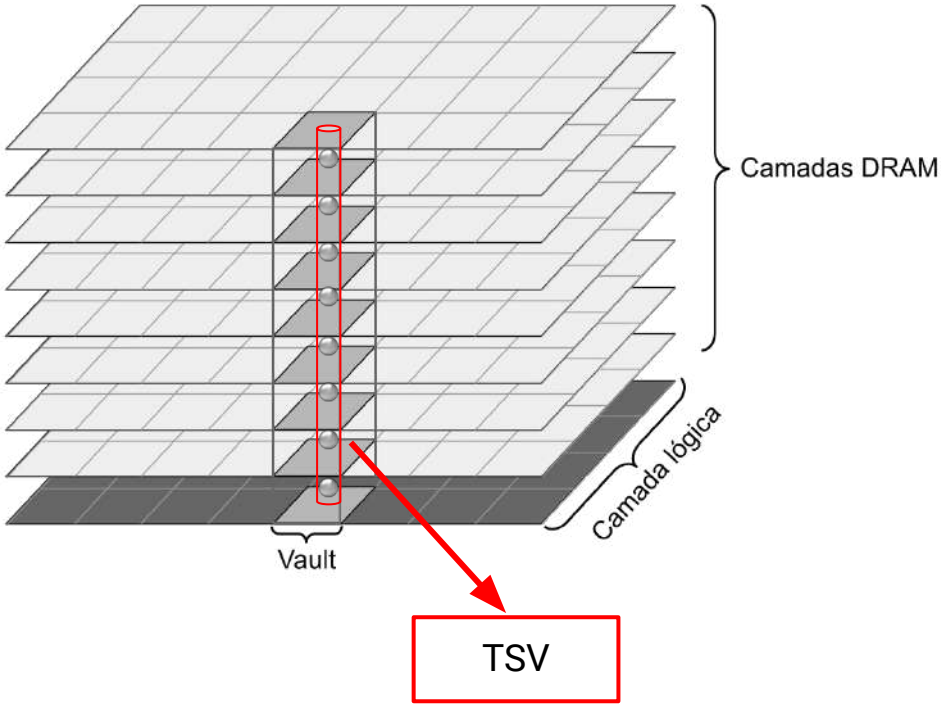


Memórias 3D

Memória 3D (HMC/HBM):

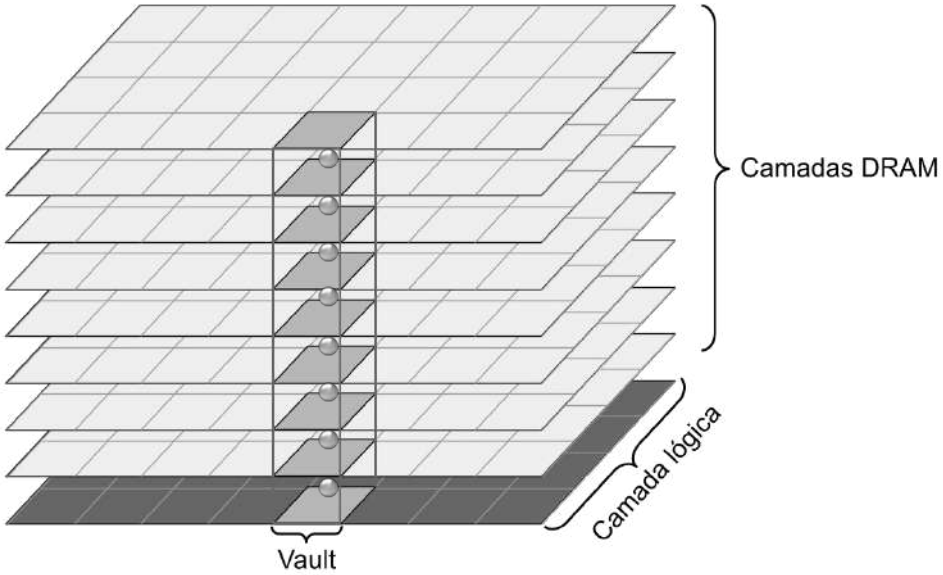
- Camadas de DRAM
- Camada lógica
- Organizadas *vaults*
- Vários *banks* por *vault*
- Conectadas por

  Through-Silicon Vias (TSVs)



Memórias 3D

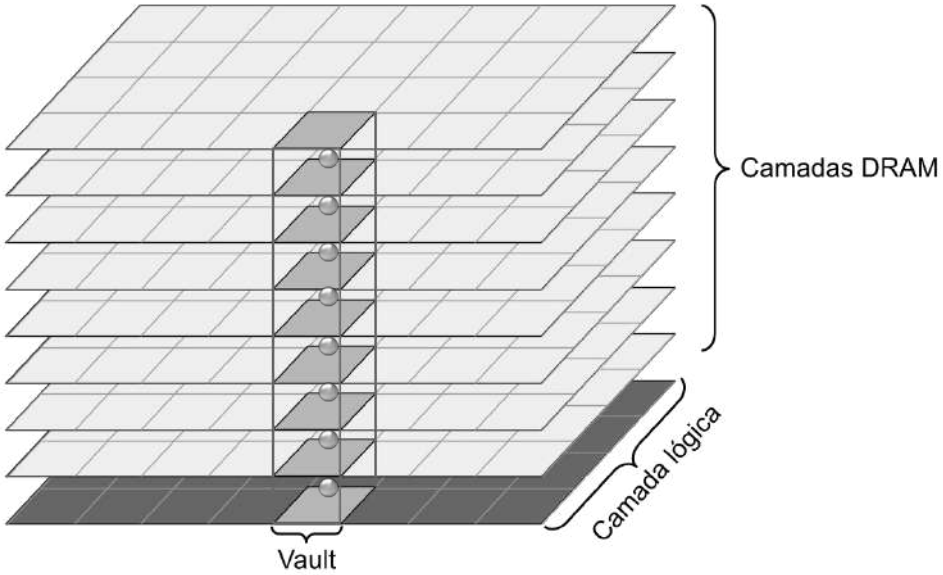
Versão	Largura de banda [1]
DDR5 - 4 canais (2020)	166 GiB/s (Baseline)
HMC (2015)	320 GiB/s (2x)
HBM2 (2018)	310 GiB/s (2x)
HBM2e (2020)	410 GiB/s (2,5x)



[1] Cordeiro, A. S. (2020). Porting machine learning algorithms to vector-in-memory architecture.
Dissertação de Mestrado, Pós-Graduação em Informática - Universidade Federal do Paraná,Curitiba - PR.

Memórias 3D

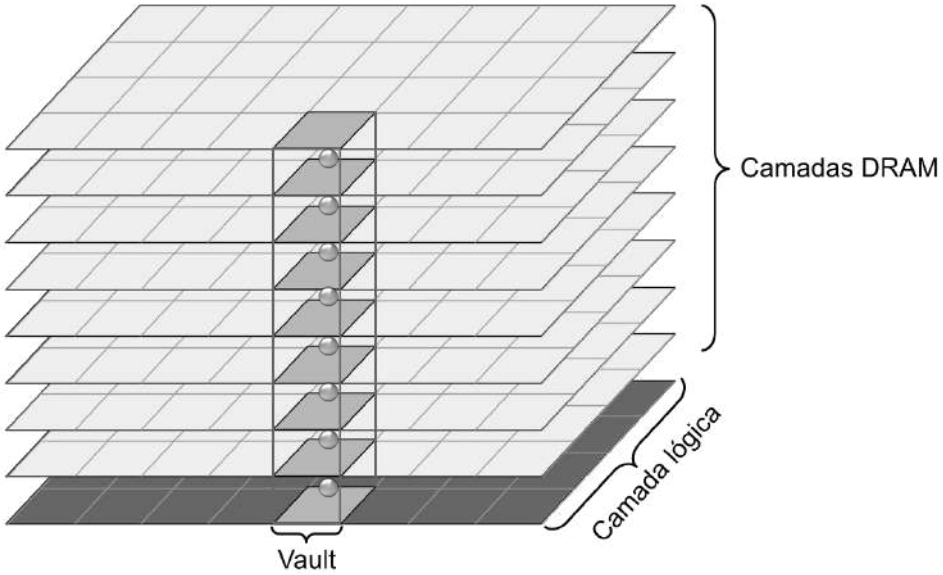
Versão	Largura de banda [1]
DDR5 - 4 canais (2020)	166 GiB/s (Baseline)
HMC (2015)	320 GiB/s (2x)
HBM2 (2018)	310 GiB/s (2x)
HBM2e (2020)	410 GiB/s (2,5x)



[1] Cordeiro, A. S. (2020). Porting machine learning algorithms to vector-in-memory architecture.
Dissertação de Mestrado, Pós-Graduação em Informática - Universidade Federal do Paraná,Curitiba - PR.

Memórias 3D

Versão	Largura de banda [1]
DDR5 - 4 canais (2020)	166 GiB/s (Baseline)
HMC (2015)	320 GiB/s (2x)
HBM2 (2018)	310 GiB/s (2x)
HBM2e (2020)	410 GiB/s (2,5x)

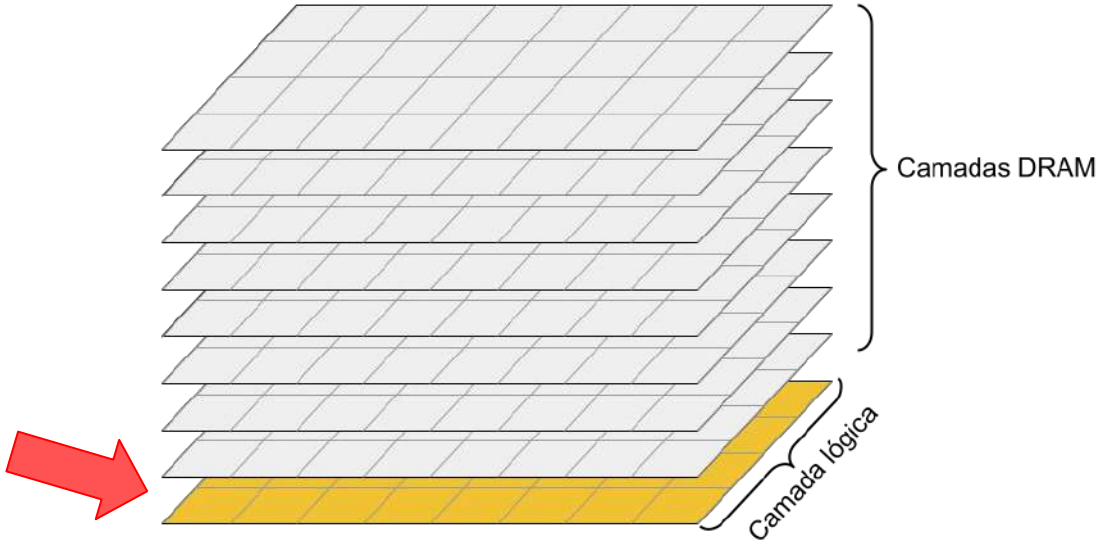


[1] Cordeiro, A. S. (2020). Porting machine learning algorithms to vector-in-memory architecture.
Dissertação de Mestrado, Pós-Graduação em Informática - Universidade Federal do Paraná,Curitiba - PR.

Memórias 3D

Memória 3D (HMC/HBM):

- PIMs podem ser incluídos na camada lógica
- Acesso direto ao paralelismo da memória 3D



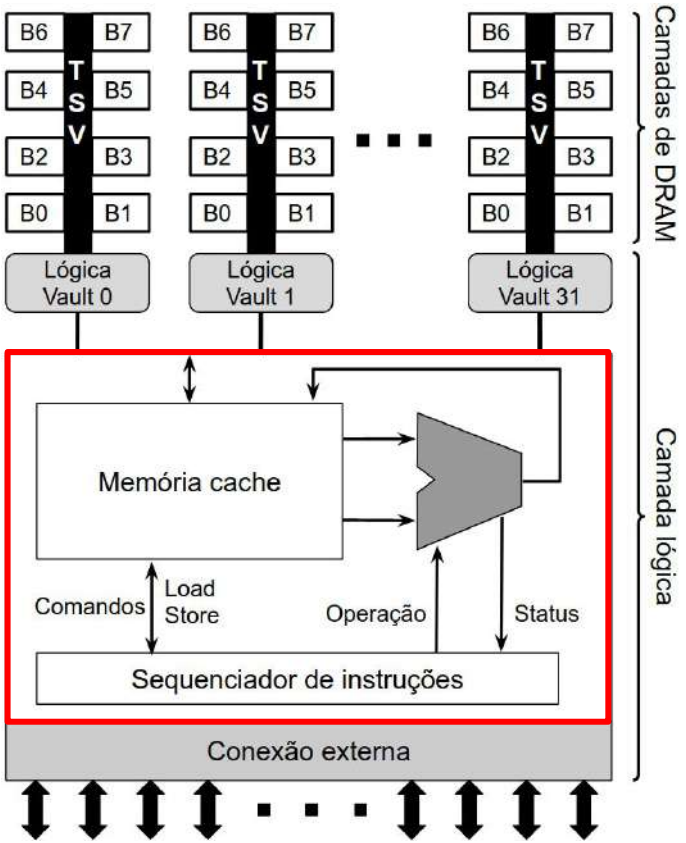
Vector-In-Memory Architecture (VIMA)

Vector-In-Memory Architecture (VIMA)

A VIMA estende a ISA do sistema

Implementa instruções vetoriais na camada lógica de memórias 3D

- Unidades funcionais
- Memória cache de dados
- Sequenciador de instruções

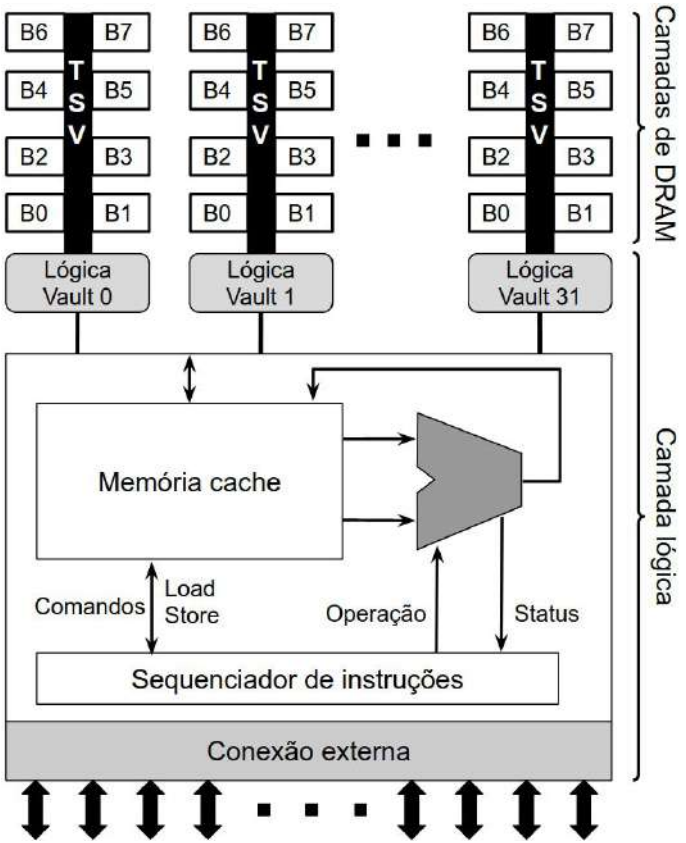


Vector-In-Memory Architecture (VIMA)

A VIMA estende a ISA do sistema

Implementa instruções vetoriais na camada lógica de memórias 3D

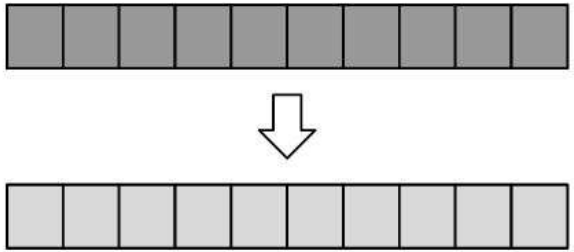
- Unidades funcionais
- Memória cache de dados
- Sequenciador de instruções
- Banco de registradores



Categorias de Instruções VIMA

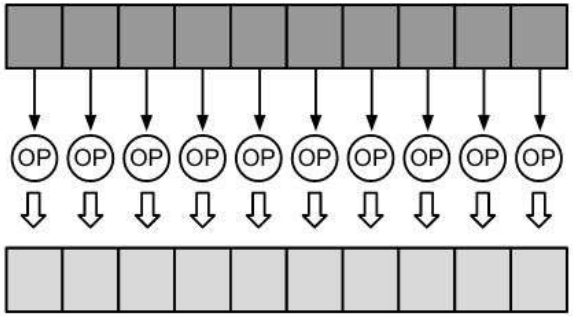
Cópia de memória

(I)



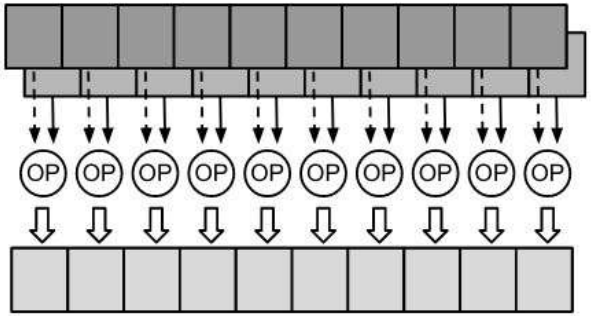
Operação sobre vetor

(II)



Operação entre vetores

(III)

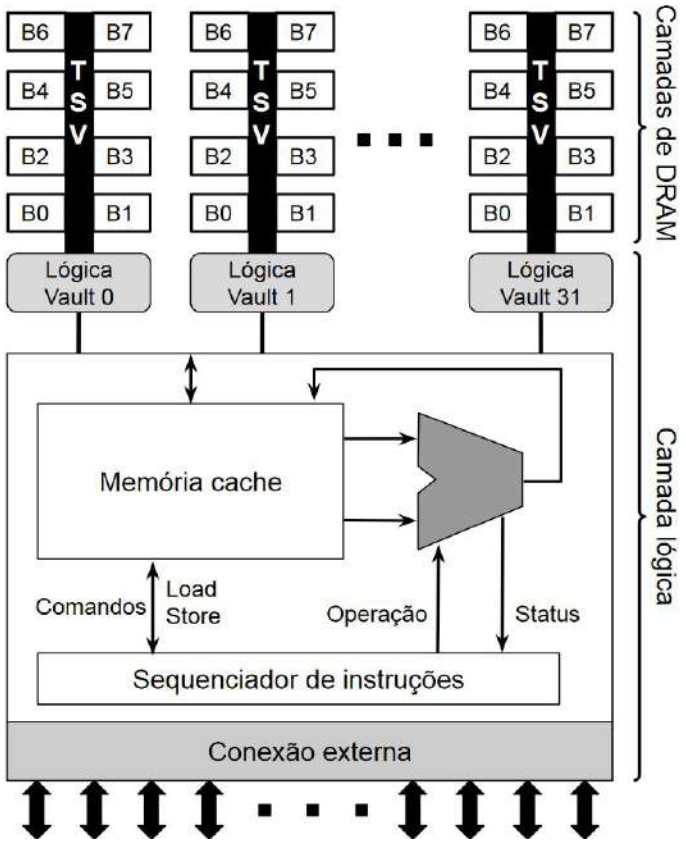


Instruções VIMA

Considerando uma memória 3D:

- 32 *vaults*
- Cada *vault* com 8 *banks* independentes
- *Row buffer* de 256 B

Os operandos das instruções VIMA variam de **256 B** a **8 KB** (paralelismo dos 32 *vaults*).

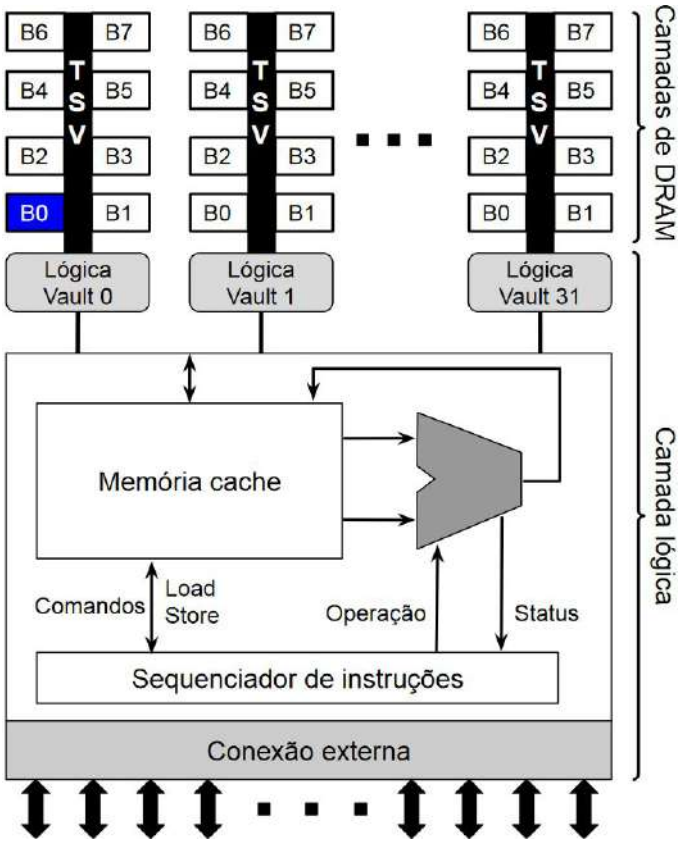


Instruções VIMA

Considerando uma memória 3D:

- 32 *vaults*
- Cada *vault* com 8 *banks* independentes
- *Row buffer* de 256 B

Os operandos das instruções VIMA variam de **256 B** a **8 KB** (paralelismo dos 32 *vaults*).

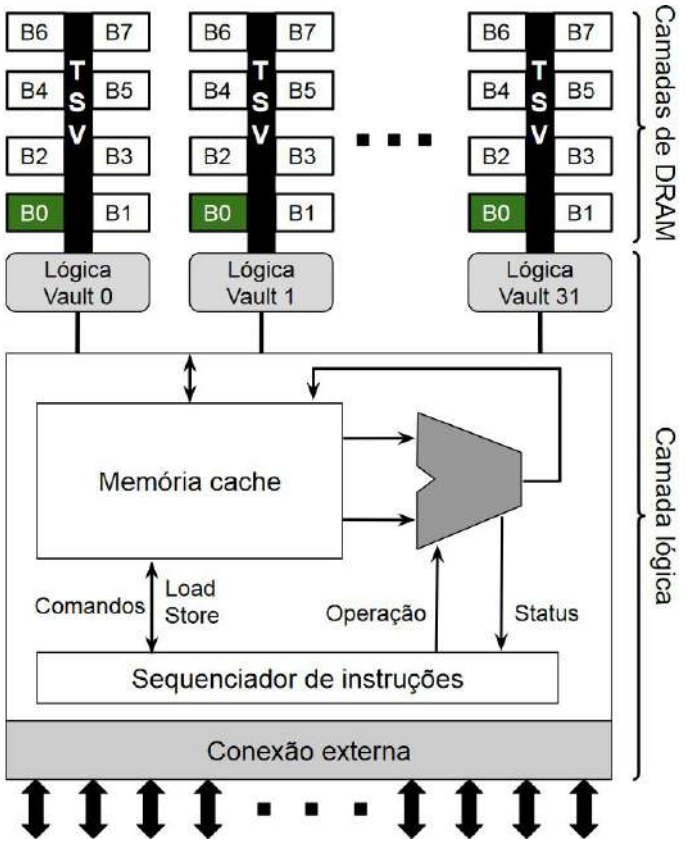


Instruções VIMA

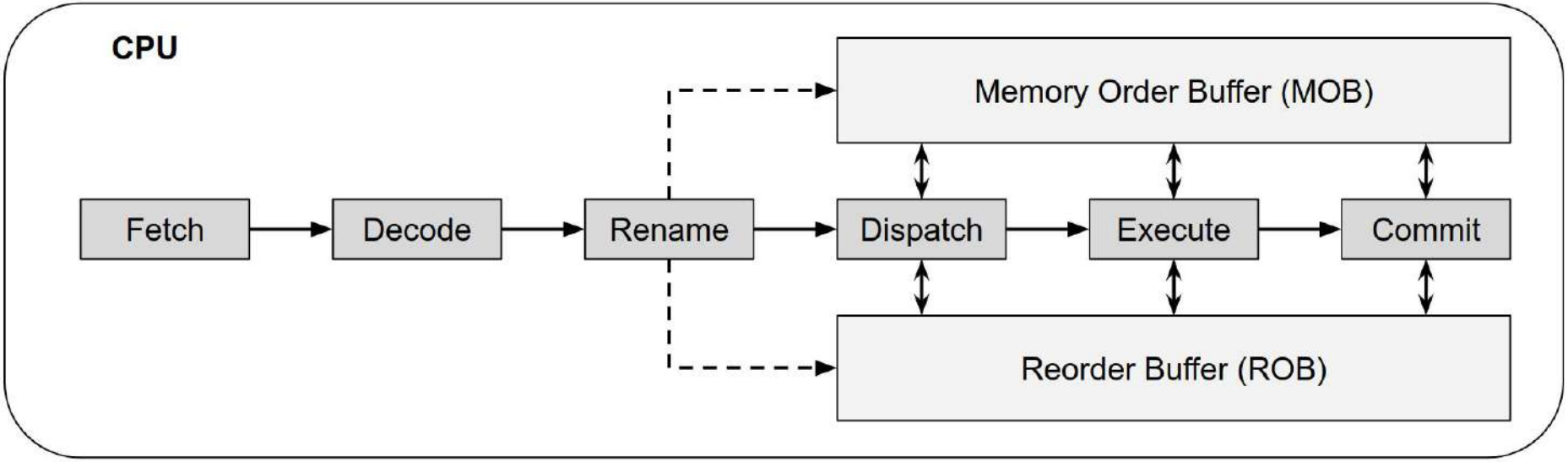
Considerando uma memória 3D:

- 32 *vaults*
- Cada *vault* com 8 *banks* independentes
- *Row buffer* de 256 B

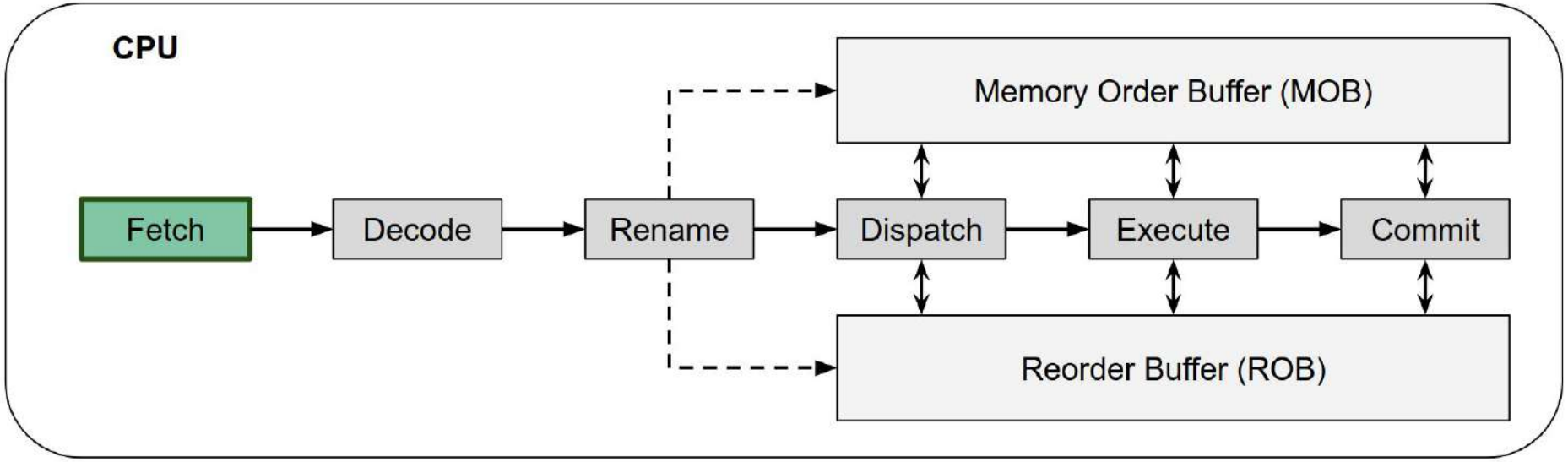
Os operandos das instruções VIMA variam de **256 B** a **8 KB** (paralelismo dos 32 *vaults*).



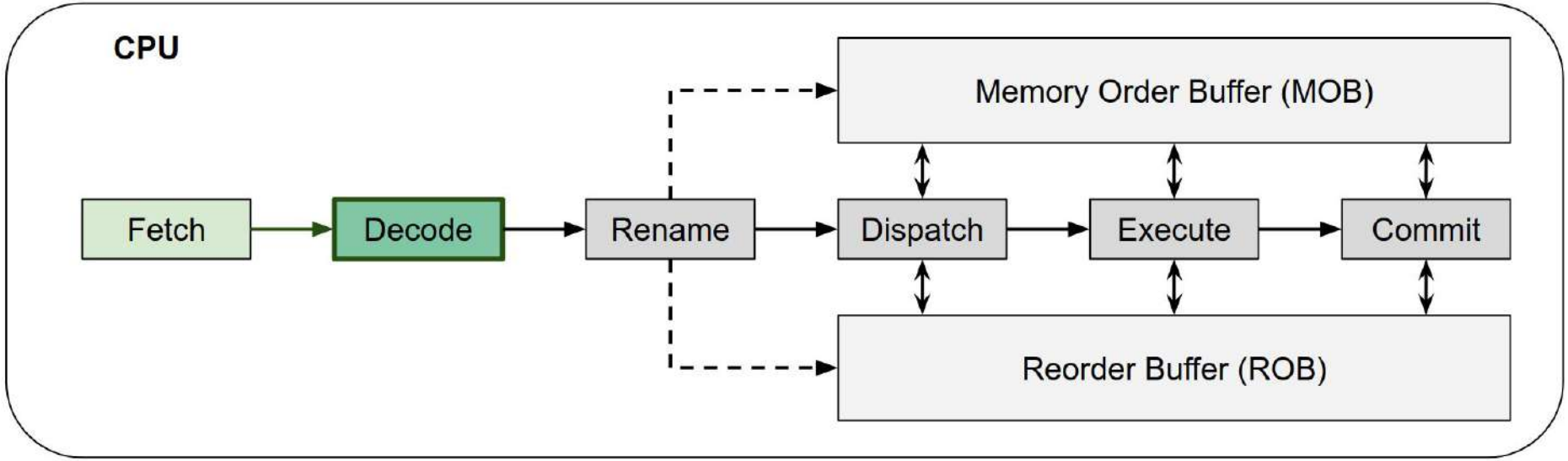
Instruções VIMA



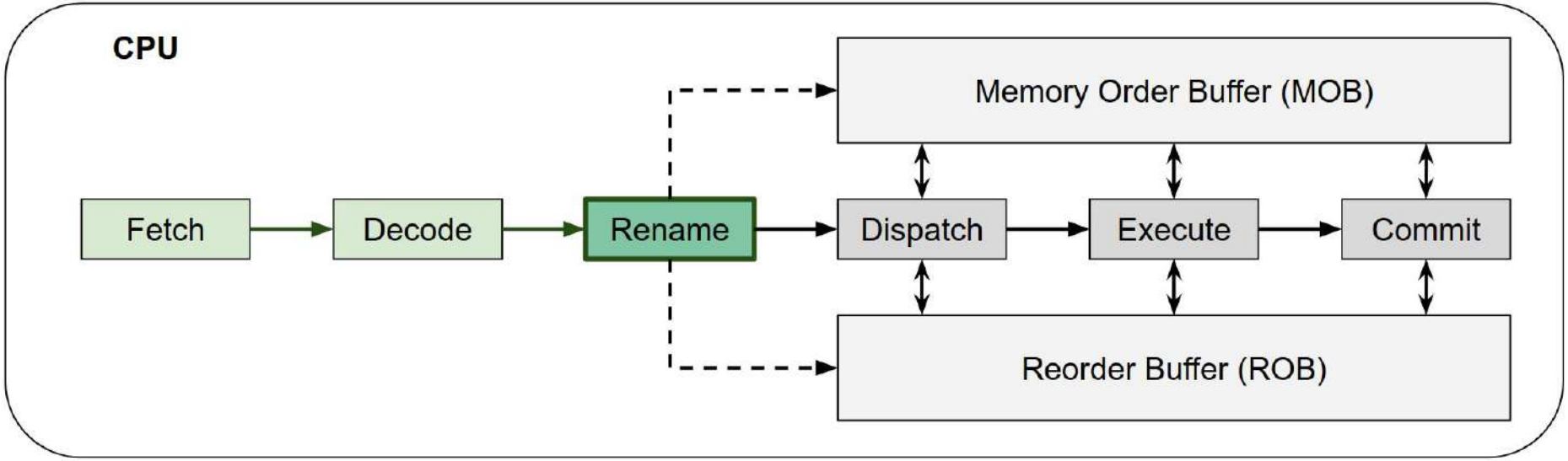
Instruções VIMA



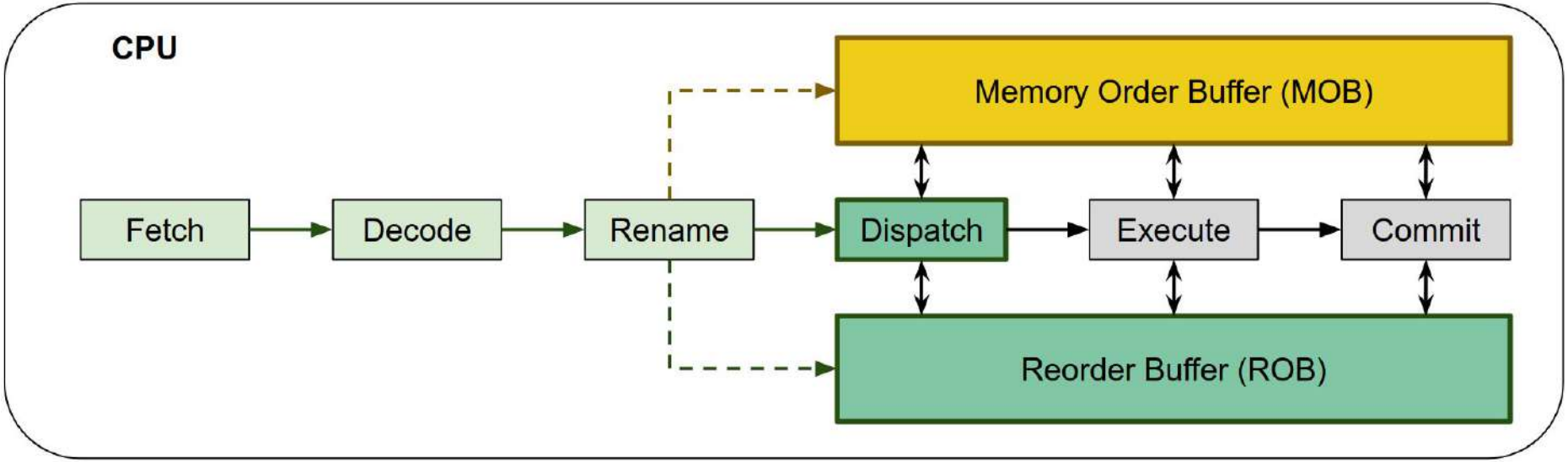
Instruções VIMA



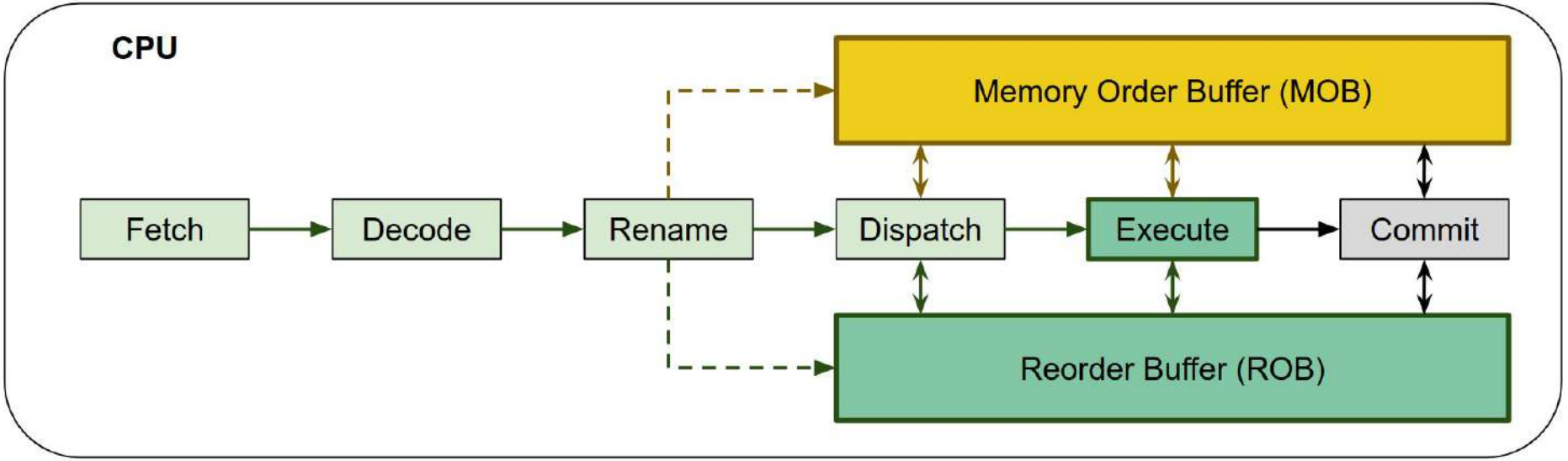
Instruções VIMA



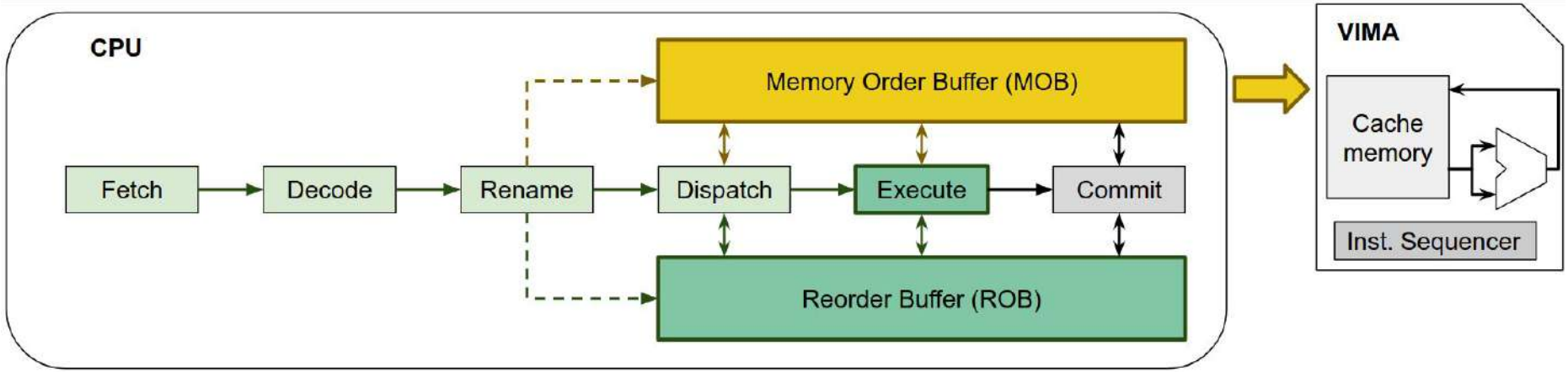
Instruções VIMA



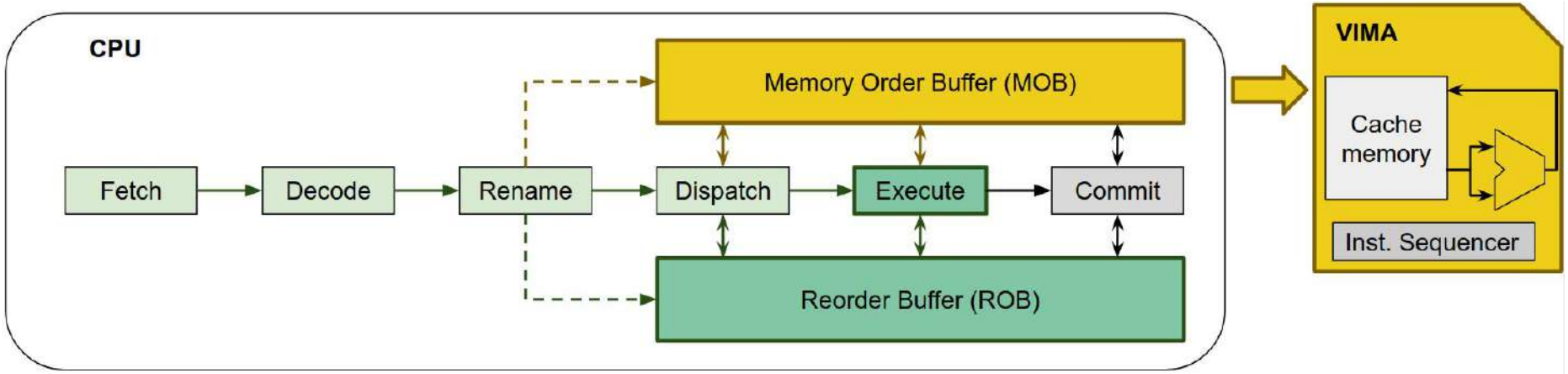
Instruções VIMA



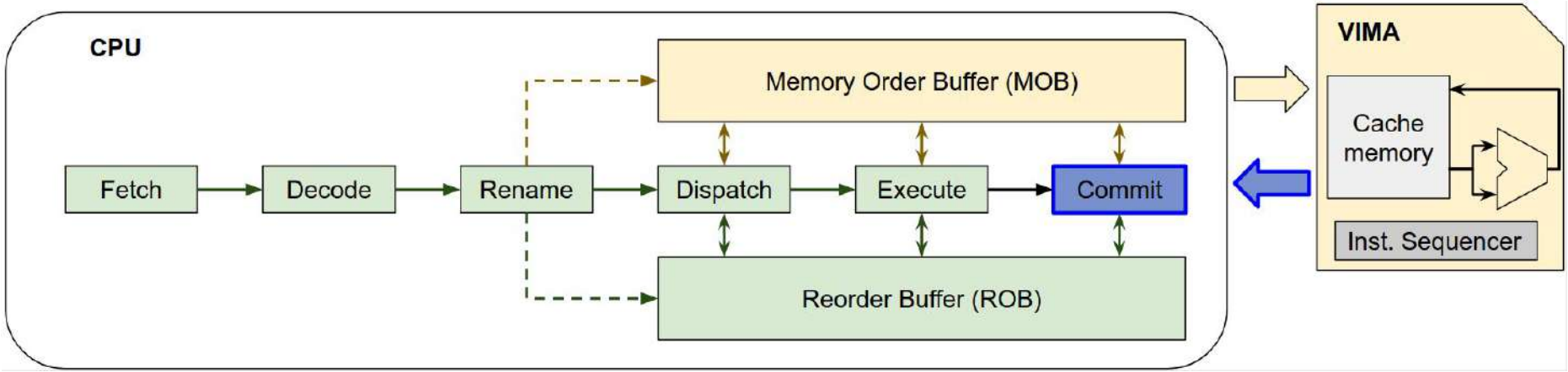
Instruções VIMA



Instruções VIMA



Instruções VIMA



Adoção das Instruções VIMA

Instruções VIMA podem ser adotadas de diversas maneiras:

- Rotinas *intrinsic*
- Bibliotecas adaptadas
- Conversões na compilação
- Conversões durante a execução

Adoção das Instruções VIMA

Instruções VIMA podem ser adotadas de diversas maneiras:

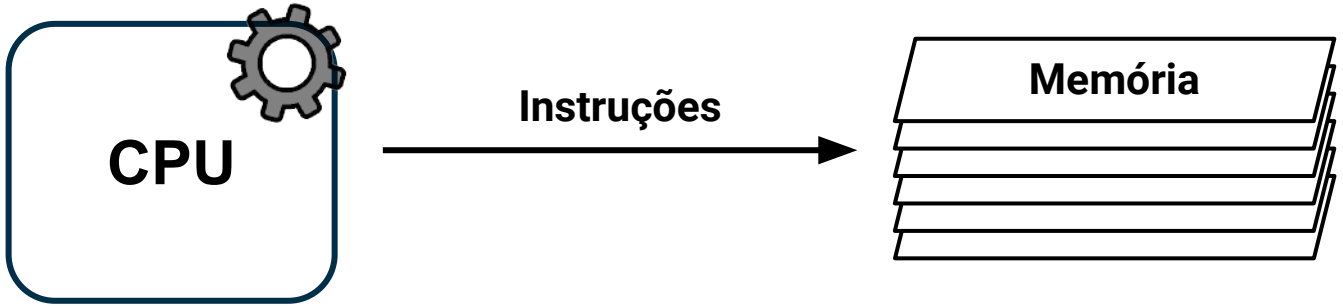
- Rotinas *intrinsic*
- Bibliotecas adaptadas
- Conversões na compilação
- Conversões durante a execução

Até onde sabemos, não existem mecanismos de conversão dinâmica para instruções PIM vetoriais

Pergunta de Pesquisa

Pergunta de Pesquisa

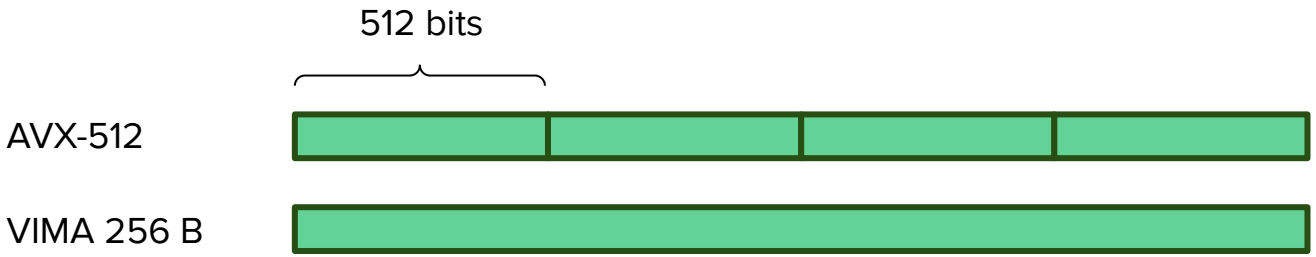
Auto-vetorização em hardware para processamento em memória pode levar a ganhos de desempenho?



Proposta

Proposta

- **Propomos o** *Simple AVX to PIM Vectorizer* (SAPIVe)
  - Mecanismo de tradução binária dinâmica em *hardware*
  - De instruções vetoriais da **CPU**
  - Para instruções vetoriais **VIMA**



Proposta

- **Propomos o** *Simple AVX to PIM Vectorizer* (SAPIVe)
  - Mecanismo de tradução binária dinâmica em hardware
  - De instruções vetoriais da **CPU**
  - Para instruções vetoriais **VIMA**

- Reduz o **gargalo** entre a CPU e a Memória
- Reduzindo o **tempo de execução**

Simple AVX to PIM Vectorizer (SAPIVe)

Ideia do SAPIVe

- Identificar padrões em instruções de **CPU**
  - Cópia de memória
  - Operação sobre vetor
  - Operação entre vetores
- Substituir com uma instrução **VIMA**

Ideia do SAPIVe

- Identificar padrões em instruções de **CPU**
- Substituir por instrução **VIMA**

```
for (...) {
  Vc[i] = Va[i] + Vb[i];
  ...
}
```

Soma vetorial

Ideia do SAPIVe

- Identificar padrões em instruções de **CPU**
- Substituir por instrução **VIMA**

```
for (...) {
  Vc[i] = Va[i] + Vb[i];
  …
}
```

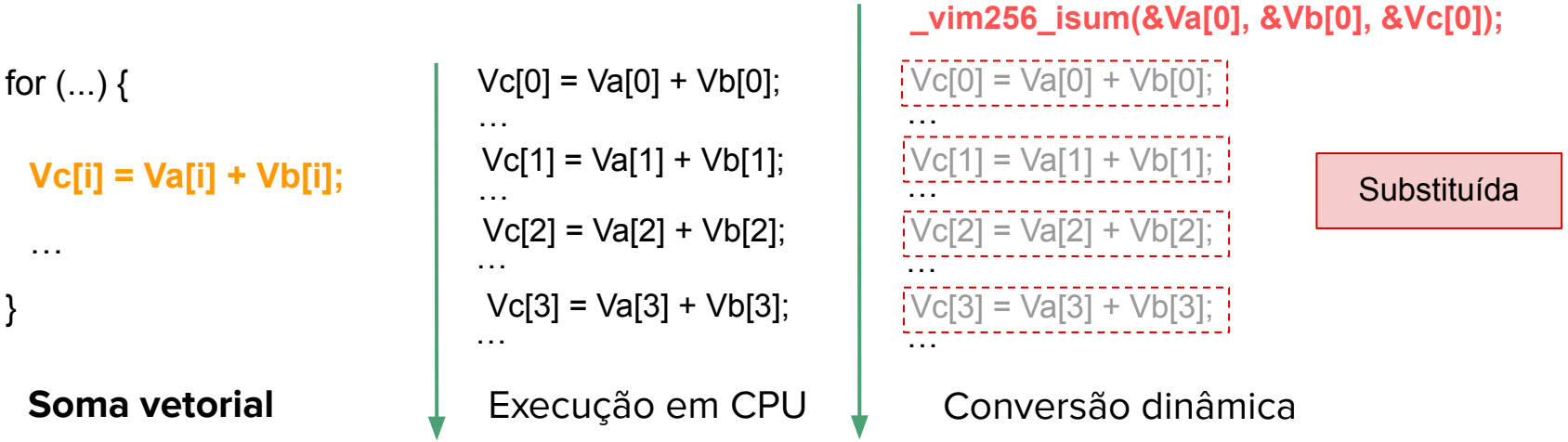
Soma vetorial

```
Vc[0] = Va[0] + Vb[0];
…
Vc[1] = Va[1] + Vb[1];
…
Vc[2] = Va[2] + Vb[2];
…
Vc[3] = Va[3] + Vb[3];
…
```

Execução em CPU

Ideia do SAPIVe

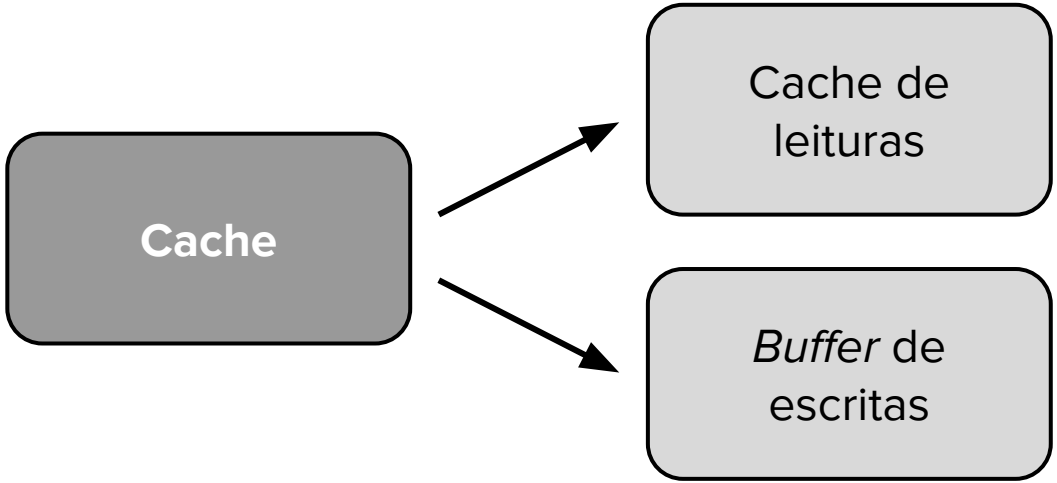
- Identificar padrões em instruções de **CPU**
- Substituir por instrução **VIMA**



Alterações na VIMA

Cache de dados separada em **cache de leitura** e ***buffer* de escritas**

Dados temporários do buffer de escritas são salvos em memória ou descartados



SAPIVe - Conversões

Cada conversão do SAPIVe passa pelas seguintes etapas:

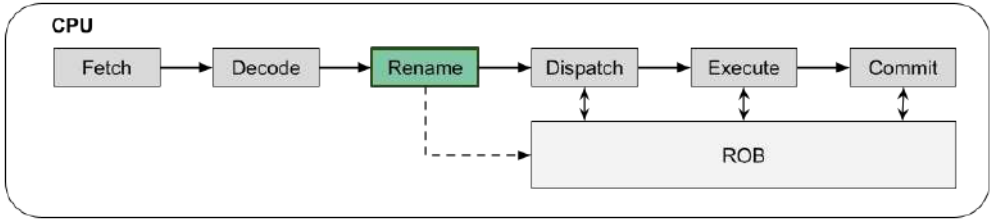
1. Identificação
2. Análise
3. Validação
4. Confirmação | Invalidação

SAPIVe - Conversões

Cada conversão do SAPIVe passa pelas seguintes etapas:

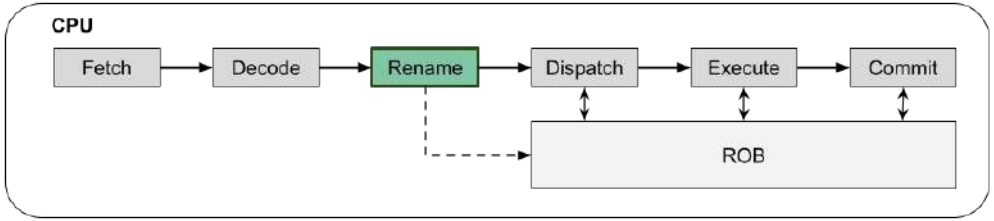
1. **Identificação:** Busca por padrões convertíveis
2. Análise
3. Validação
4. Confirmação | Invalidação

Identificação de Conjuntos

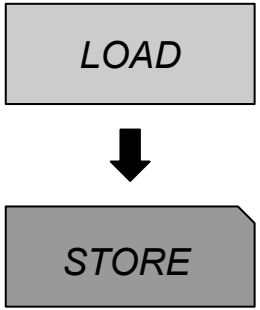


Objetivo: Avaliar dependências entre micro-ops renomeadas em busca de laços com padrões de instruções que podem ser convertidos

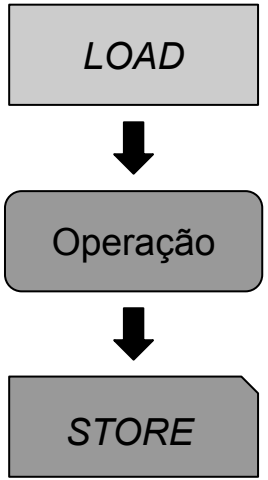
Identificação de Conjuntos



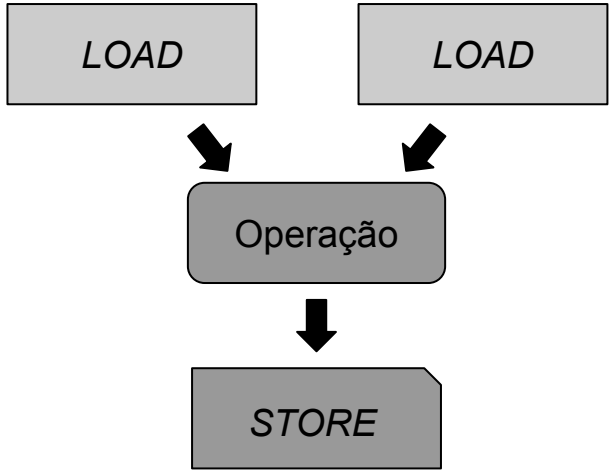
Objetivo: Avaliar dependências entre micro-ops renomeadas em busca de laços com padrões de instruções que podem ser convertidos



(I) Cópia de memória



(II) Operação sobre vetor



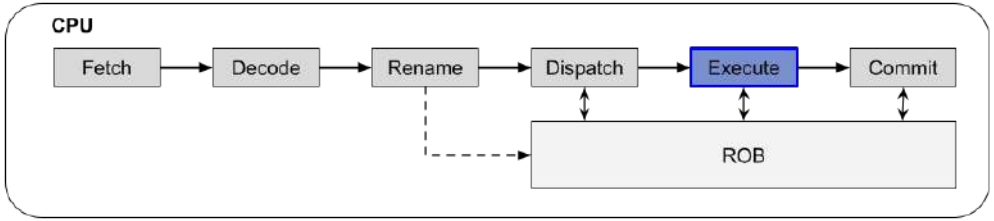
(III) Operação entre vetores

SAPIVe - Conversões

Cada conversão do SAPIVe passa pelas seguintes etapas:

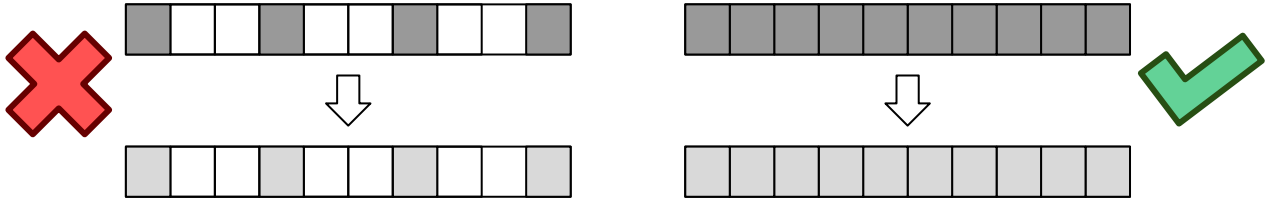
1. Identificação
2. **Análise:** Avalia os acessos à memória do padrão
3. Validação
4. Confirmação | Invalidação

Análise de Candidatos



Objetivo: Verificar se os acessos à memória do padrão são contíguos

Address Generation Unit (AGU) obtém os endereços de acesso à memória das duas primeiras iterações do padrão



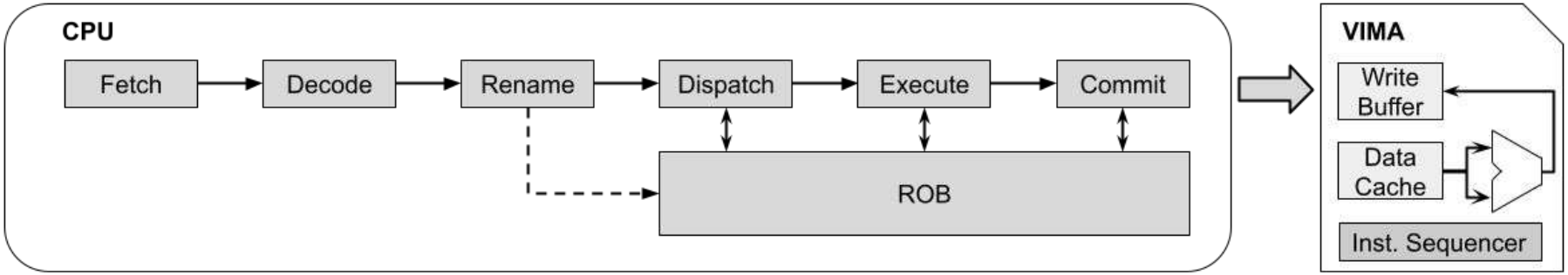
SAPIVe - Conversões

Cada conversão do SAPIVe passa pelas seguintes etapas:

1. Identificação
2. Análise
3. **Validação**: Converte instruções da CPU em instruções VIMA
4. Confirmação | Invalidação

Validação de Conversão

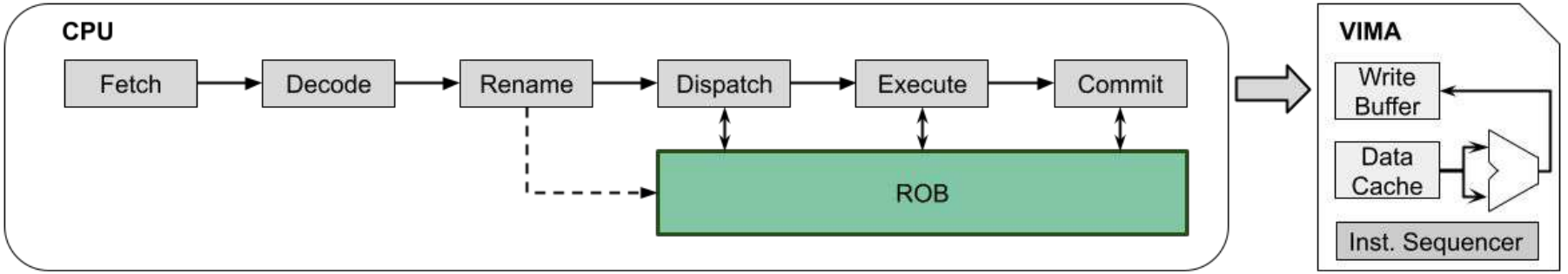
Objetivo: Executar uma instrução VIMA equivalente, enquanto a corretude da conversão é verificada



Validação de Conversão

Objetivo: Executar uma instrução VIMA equivalente, enquanto a corretude da conversão é verificada

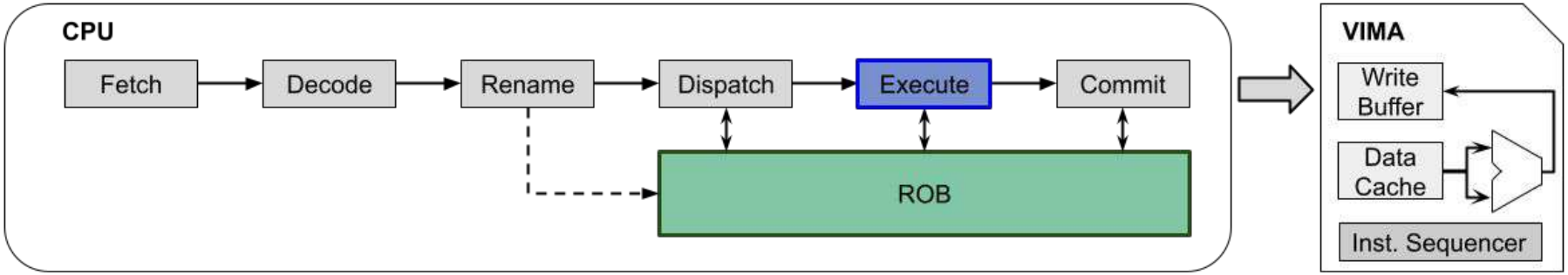
- Iterações do padrão são bloqueadas no ROB sem executar



Validação de Conversão

Objetivo: Executar uma instrução VIMA equivalente, enquanto a corretude da conversão é verificada

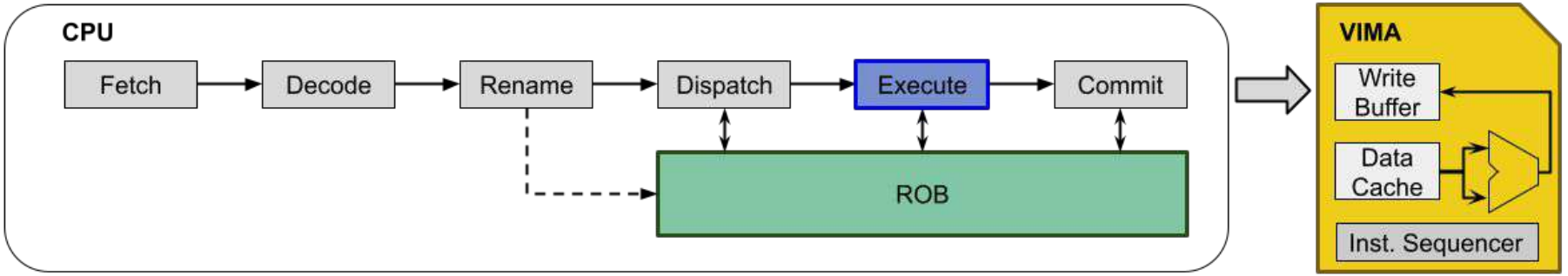
- Iterações do padrão são bloqueadas no ROB sem executar
- A AGU calcula os endereços de acesso à memória de cada iteração



Validação de Conversão

Objetivo: Executar uma instrução VIMA equivalente, enquanto a corretude da conversão é verificada

- Iterações do padrão são bloqueadas no ROB sem executar
- A AGU calcula os endereços de acesso à memória de cada iteração
- Paralelamente, uma instrução VIMA correspondente é executada



SAPIVe - Conversões

Cada conversão do SAPIVe passa pelas seguintes etapas:

1. Identificação
2. Análise
3. Validação
4. **Confirmação** | **Invalidação**: Encerramento da conversão

Confirmação de Conversão

Objetivo: Armazenar de forma permanente os resultados da conversão, substituindo instruções da CPU

Requisitos:

- O padrão convertido tem iterações suficientes para a conversão

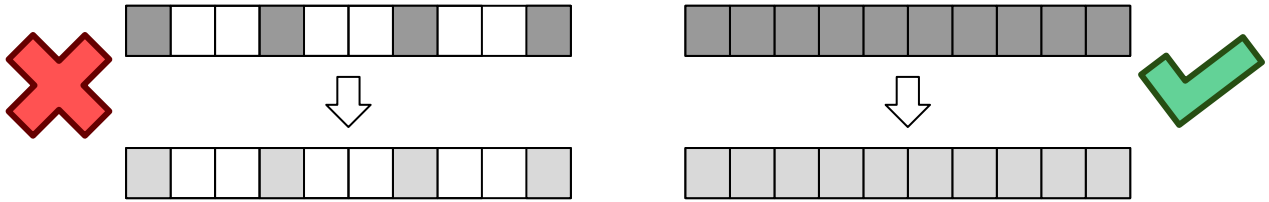
	VIMA-256 B	VIMA-512 B
AVX-256	8	16
AVX-512	4	8

Confirmação de Conversão

Objetivo: Armazenar de forma permanente os resultados da conversão, substituindo instruções da CPU

Requisitos:

- O padrão convertido tem iterações suficientes para a conversão
- Os acessos à memória permaneceram contíguos durante as iterações

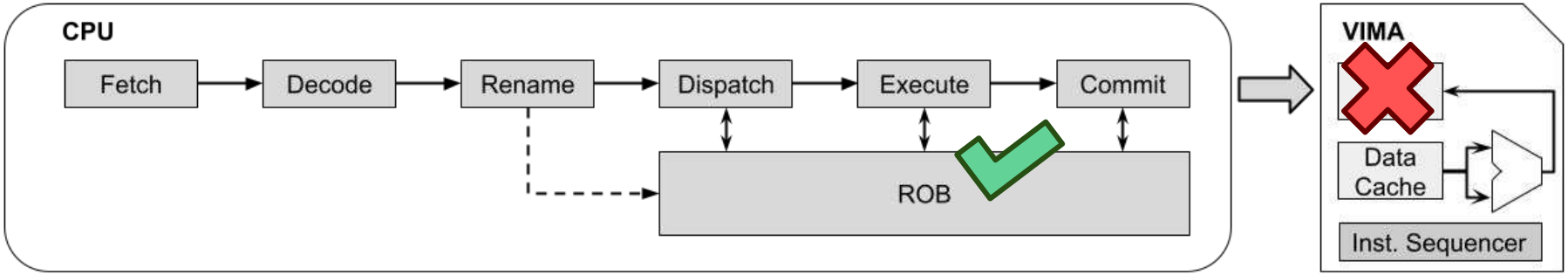


Invalidação de Conversão

Objetivo: Descartar os resultados da conversão, executando instruções bloqueadas no ROB

Requisitos:

- **Falha** da conversão



Otimizações

- **Lista de Conversões Falhas**
- **Conversões Consecutivas**
- **Agressividade de Adiantamentos**

Otimizações

Lista de Conversões Falhas

- Contém padrões que falham na primeira tentativa de conversão

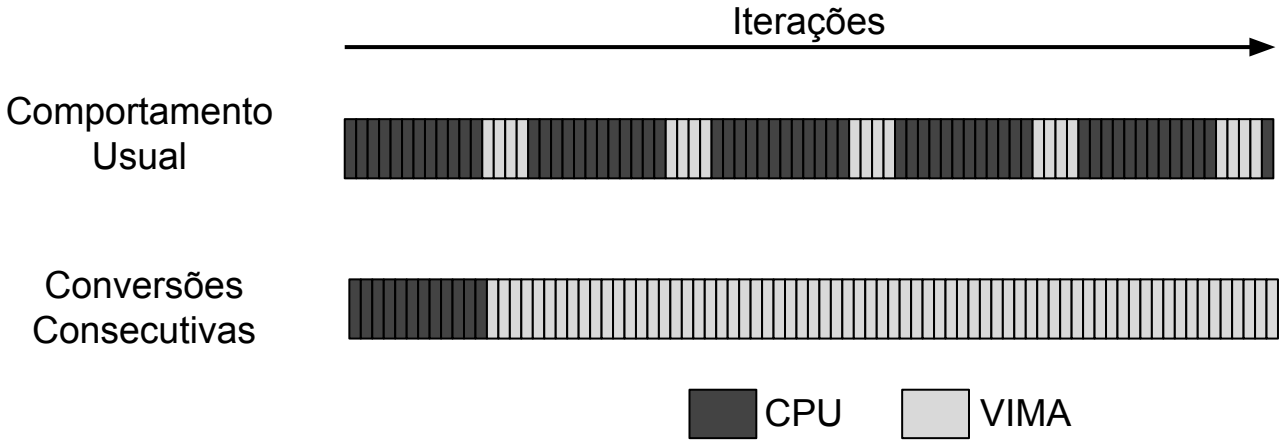
- Evita novas tentativas falhas de conversão

```
for (int i = 0; i < 3; ++i) {
        V[i] = T[i];
}
```

Otimizações

Conversões Consecutivas

- Após a primeira conversão em um laço, as próximas iterações dele são convertidas consecutivamente



Otimizações

Conversões Consecutivas

- Após a primeira conversão em um laço, as próximas iterações dele são convertidas consecutivamente

Agressividade de Adiantamentos

- Laços com conversões consecutivas adiantam conversões futuras
- A agressividade corresponde ao número de conversões adiantadas paralelamente

  [+] Melhora o paralelismo no acesso à memória

  [-] Exige mais espaço no ROB

Método

Método

- **Mecanismo:** *Simple AVX to PIM Vectorizer* (SAPIVe)

- **Simulador**: *Ordinary Computing Simulator* (OrCS) [1]
  - Orientado a traços de execução
  - Precisão de ciclo
  - Parâmetros inspirados no Intel Skylake [2]

[1] Köhler, R. (2019). Aceleração de cache misses prováveis através de requisições paralelas.
Dissertação de Mestrado, Pós-Graduação em Informática - Universidade Federal do Paraná, Curitiba - PR.
[2] Doweck, J., Kao, W.-F., Lu, A. K.-y., Mandelblat, J., Rahatekar, A., Rappoport, L., Rotem, E., Yasin, A. e Yoaz, A. (2017).
Inside 6th-generation intel core: New microarchitecture code-named skylake. IEEE Micro, 37(2):52–62.

Aplicações Ideais

- Cópia de memória
- Soma entre vetores

Método - Aplicações Ideais

- **Compilados** com instruções vetoriais
  - AVX-256
  - AVX-512

Método - Aplicações Ideais

- **Compilados** com instruções vetoriais
  - AVX-256
  - AVX-512
- **Traços de execução** de dois programas sintéticos
  - Soma entre vetores (vecsum)
  - Cópia de memória (memcpy)

Método - Aplicações Ideais

- **Compilados** com instruções vetoriais
  - AVX-256
  - AVX-512
- **Traços de execução** de dois programas sintéticos
  - Soma entre vetores
  - Cópia de memória
- **Convertidas** em
  - VIMA-256 B
  - VIMA-512 B
  - VIMA-1024 B

Método - Aplicações Ideais

- **Compilados** com instruções vetoriais
  - AVX-256
  - AVX-512
- **Traços de execução** de dois programas sintéticos
  - Soma entre vetores
  - Cópia de memória
- **Convertidas** em
  - **VIMA-256 B**
  - VIMA-512 B
  - VIMA-1024 B

Método - Aplicações Ideais

- **Compilados** com instruções vetoriais
  - AVX-256
  - AVX-512
- **Traços de execução** de dois programas sintéticos
  - Soma entre vetores
  - Cópia de memória
- **Convertidas** em
  - VIMA-256 B
  - VIMA-512 B
  - **VIMA-1024 B**

Experimentos Sobre as Aplicações Ideais

Para entender o comportamento do SAPIVe, avaliamos o impacto dos seguintes parâmetros:

- Tamanho da instrução AVX
- Tamanho da instrução VIMA
- Agressividade do mecanismo
- Número de iterações do laço convertido

Resultados - Aplicações Ideais

Impacto da Versão AVX Utilizada

Objetivo: Avaliar o impacto da versão de AVX utilizada em conversões do SAPIVe

	VIMA-256 B	VIMA-512 B	VIMA-1024 B
AVX-256	8	16	32
AVX-512	4	8	16

Impacto da Versão AVX Utilizada

Objetivo: Avaliar o impacto da versão de AVX utilizada em conversões do SAPIVe

	VIMA-256 B	VIMA-512 B	VIMA-1024 B
AVX-256	8	16	32
AVX-512	4	8	16

Impacto da Versão AVX Utilizada

Objetivo: Avaliar o impacto da versão de AVX utilizada em conversões do SAPIVe

	VIMA-256 B	VIMA-512 B	VIMA-1024 B
AVX-256	8	16	32
AVX-512	4	8	16

Impacto da Versão AVX Utilizada

Objetivo: Avaliar o impacto da versão de AVX utilizada em conversões do SAPIVe

	VIMA-256 B	VIMA-512 B	VIMA-1024 B
AVX-256	8	**16**	32
AVX-512	4	8	16

Impacto da Versão AVX Utilizada

Objetivo: Avaliar o impacto da versão de AVX utilizada em conversões do SAPIVe

	VIMA-256 B	VIMA-512 B	VIMA-1024 B
AVX-256	8	16	32
AVX-512	4	**8**	16

Impacto da Versão AVX Utilizada

Objetivo: Avaliar o impacto da versão de AVX utilizada em conversões do SAPIVe

	VIMA-256 B	VIMA-512 B	VIMA-1024 B
AVX-256	8	16	32
AVX-512	4	**8**	16

- Essa redução permite um maior número de conversões paralelas

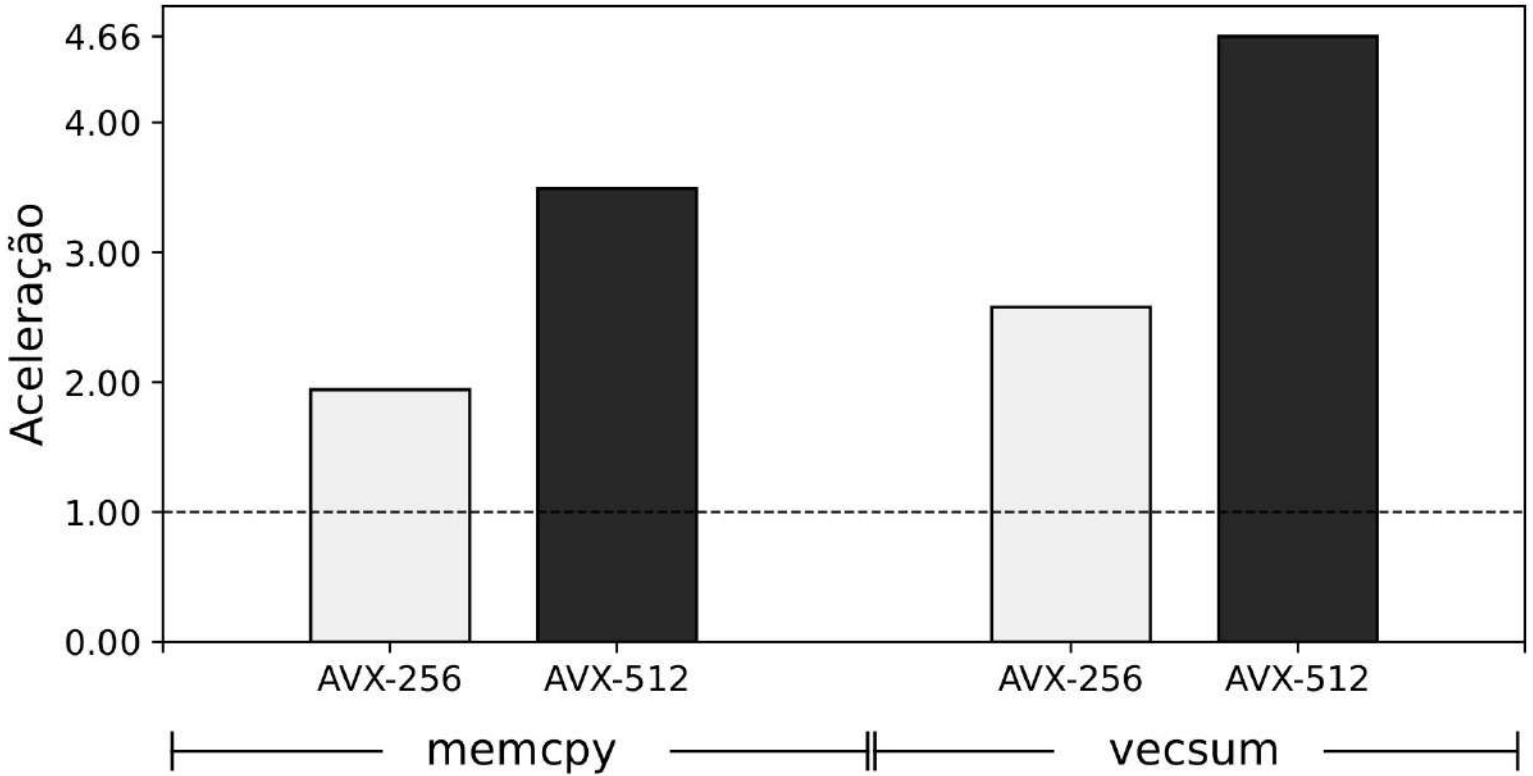
Impacto da Versão AVX Utilizada

Objetivo: Avaliar o impacto da versão de AVX utilizada em conversões do SAPIVe

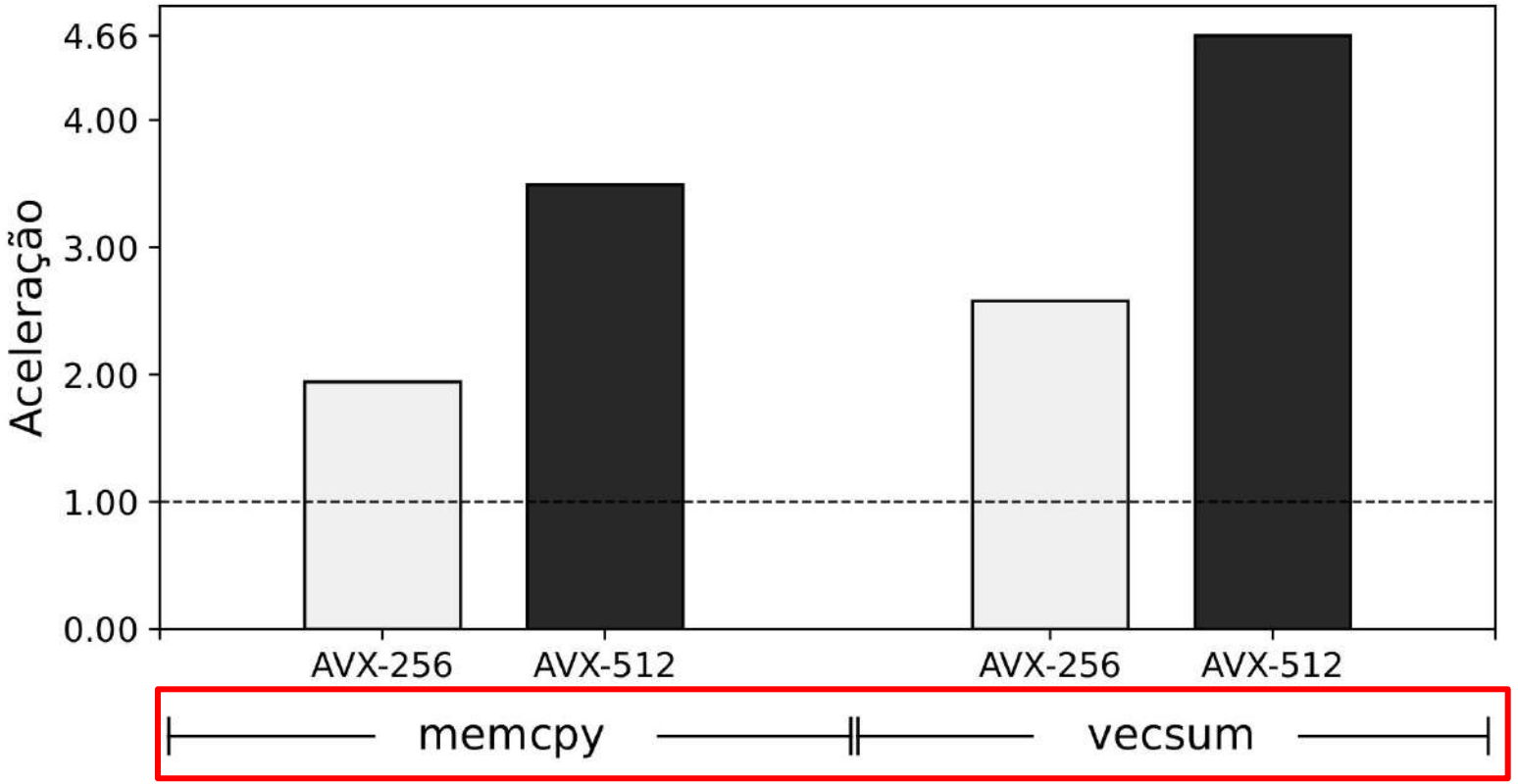
Parâmetros utilizados:

- Conversão **de** AVX-256 e AVX-512
- Conversão **para** VIMA-256 B
- **Agressividade** 8

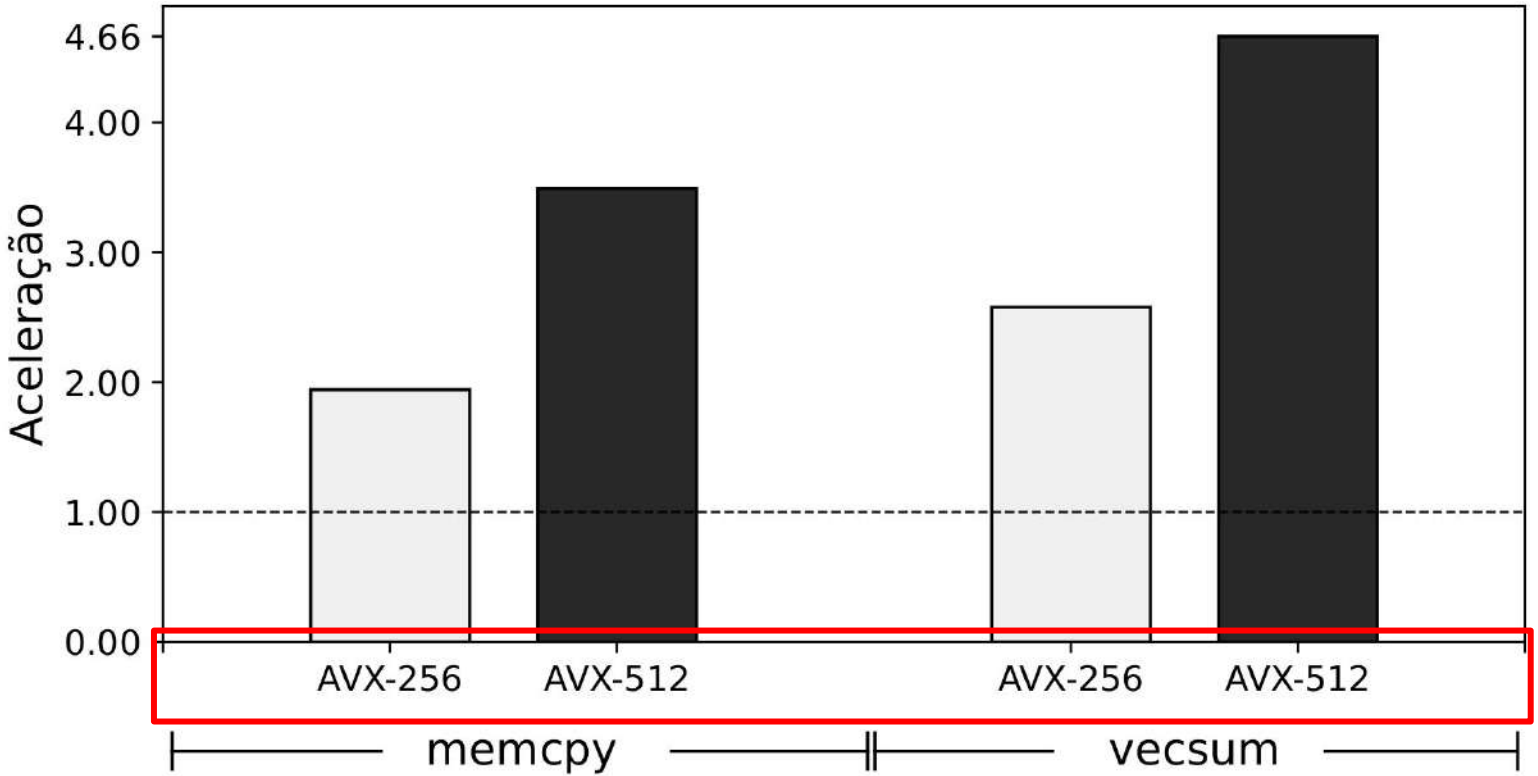
Versão de AVX Utilizada



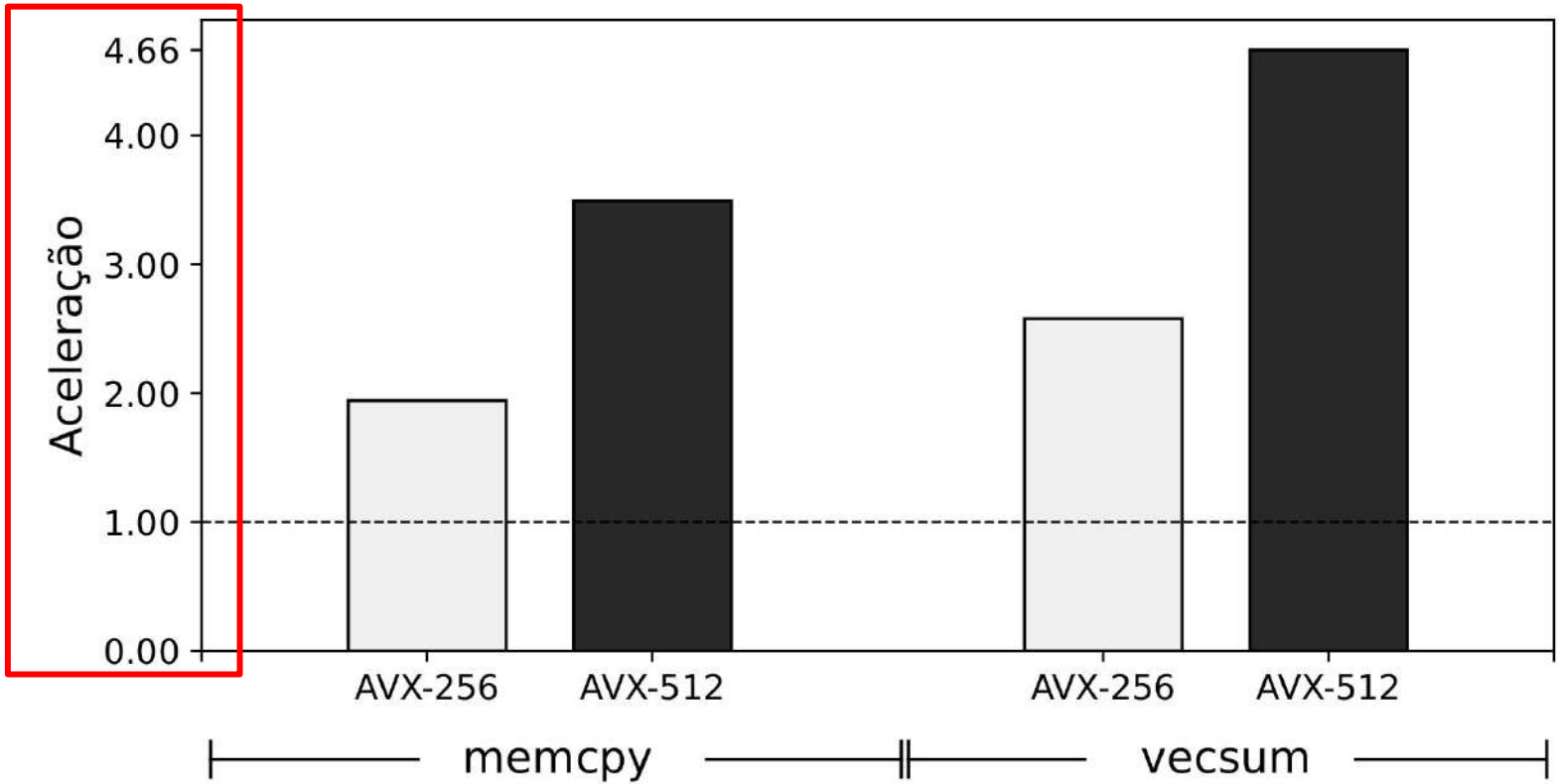
Versão de AVX Utilizada



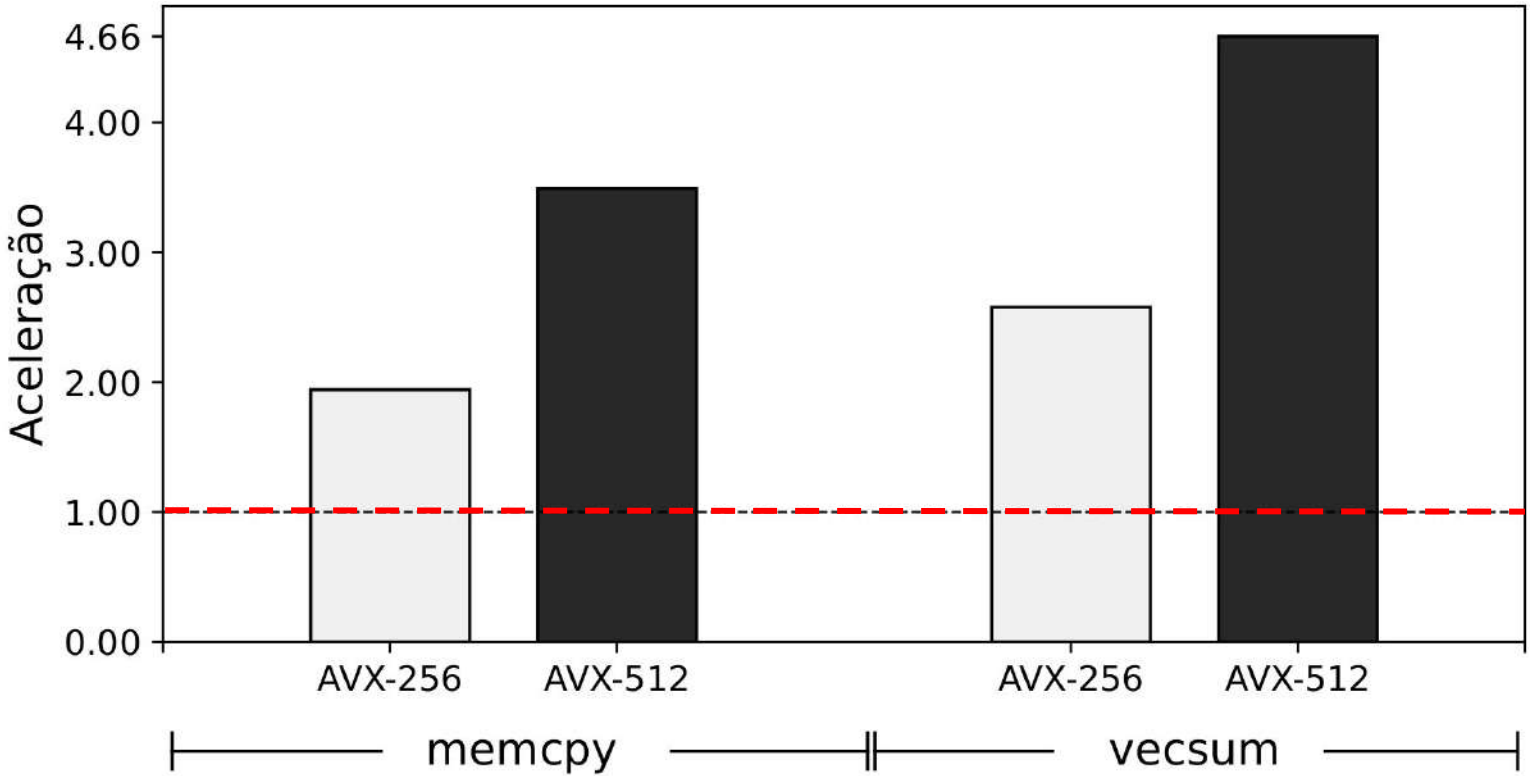
Versão de AVX Utilizada



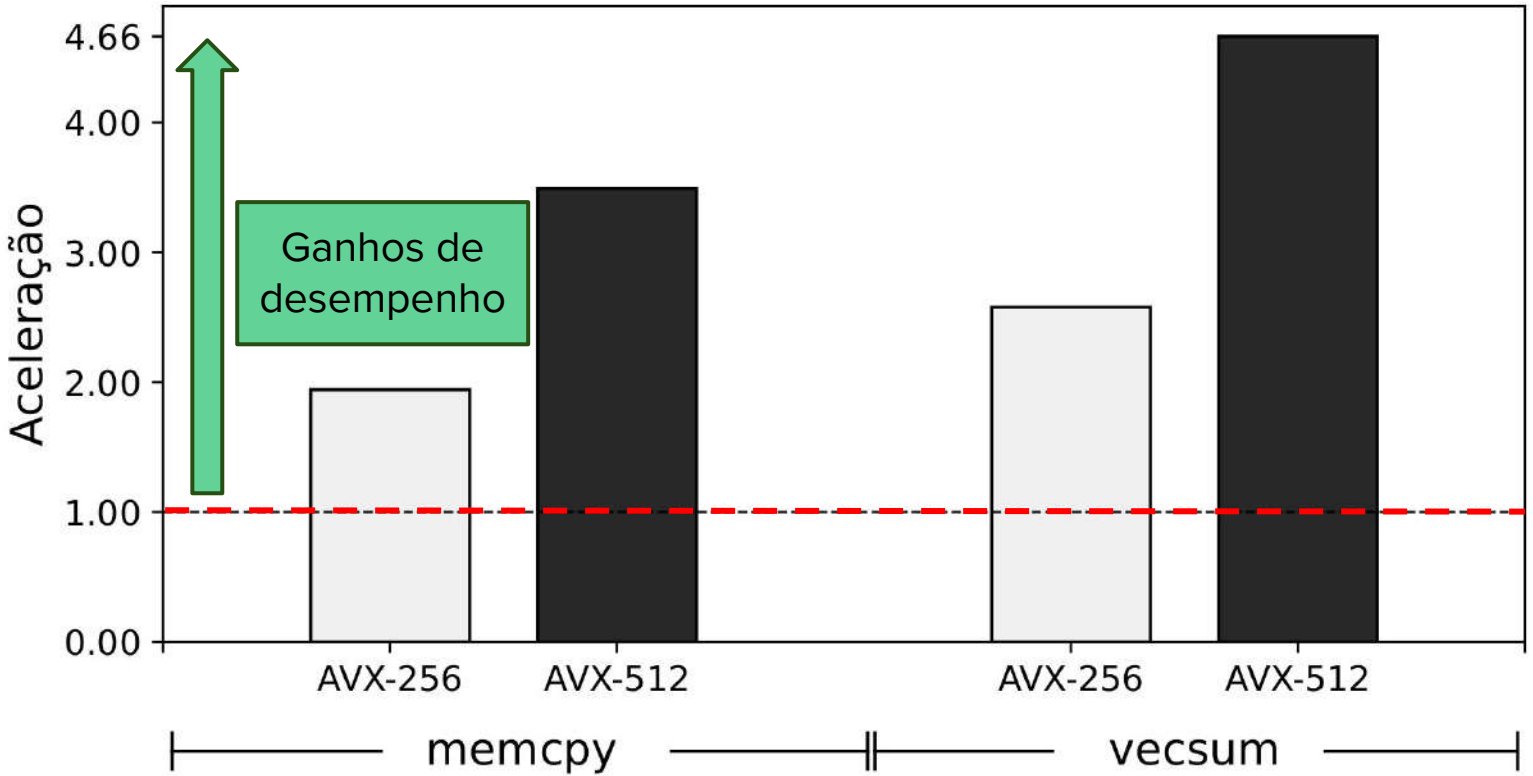
Versão de AVX Utilizada



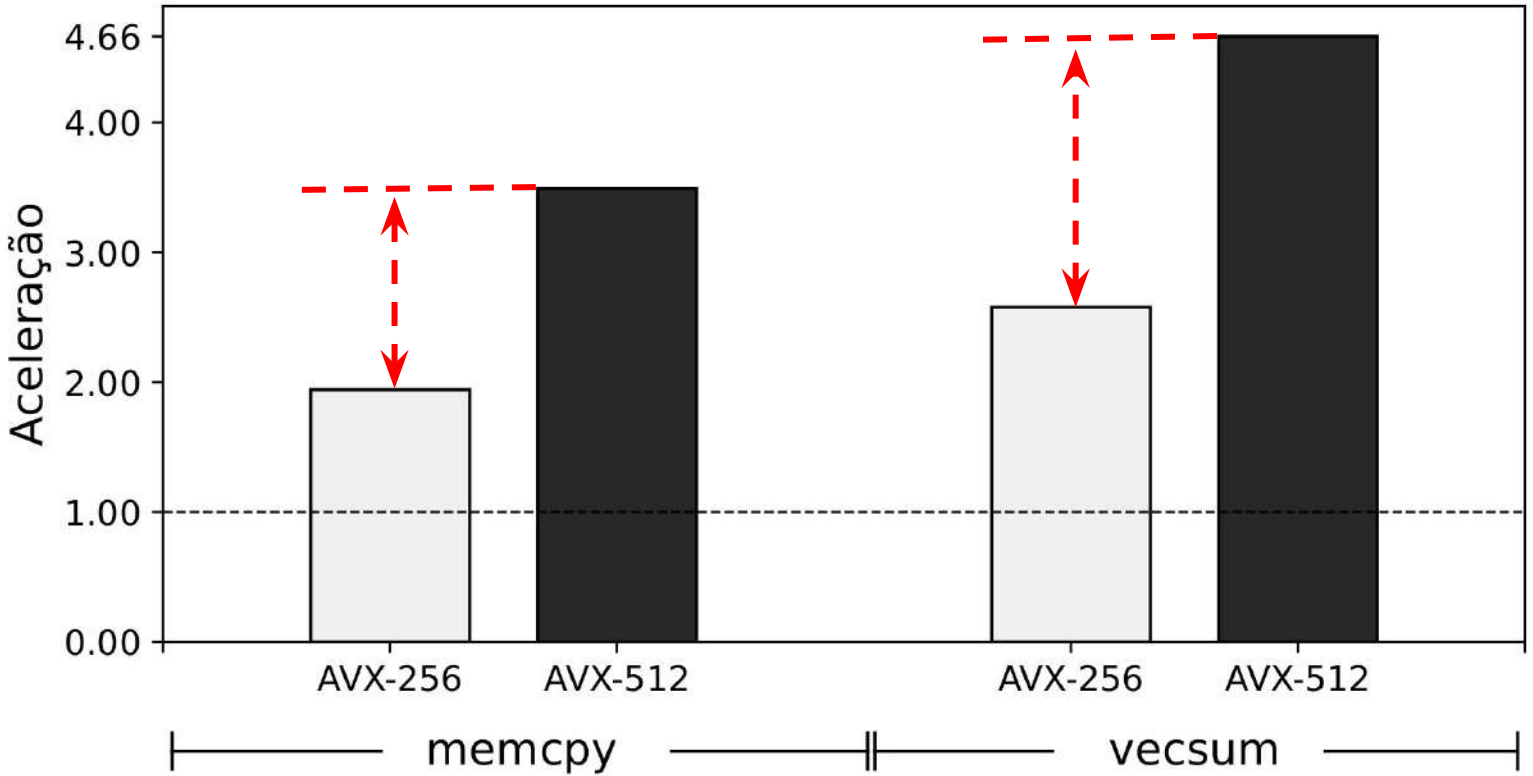
Versão de AVX Utilizada



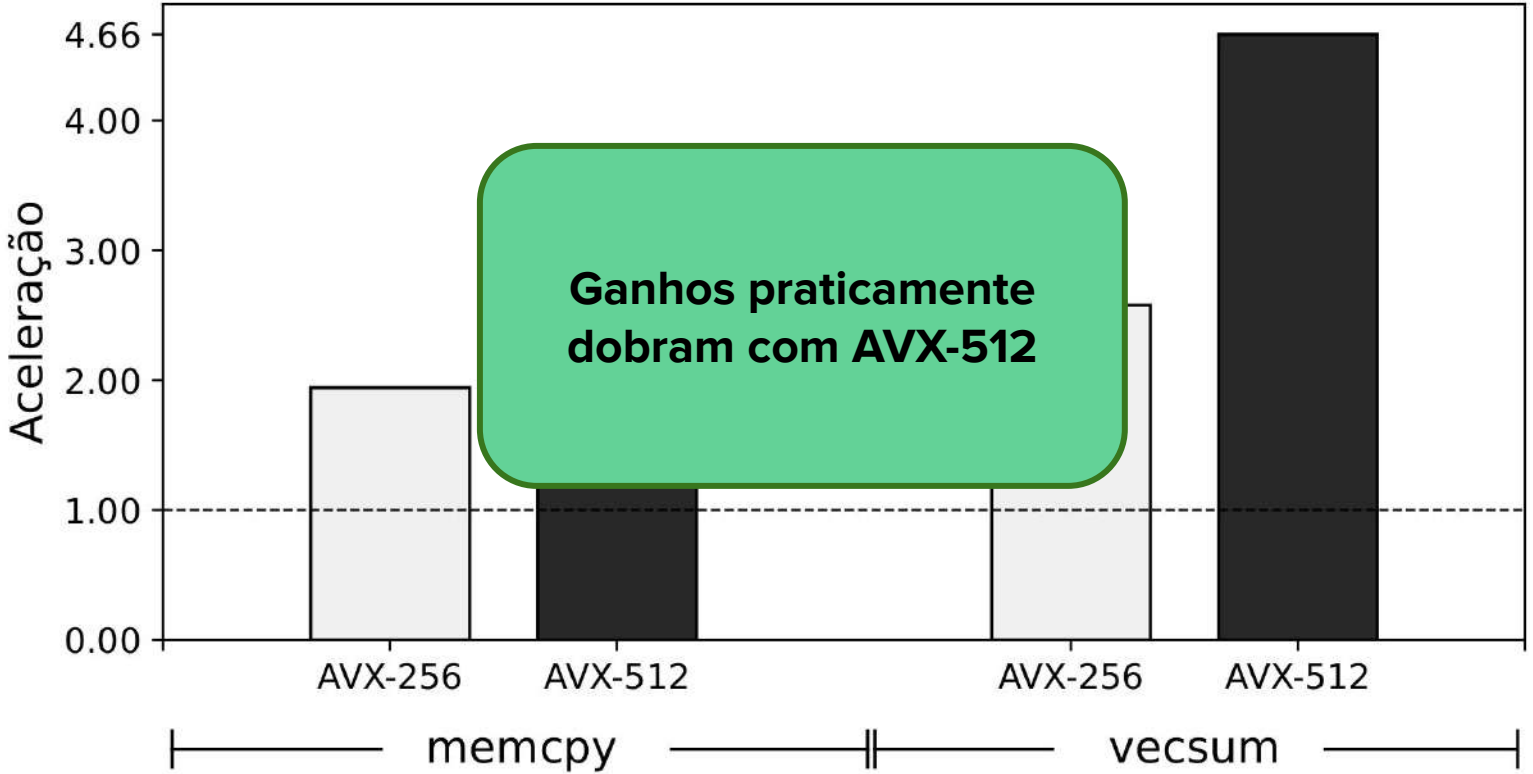
Versão de AVX Utilizada



Versão de AVX Utilizada



Versão de AVX Utilizada



Versão da Instrução VIMA

Objetivo: Avaliar o impacto da versão VIMA utilizada em conversões do SAPIVe

Instruções VIMA maiores

[+] Mais dados operados a cada conversão

[-] Mais instruções acumuladas no ROB por conversão

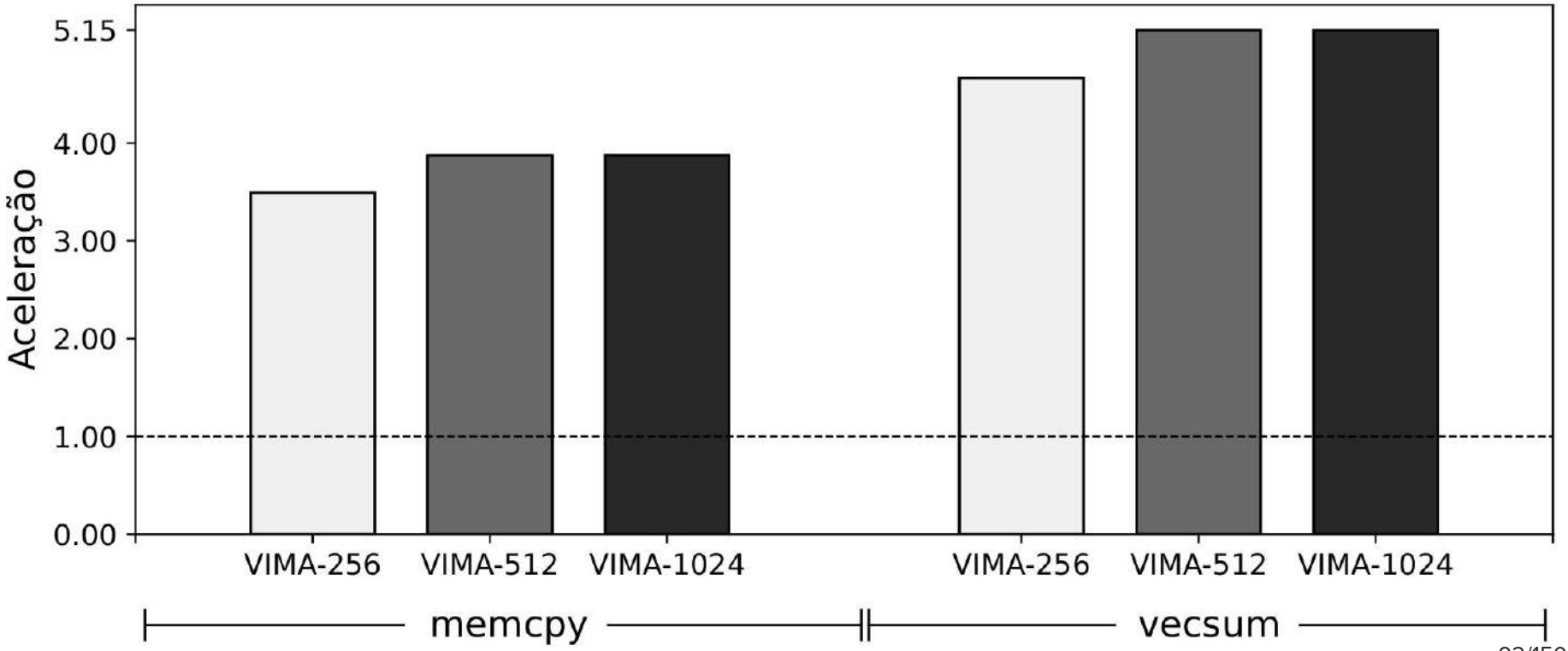
Versão da Instrução VIMA

Objetivo: Avaliar o impacto da versão VIMA utilizada em conversões do SAPIVe

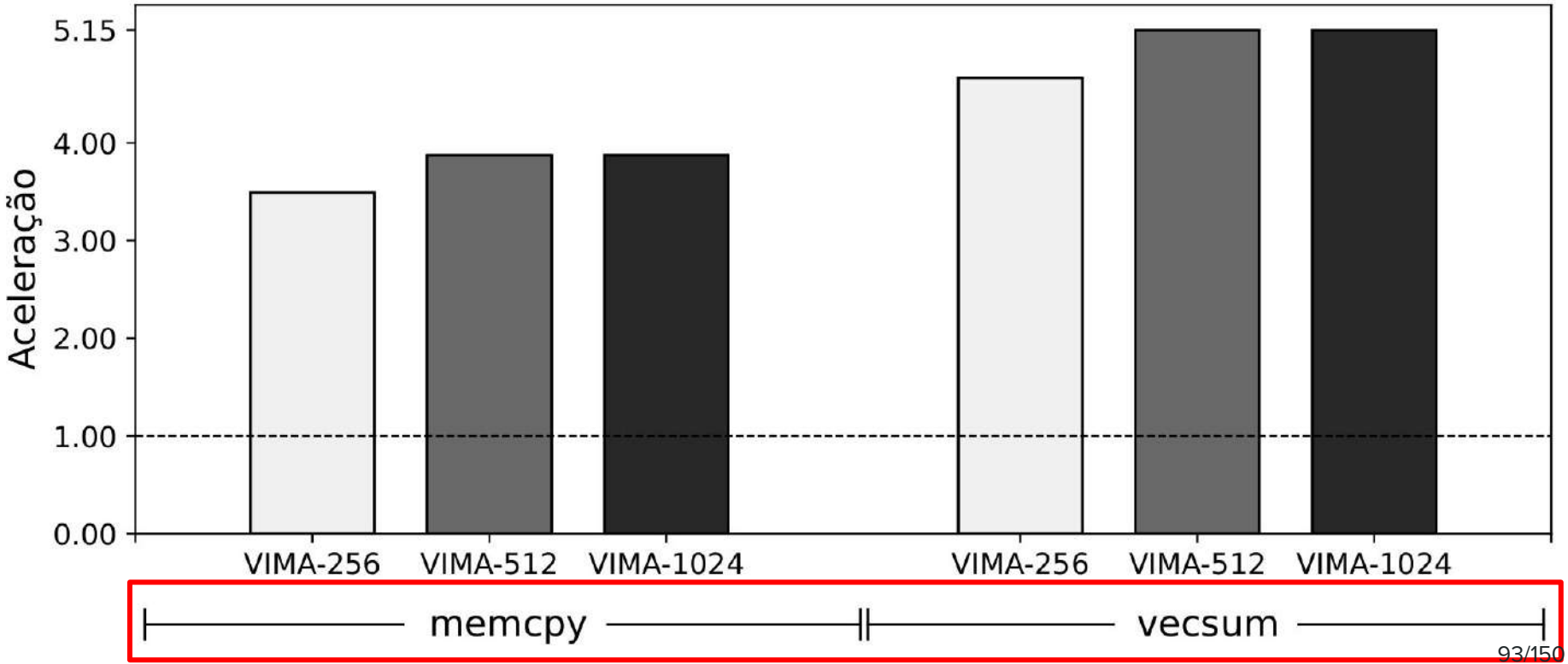
Parâmetros utilizados:

- Conversão **de** AVX-512
- Conversão **para** VIMA-256 B, VIMA-512 B e VIMA-1024 B
- **Agressividade** 8

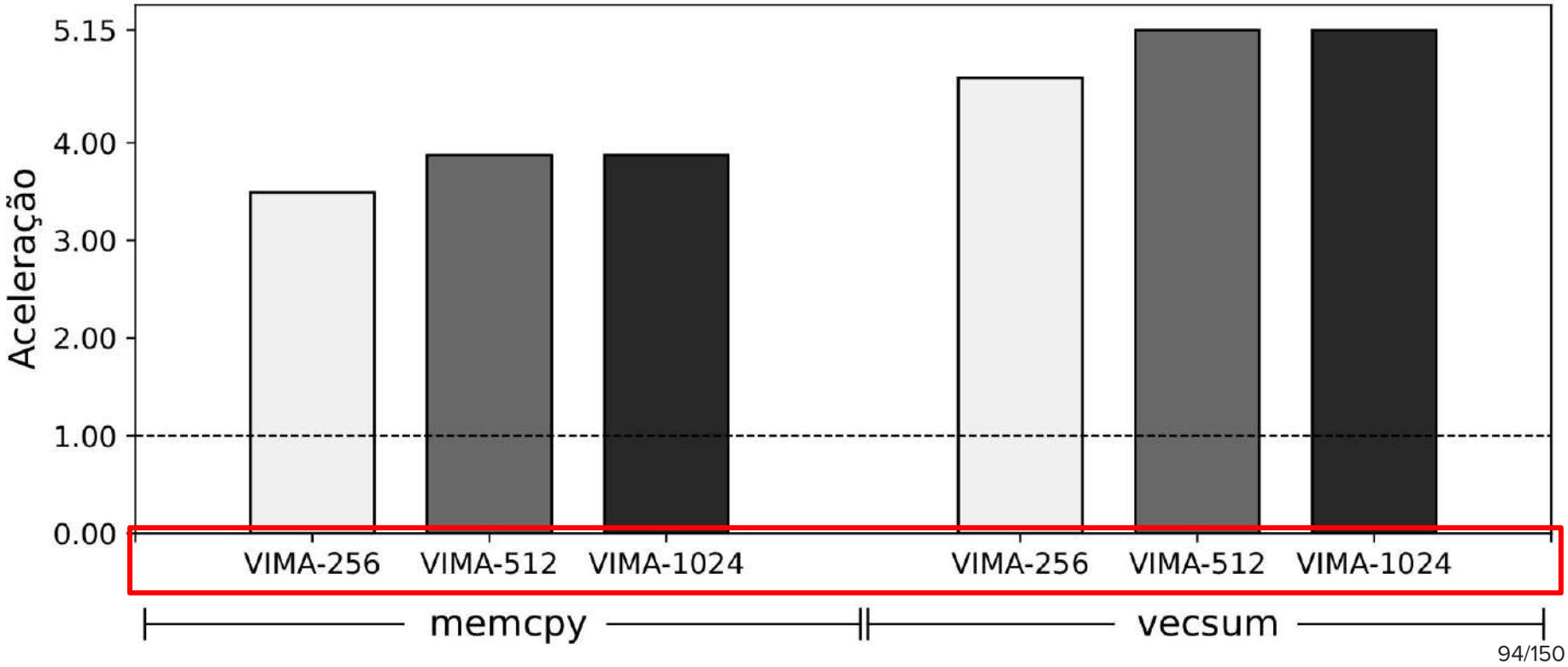
Versão da Instrução VIMA



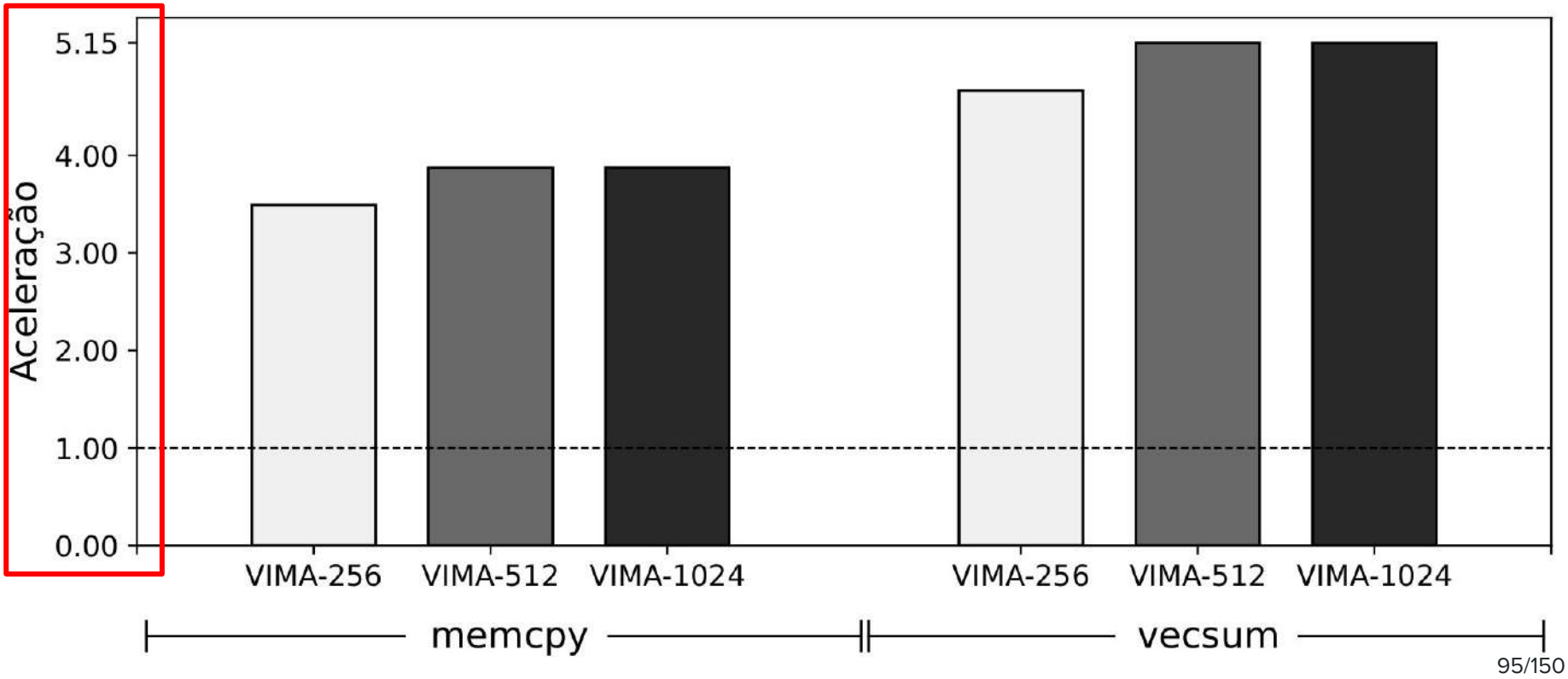
Versão da Instrução VIMA



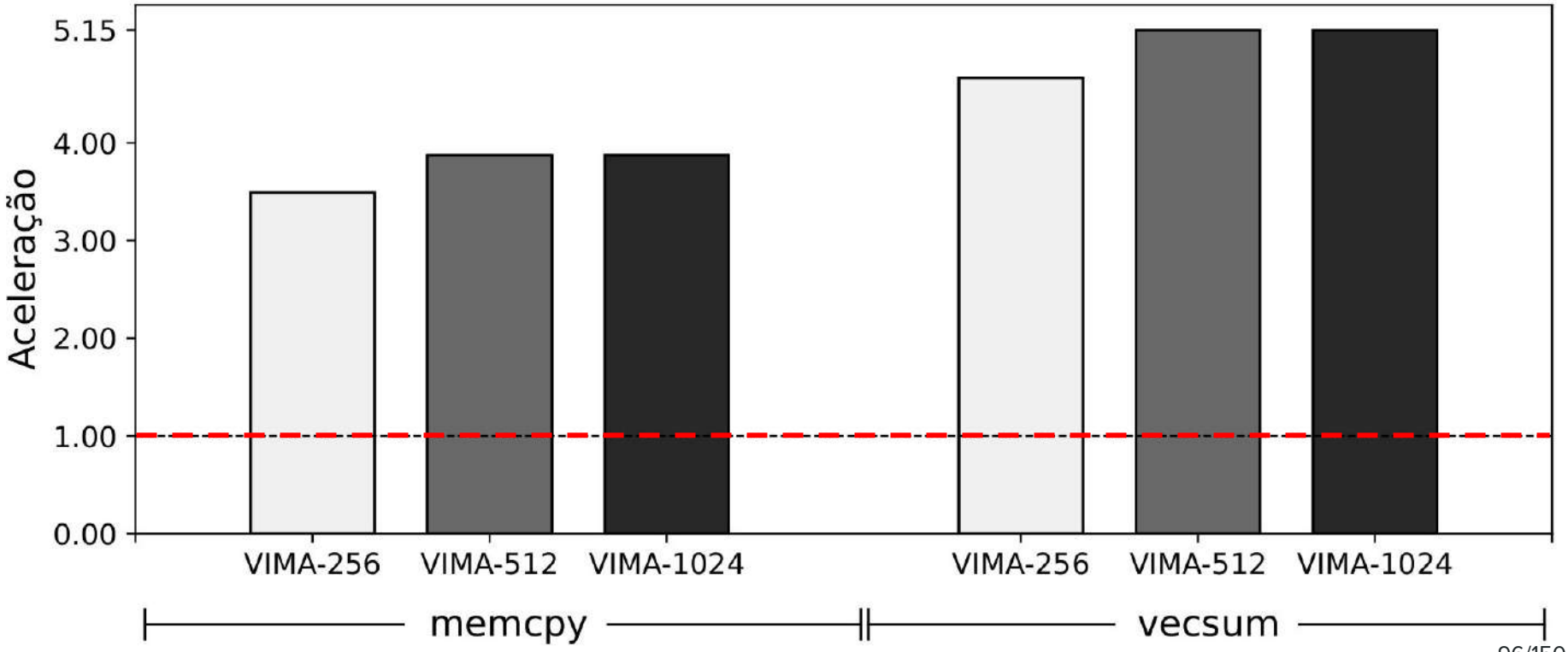
Versão da Instrução VIMA



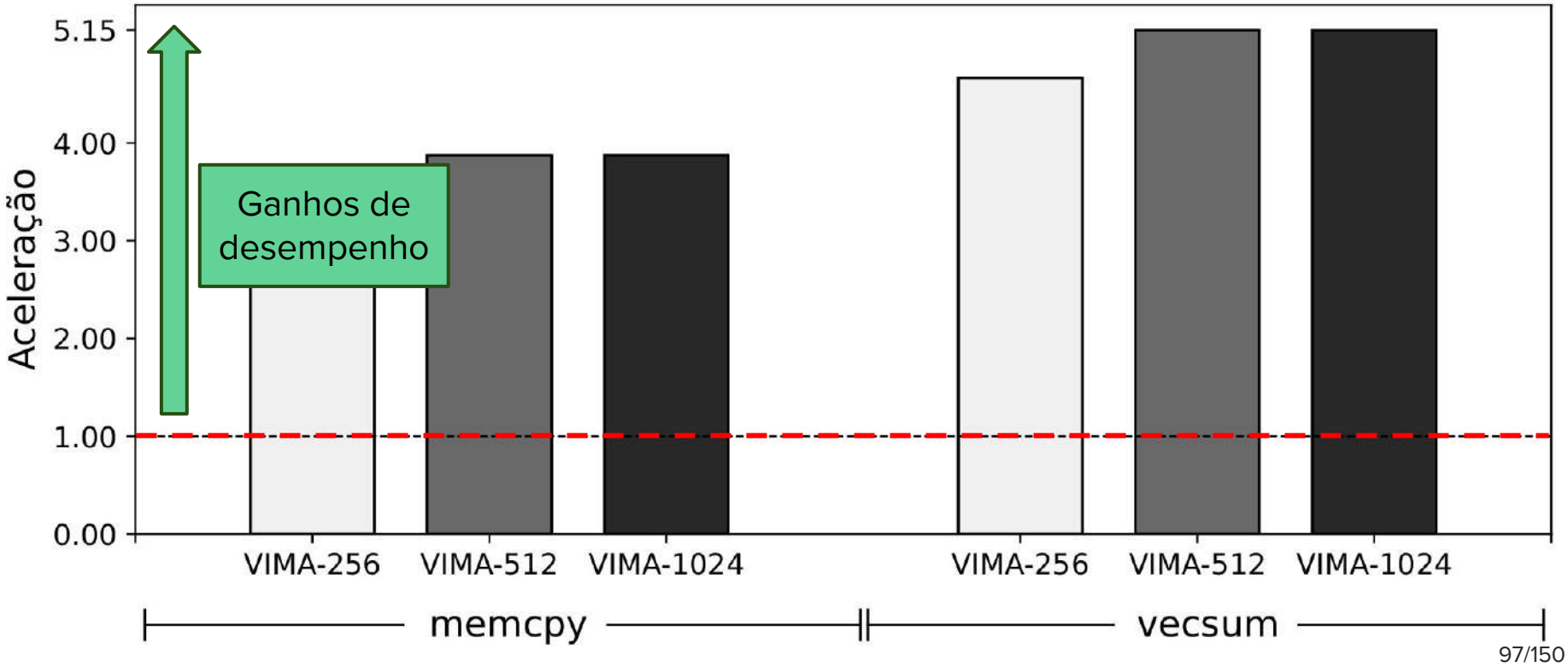
Versão da Instrução VIMA



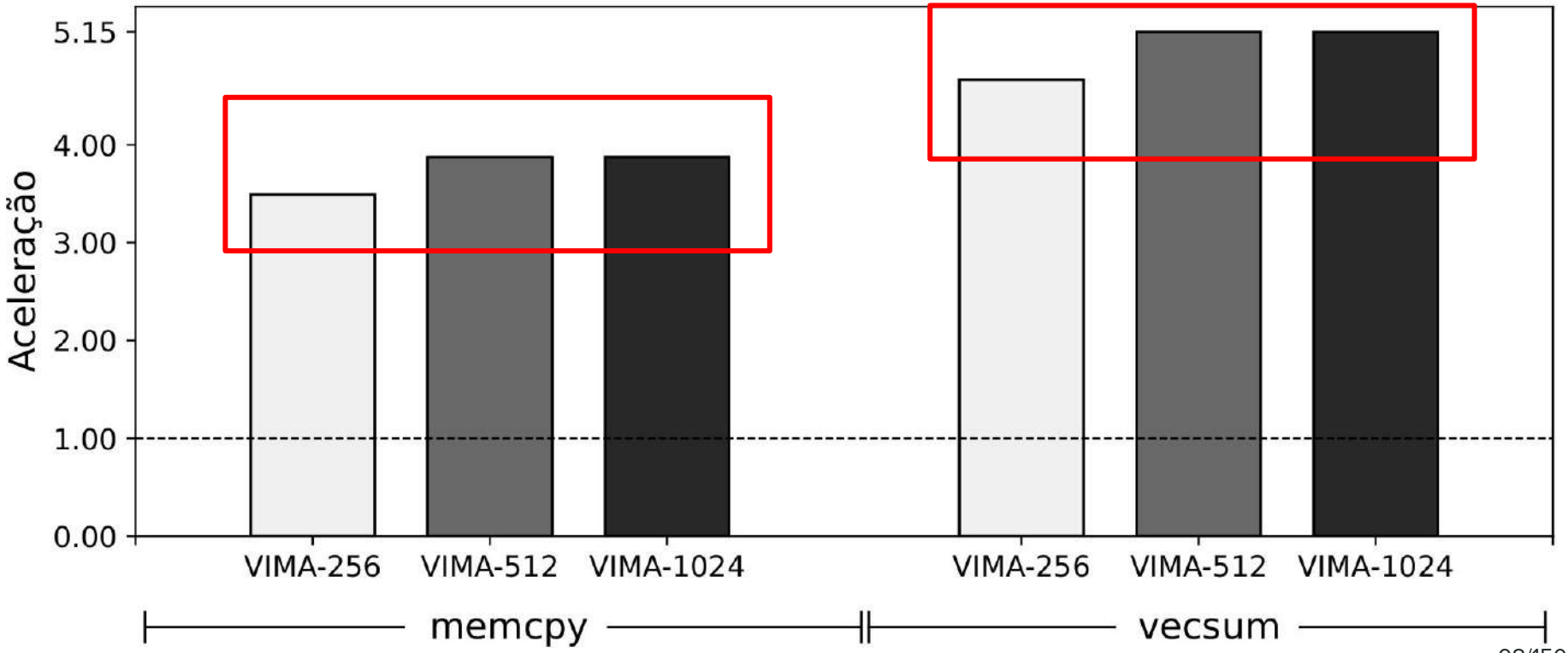
Versão da Instrução VIMA



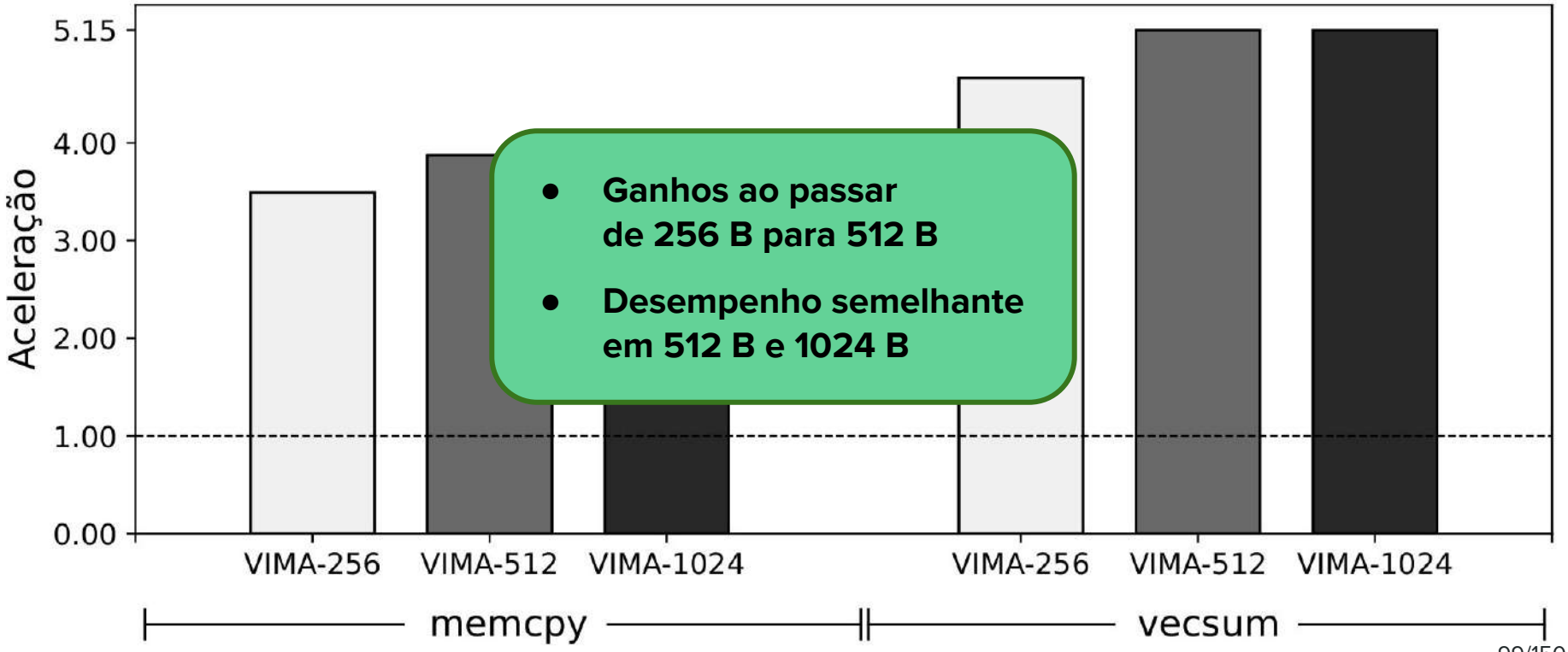
Versão da Instrução VIMA



Versão da Instrução VIMA



Versão da Instrução VIMA



- **Ganhos ao passar de 256 B para 512 B**
- **Desempenho semelhante em 512 B e 1024 B**

Agressividade e Tamanho do Laço

Objetivo: Avaliar o impacto da **agressividade** do SAPIVe em conversões sobre laços de diferentes tamanhos

Maior **agressividade**

- [+] Melhor uso do paralelismo interno da memória 3D
- [-] Maior ocupação do ROB
- [-] Redução no número de conversões verificadas paralelamente

Agressividade e Tamanho do Laço

Objetivo: Avaliar o impacto da agressividade do SAPIVe em conversões sobre **laços de diferentes tamanhos**

Maior **número de iterações**

- [+] Possíveis ganhos de desempenho intensificados
- [+] Redução no impacto da invalidação final
- [-] Possíveis perdas de desempenho intensificadas

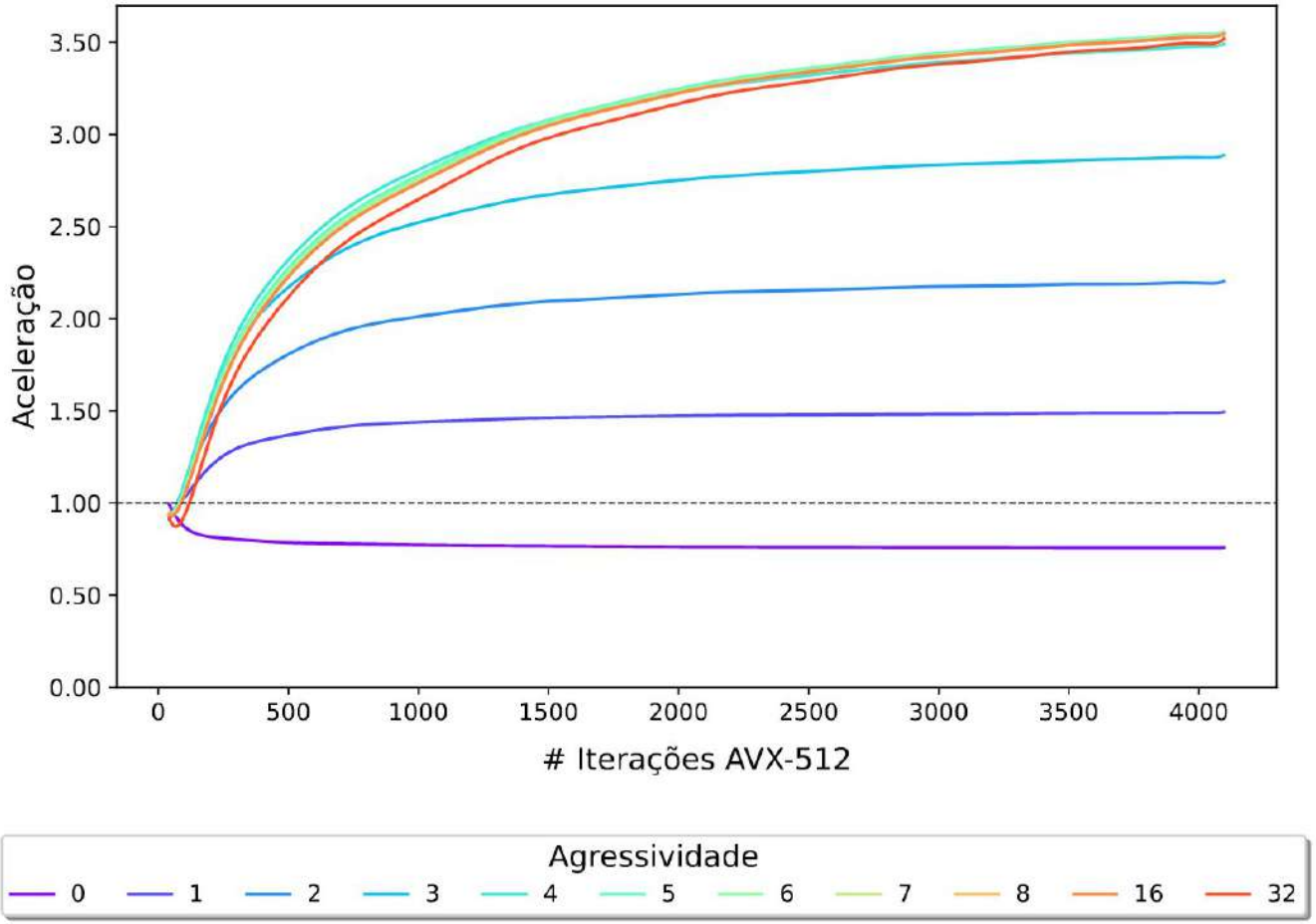
Agressividade e Tamanho do Laço

Objetivo: Avaliar o impacto da agressividade do SAPIVe em conversões sobre laços de diferentes tamanhos

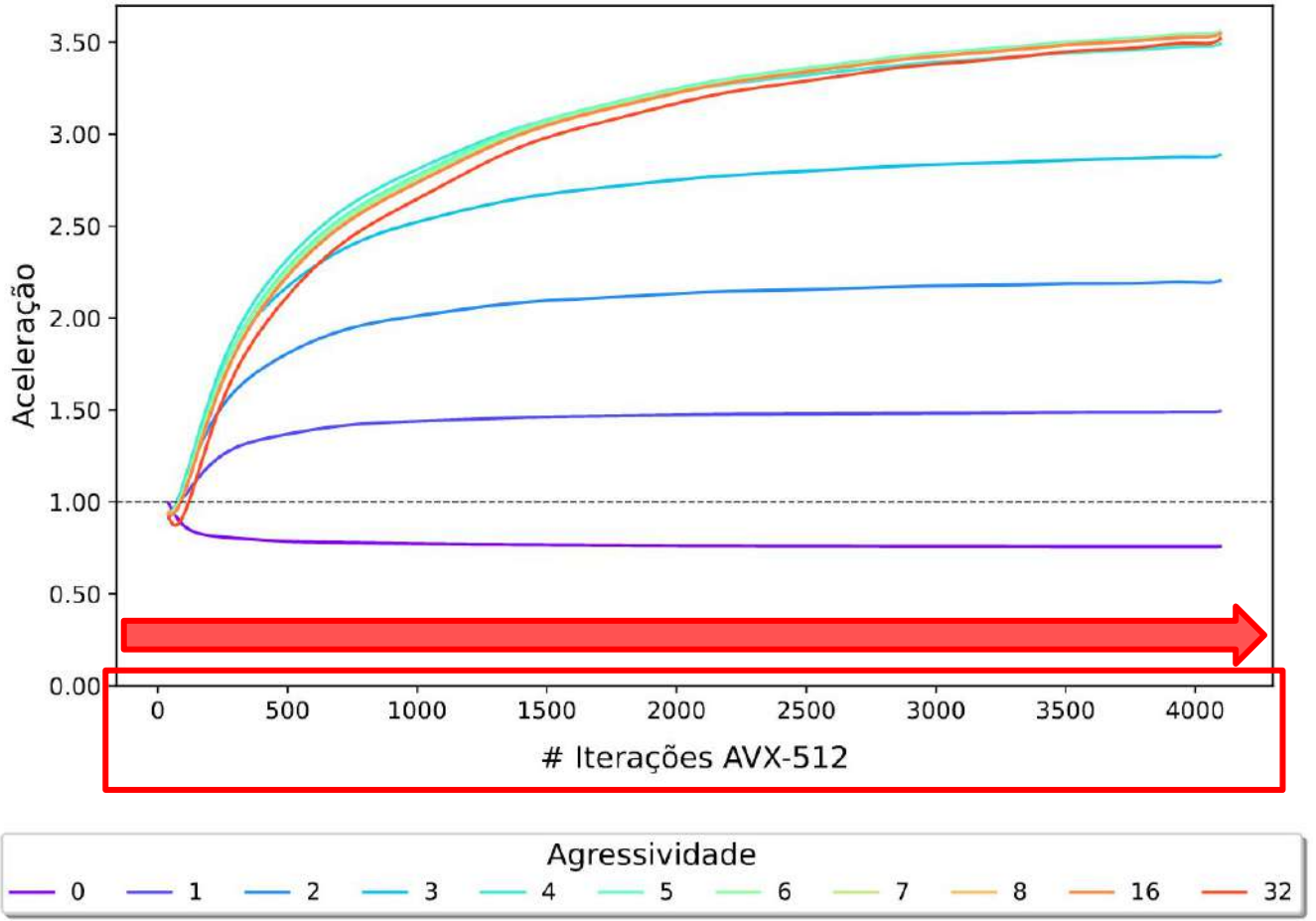
Parâmetros utilizados:

- Conversão **de** AVX-512
- Conversão **para** VIMA-512 B
- **Agressividades** entre 0 e 32
- Laços entre 1 e 4096 **iterações**

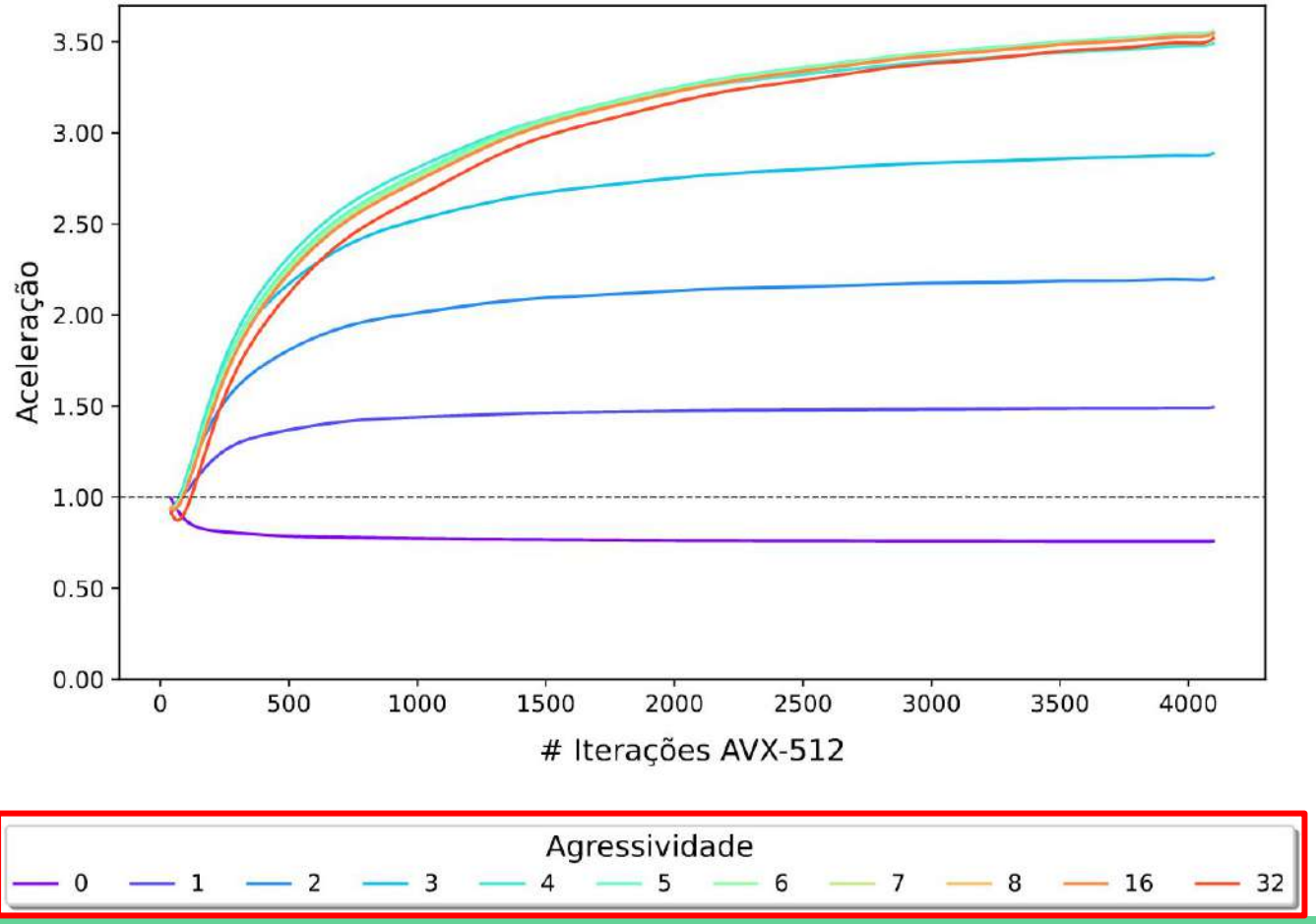
Agressividade e Tamanho do Laço - Cópia de memória



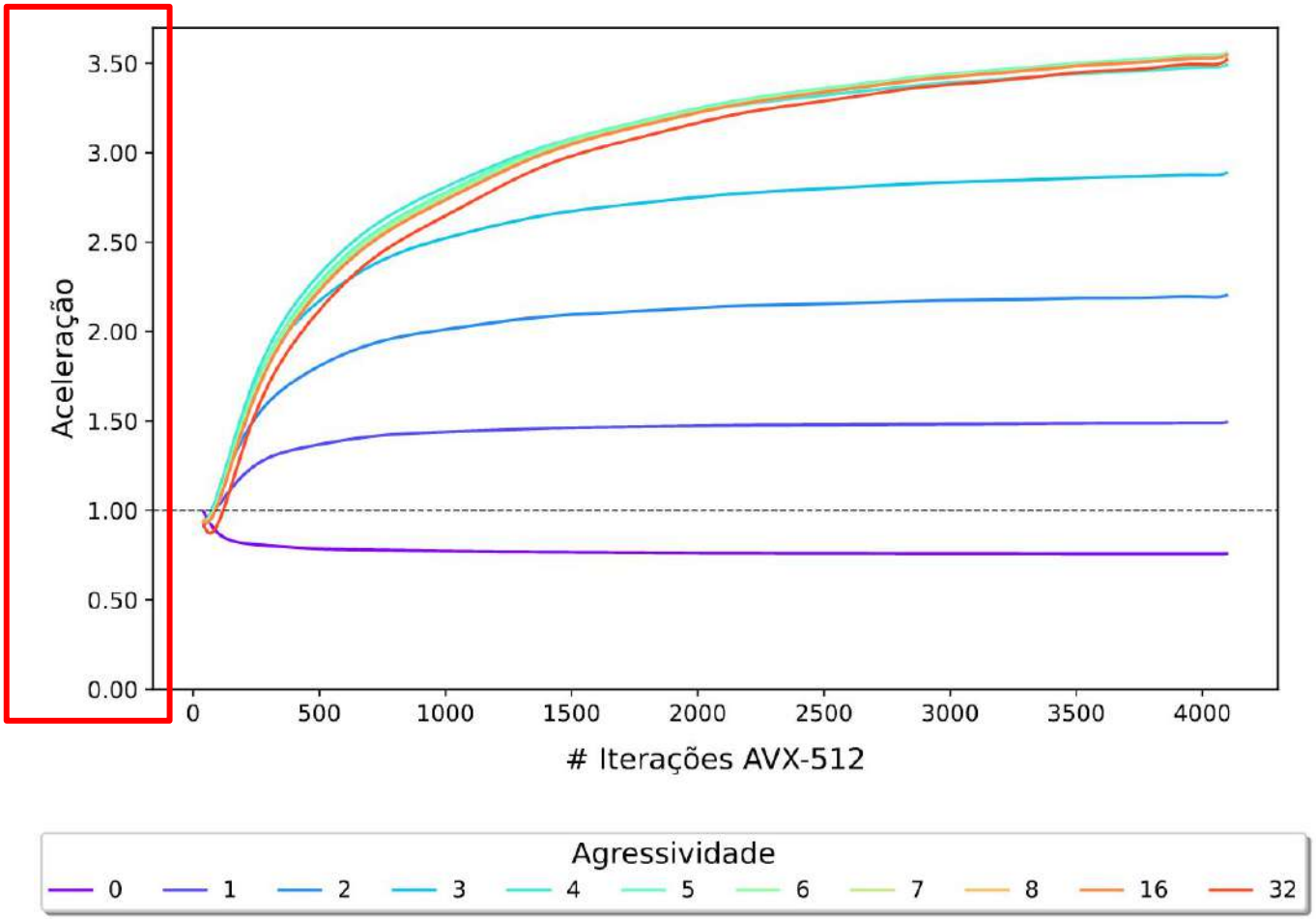
Agressividade e Tamanho do Laço - Cópia de memória



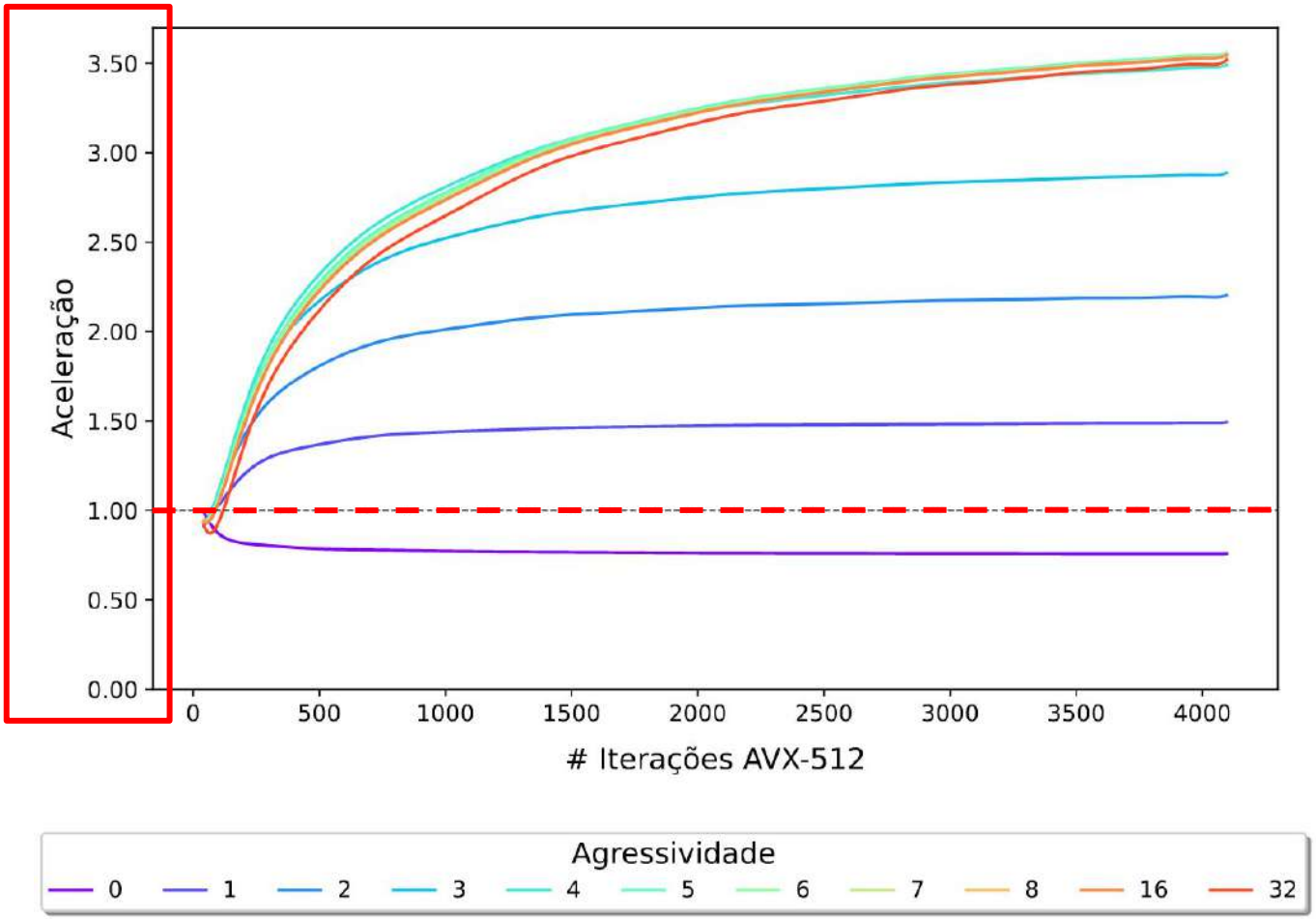
Agressividade e Tamanho do Laço - Cópia de memória



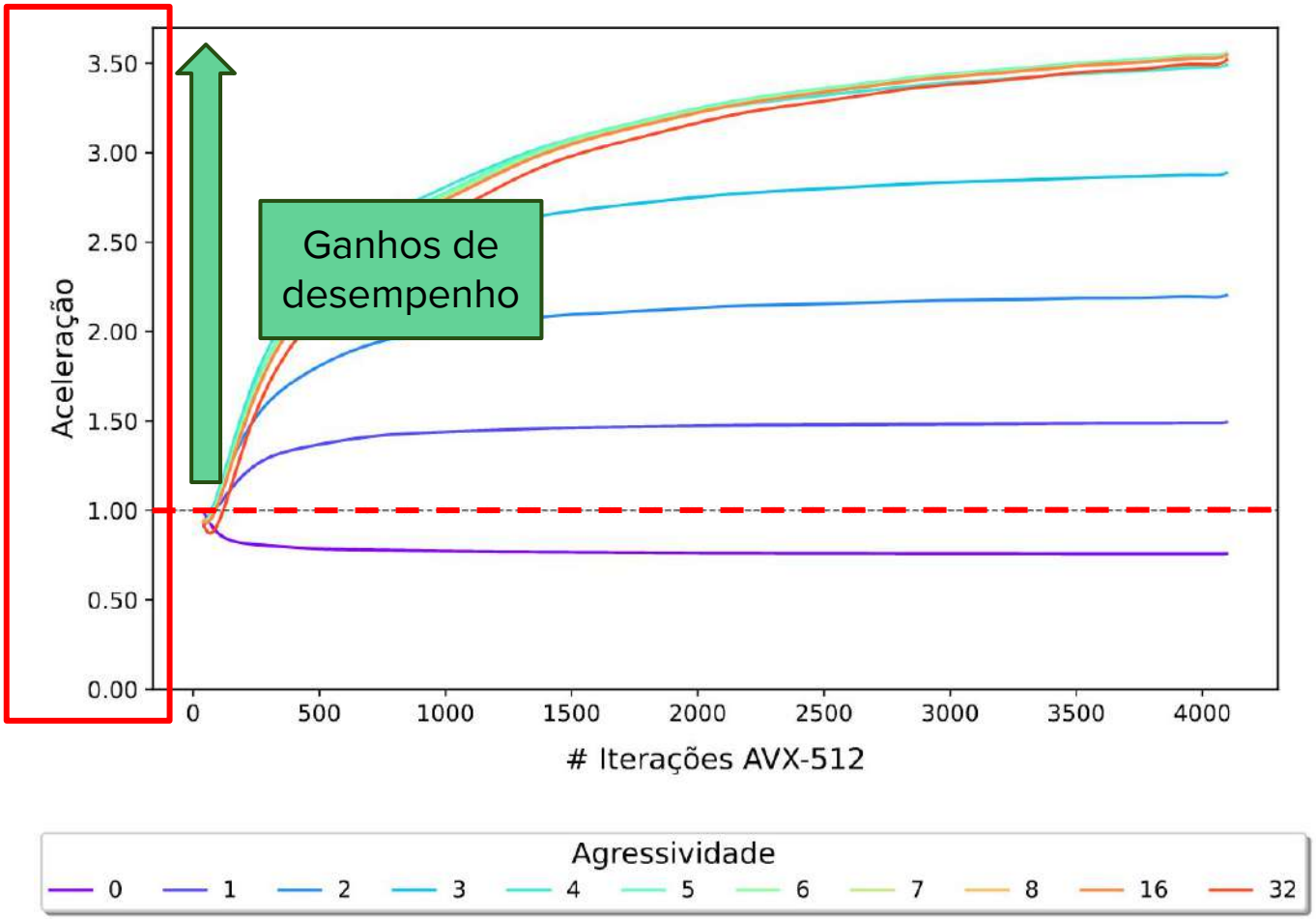
Agressividade e Tamanho do Laço - Cópia de memória



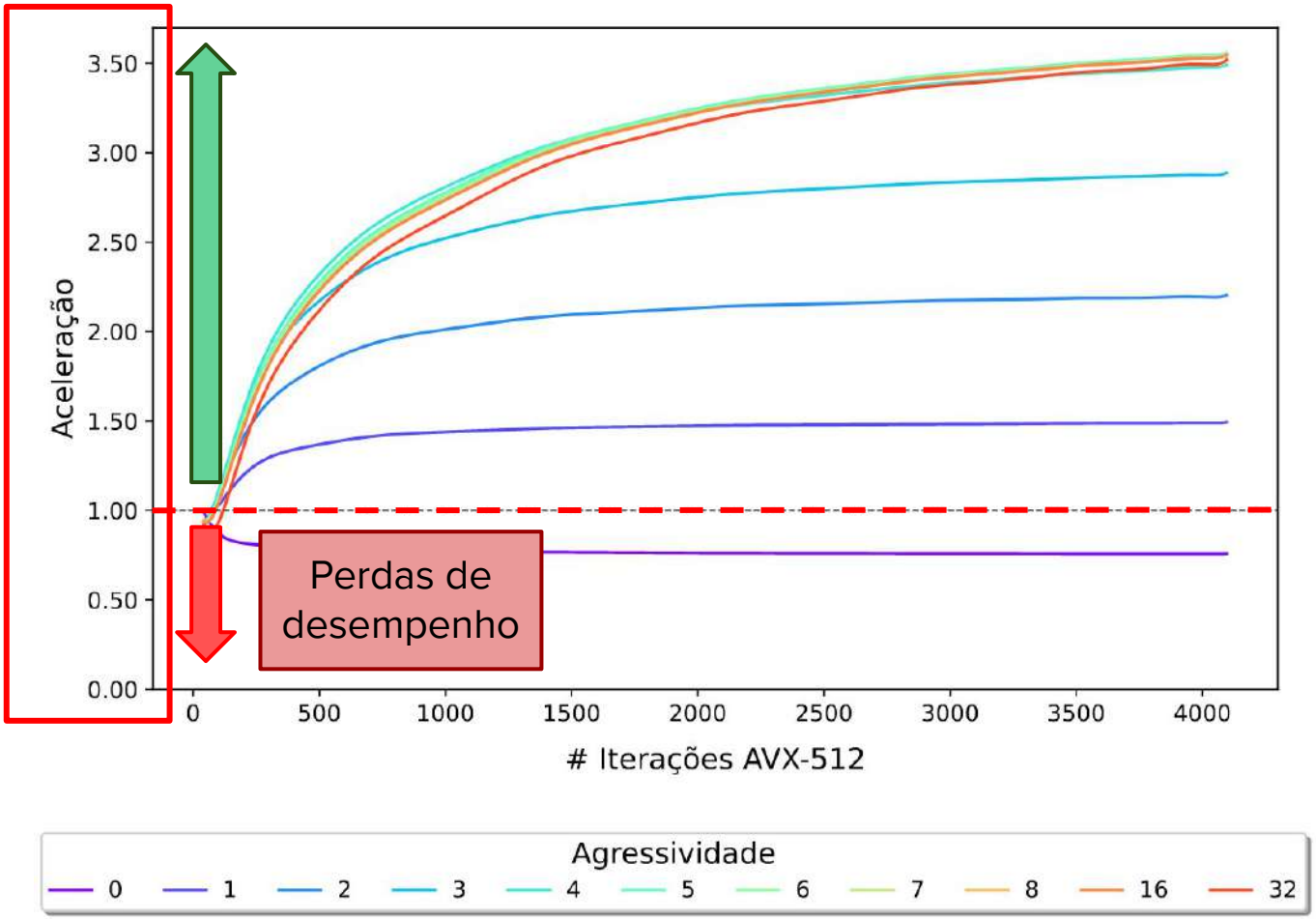
Agressividade e Tamanho do Laço - Cópia de memória



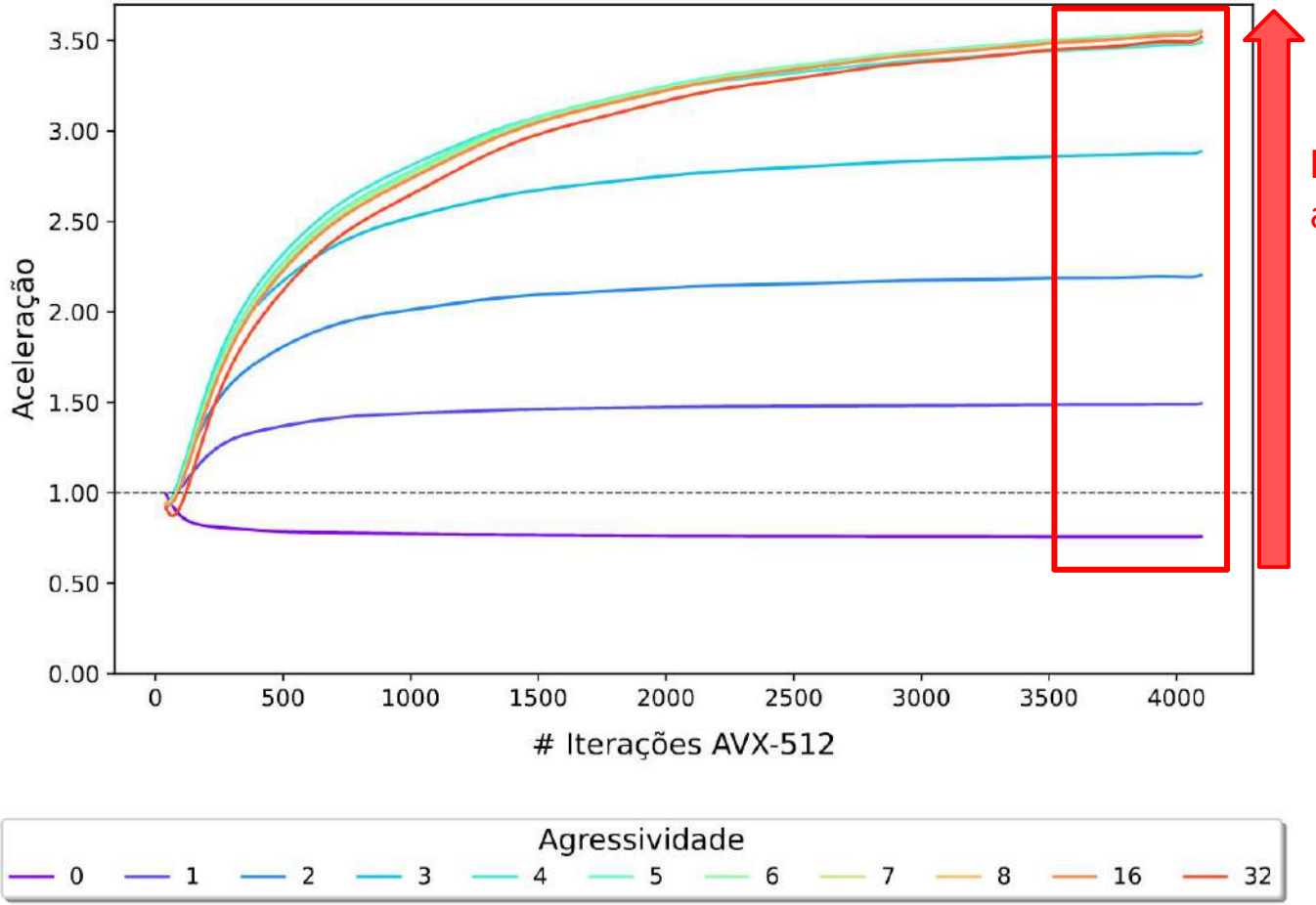
Agressividade e Tamanho do Laço - Cópia de memória



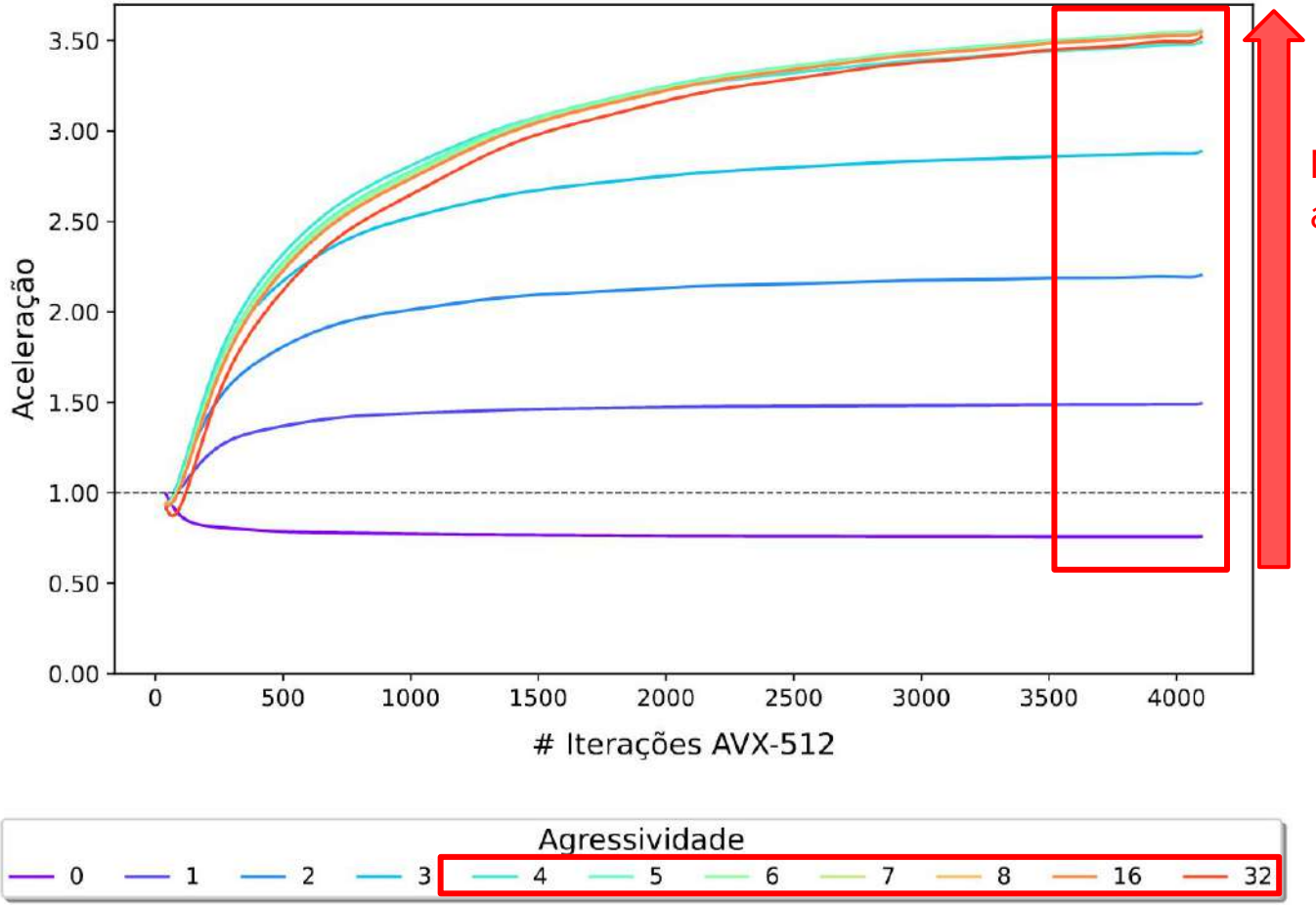
Agressividade e Tamanho do Laço - Cópia de memória



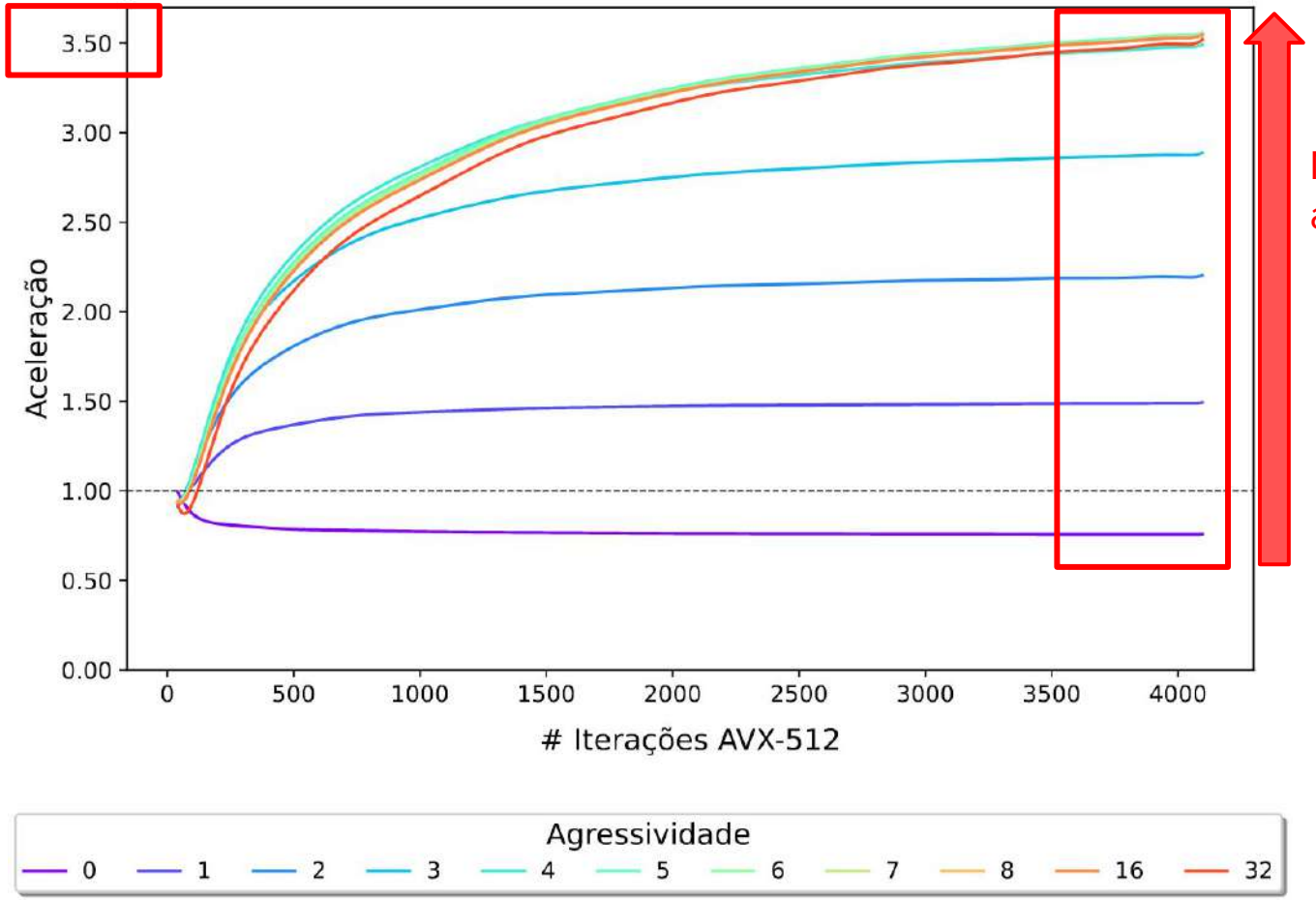
Agressividade e Tamanho do Laço - Cópia de memória



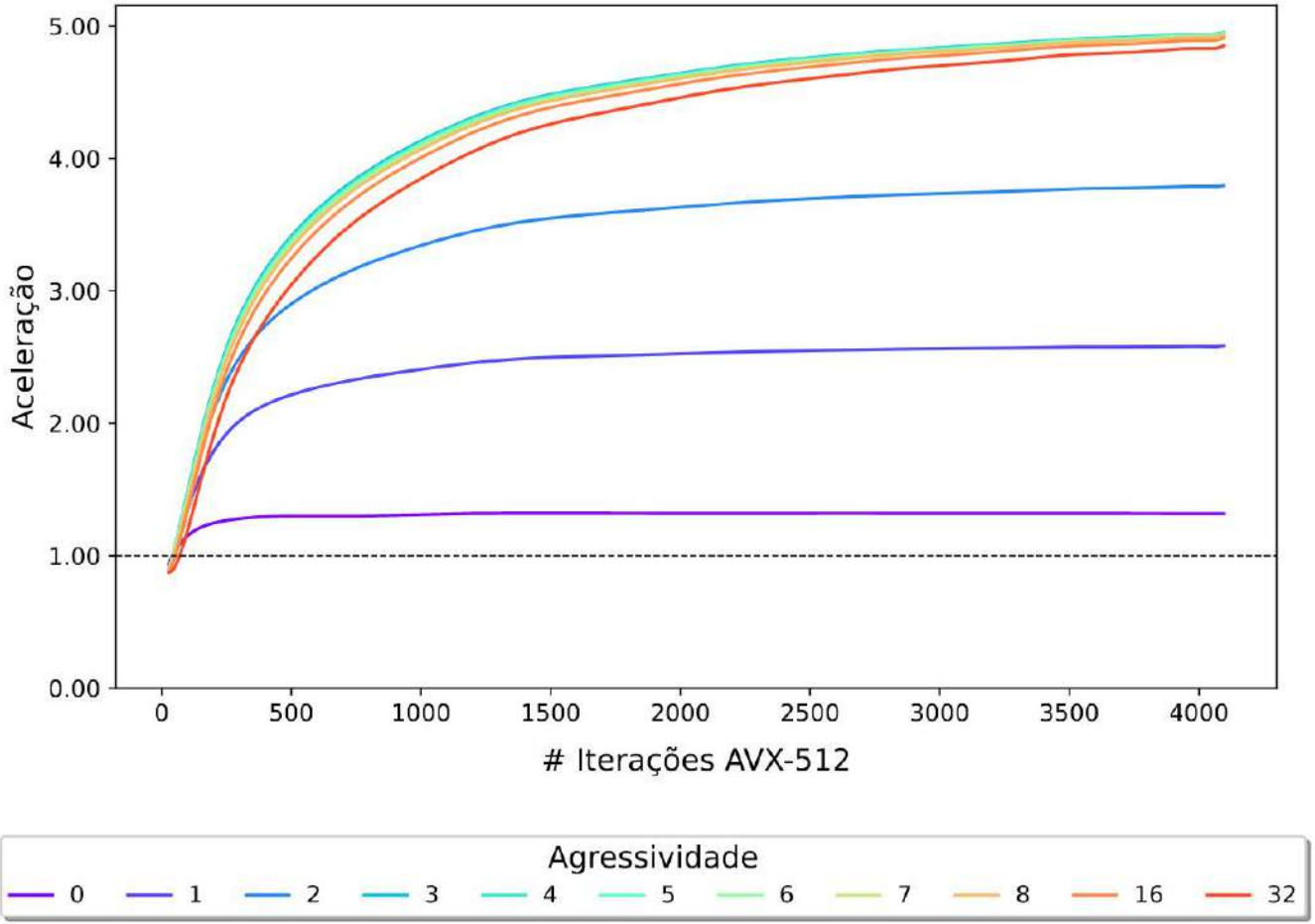
Agressividade e Tamanho do Laço - Cópia de memória



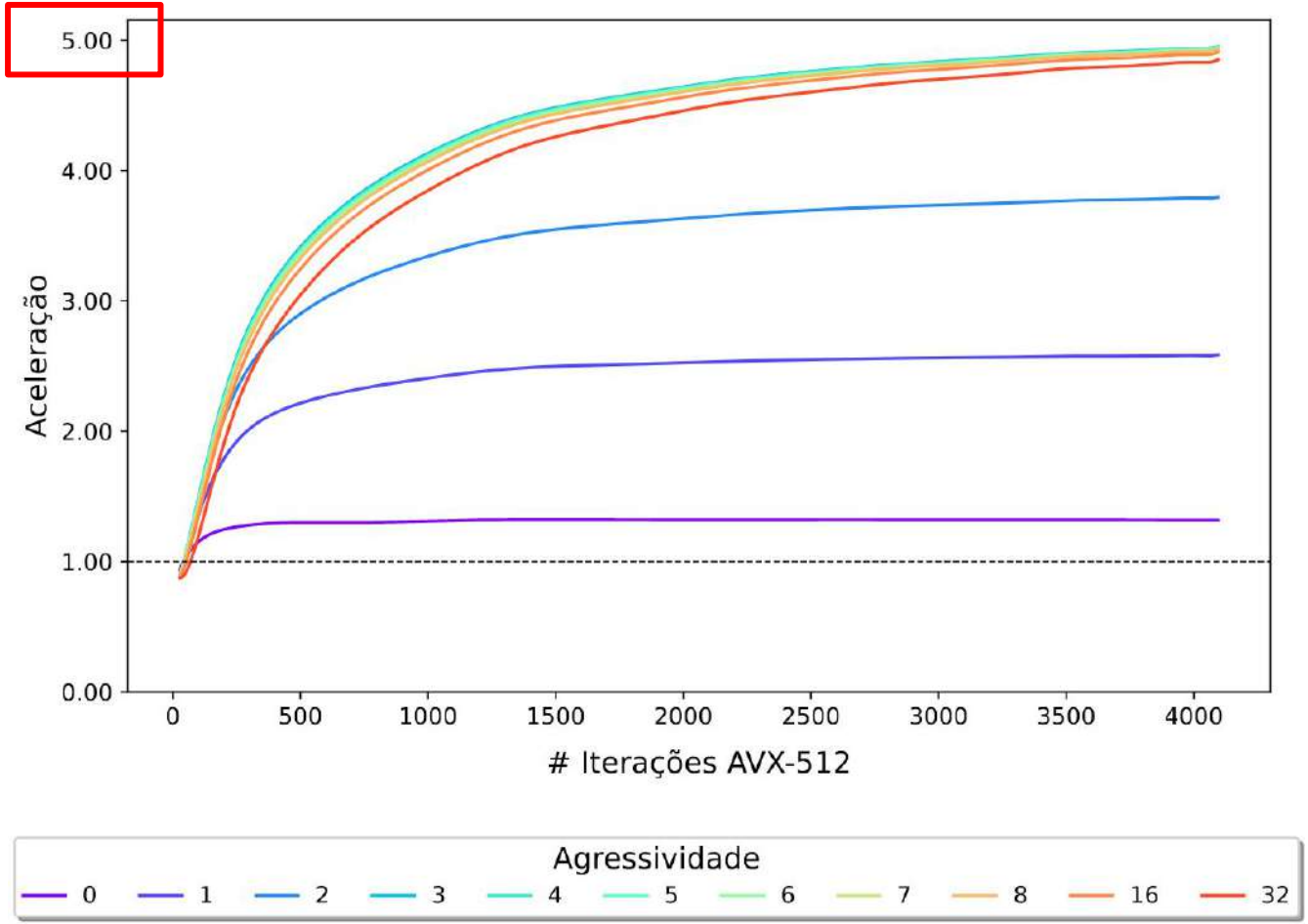
Agressividade e Tamanho do Laço - Cópia de memória



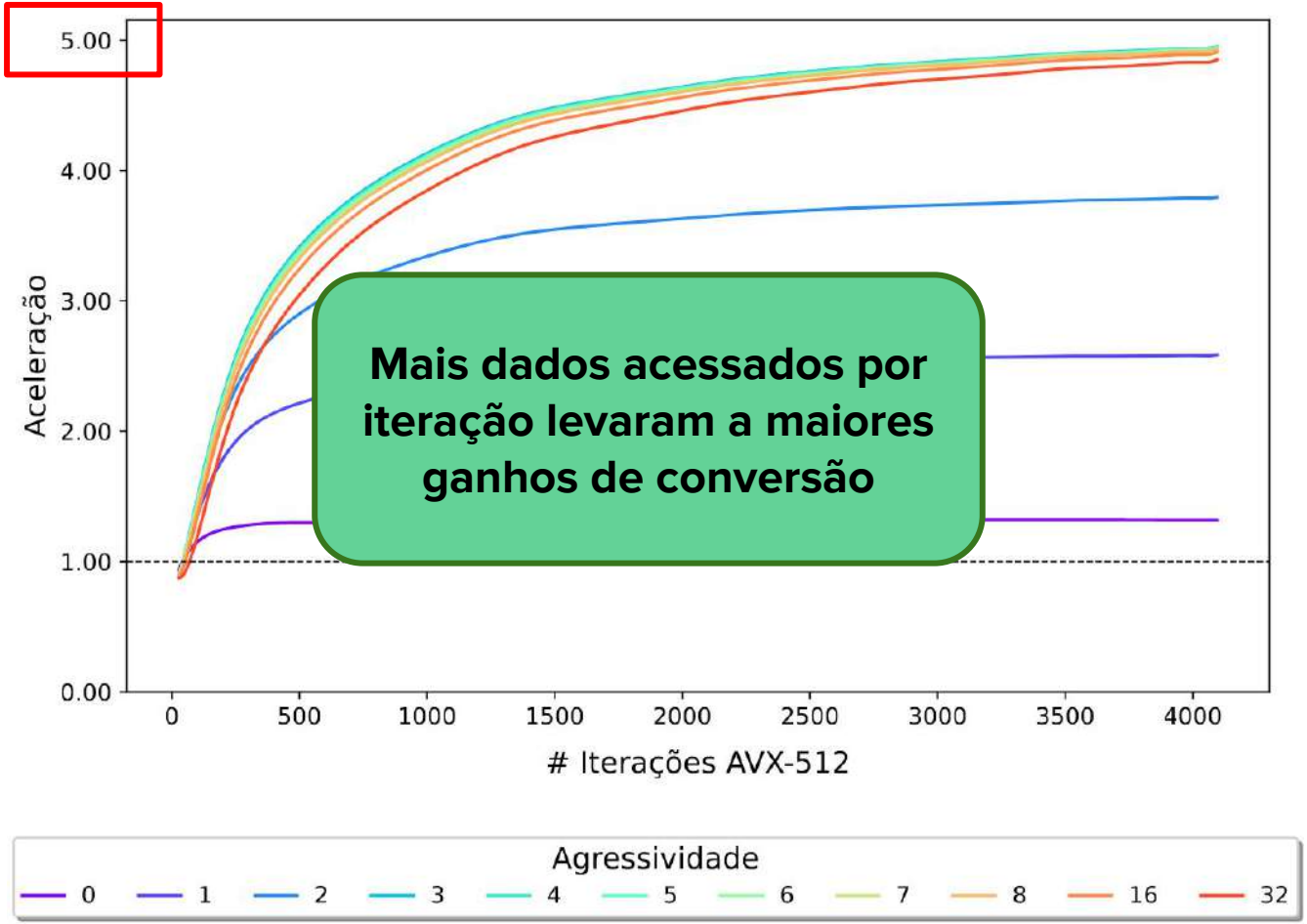
Agressividade e Tamanho do Laço - Soma entre vetores



Agressividade e Tamanho do Laço - Soma entre vetores



Agressividade e Tamanho do Laço - Soma entre vetores



Comparação com Implementação Manual

Estamos perto dos ganhos máximos das conversões?

- Comparação com **rotinas *intrinsic***, inseridas manualmente

Parâmetros utilizados:

- Conversão **de** AVX-512
- Conversão **para** VIMA-512 B e VIMA-1024 B
- **Agressividades** entre 0 e 32
- **Laços** operando sobre vetores de 8 MB

Comparação com Implementação Manual

Na maior parte dos casos, SAPIVe obteve menos de 50 % da aceleração obtida pelas intrinsics

SAPIVe não é capaz de acessar o potencial completo da VIMA:

- As **verificações** de conversão são **limitadas pelo tamanho do ROB**, impedindo um maior número de conversões paralelas
    - Verificações da CPU são um gargalo para as conversões

Comparação com Implementação Manual

Na maior parte dos casos, SAPIVe obteve menos de 50 % da aceleração obtida pelas intrinsics

Considerando um sistema futuro com a largura dos *buffers* 4x maior?

Comparação com Implementação Manual

Na maior parte dos casos, SAPIVe obteve menos de 50 % da aceleração obtida pelas intrinsics

Considerando um sistema futuro com a largura dos *buffers* 4x maior?

- Mais verificações em paralelo
- Aumento dos ganhos
- Em alguns casos chegando aos mesmos das intrinsics (10x)

Aplicações Reais

- SPEC-CPU 2017

Método - Aplicações Reais

- Traços de execução do SPEC-2017
  - Aplicações compiladas com flags para **habilitar** SSE, AVX, AVX2 e AVX-512
  - **Traços de execução,** com 200 milhões de instruções, obtidos usando o PinPoints [1]

[1] Patil, H., Cohn, R. et al. (2004). Pinpointing representative portions of large intel ® itanium ® programs with dynamic instrumentation. Em Int. Symp. on Microarchitecture.

Método - Aplicações Reais

- Traços de execução do SPEC-2017
  - Aplicações compila VX, AVX2 e AVX-512
  - **Traços de execuçã** btidos usando o PinPoints [1]

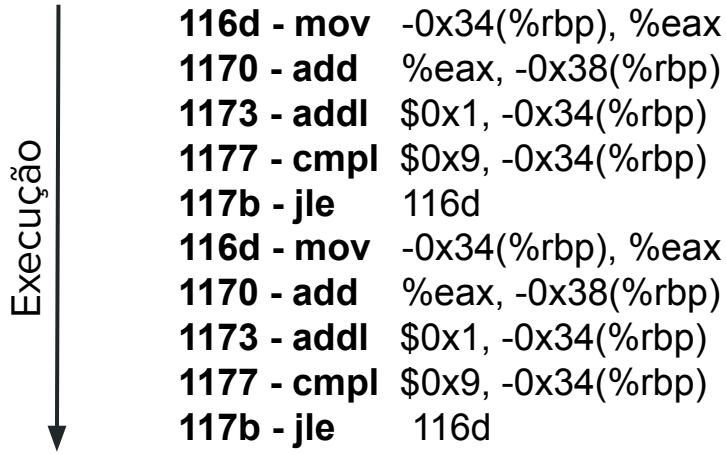
SAPIVe **não foi capaz de realizar conversões** sobre aplicações reais!

[1] Patil, H., Cohn, R. et al. (2004). Pinpointing representative portions of large intel ® itanium ® programs with dynamic instrumentation. Em Int. Symp. on Microarchitecture.

Identificação de Laços

Laço = instruções entre um salto para trás e sua instrução alvo

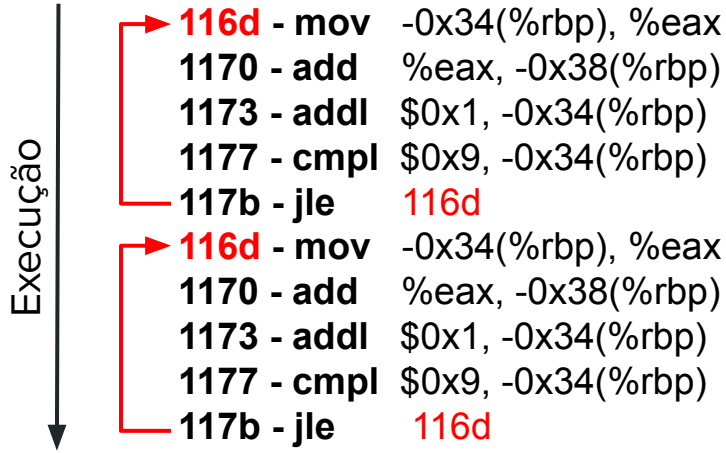
- Chamadas a rotinas e retornos não foram considerados saltos



Identificação de Laços

Laço = instruções entre um salto para trás e sua instrução alvo

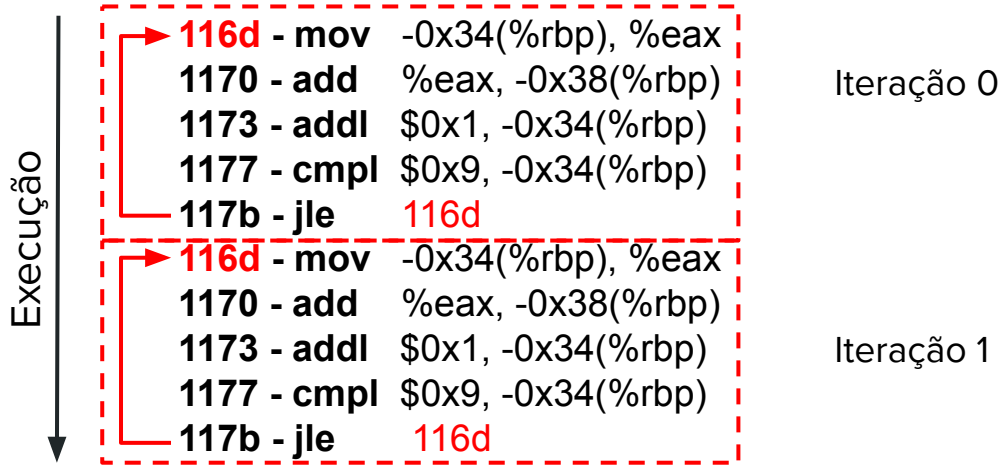
- Chamadas a rotinas e retornos não foram considerados saltos



Identificação de Laços

Laço = instruções entre um salto para trás e sua instrução alvo

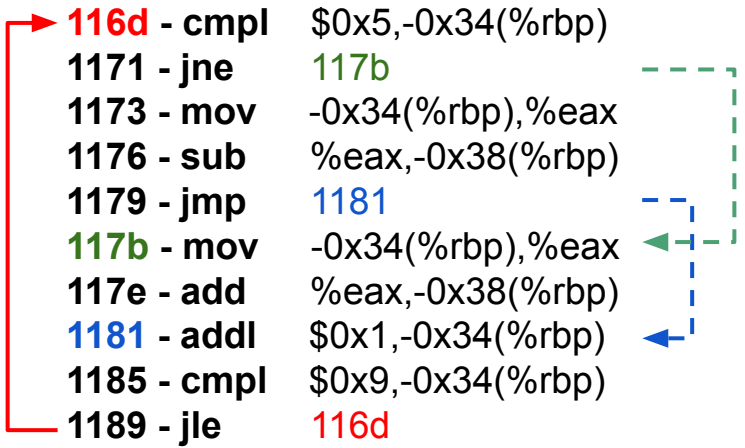
- Chamadas a rotinas e retornos não foram considerados saltos



Identificação de Laços

Laço = instruções entre um salto para trás e sua instrução alvo

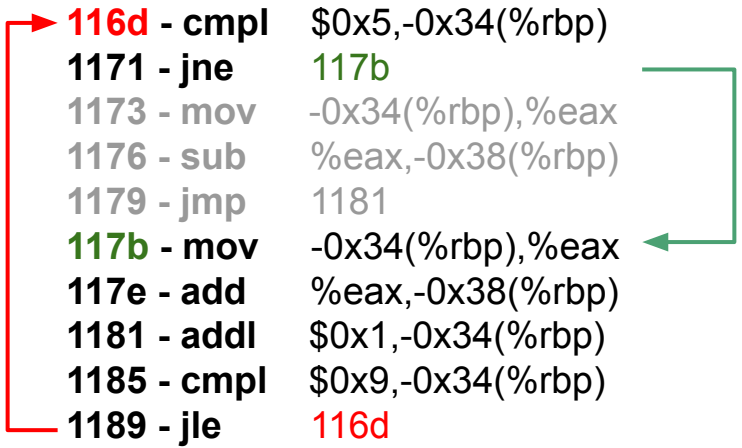
- Chamadas a rotinas e retornos não foram considerados saltos
- Controle de fluxo: o **primeiro caminho** executado foi considerado o corpo do laço



Identificação de Laços

Laço = instruções entre um salto para trás e sua instrução alvo

- Chamadas a rotinas e retornos não foram considerados saltos
- Controle de fluxo: o **primeiro caminho** executado foi considerado o corpo do laço



Classificação de Laços

Classificação dos laços:

- Escalar
- 128 bits
- 256 bits
- 512 bits

Os traços de execução podem não conter todos os tipos de laço

Dificuldades de Conversão

Avaliação das dificuldades de conversão dessas aplicações:

- Tamanho dos laços
- Padrões com instruções adjacentes problemáticas

Dificuldades de Conversão

Uma conversão pode apresentar problemas se instruções adjacentes apresentarem:

- Acessos à memória

```
for (...) {
  Vc[i] = Va[i] + Vb[i];
   Vc[3] += 1;
}
```

Dificuldades de Conversão

Uma conversão pode apresentar problemas se instruções adjacentes apresentarem:

- Acessos à memória
- Alterações no fluxo de execução

```
for (...) {
  Vc[i] = Va[i] + Vb[i];
  Vc[3] += 1;
}
```

```
for (...) {
  if (i % 2)
    Vc[i] = Va[i] + Vb[i];
}
```

Dificuldades de Conversão

Uma conversão pode apresentar problemas se instruções adjacentes apresentarem:

- Acessos à memória
- Alterações no fluxo de execução
- Leitura de registradores escritos por instruções que estão sendo convertidas

```
for (...) {
  Vc[i] = Va[i] + Vb[i];
  Vc[3] += 1;
}
```

```
for (...) {
  if (i % 2)
    Vc[i] = Va[i] + Vb[i];
}
```

```
for (...) {
  tmp = Va[i];
  Vc[i] = tmp + Vb[i];
  x = x + tmp;
}
```

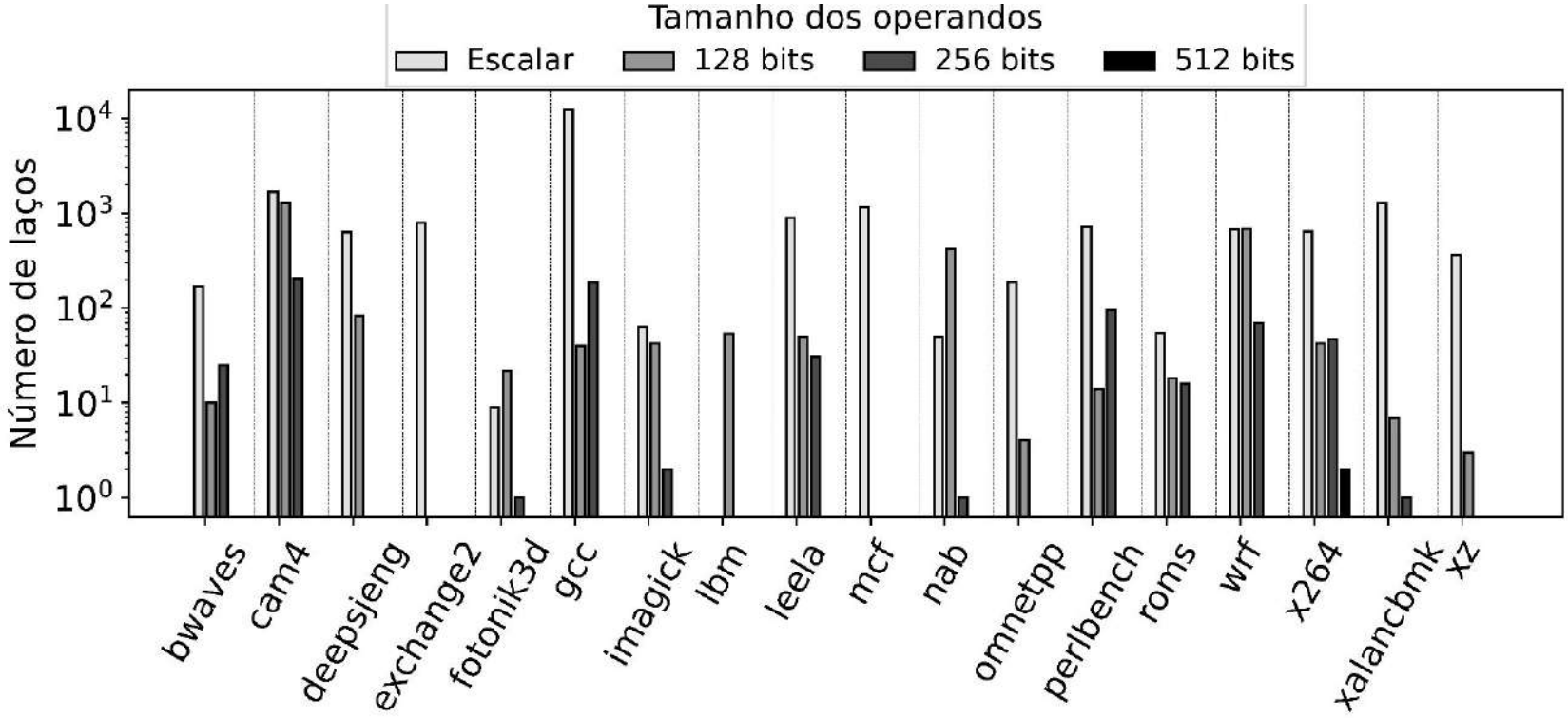
Resultados - Aplicações Reais

Tamanho dos Laços

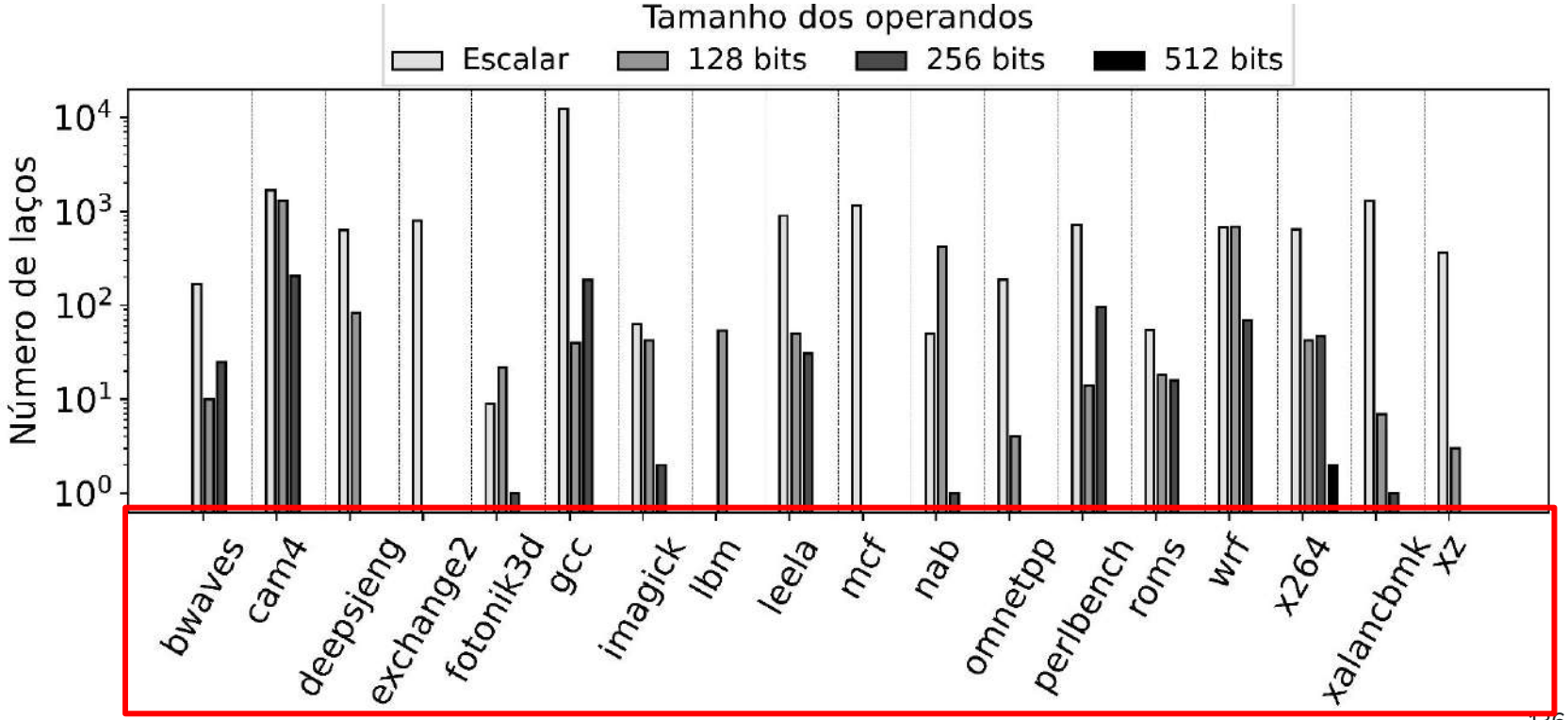
Percentagem com iterações suficientes para conversão:

Tipo	Escalar	128 bits	256 bits	512 bits
Percentagem	24 %	27 %	20 %	100 %

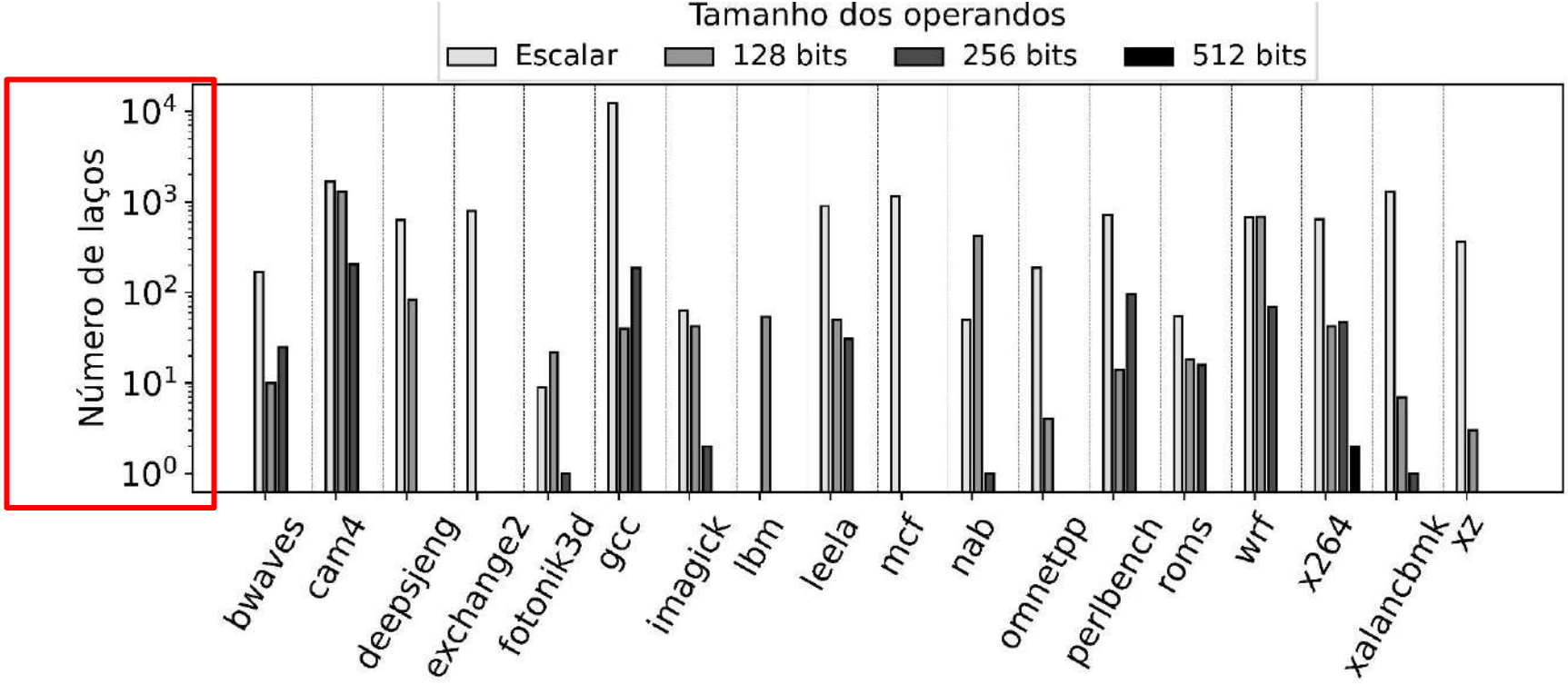
Relevância dos Laços



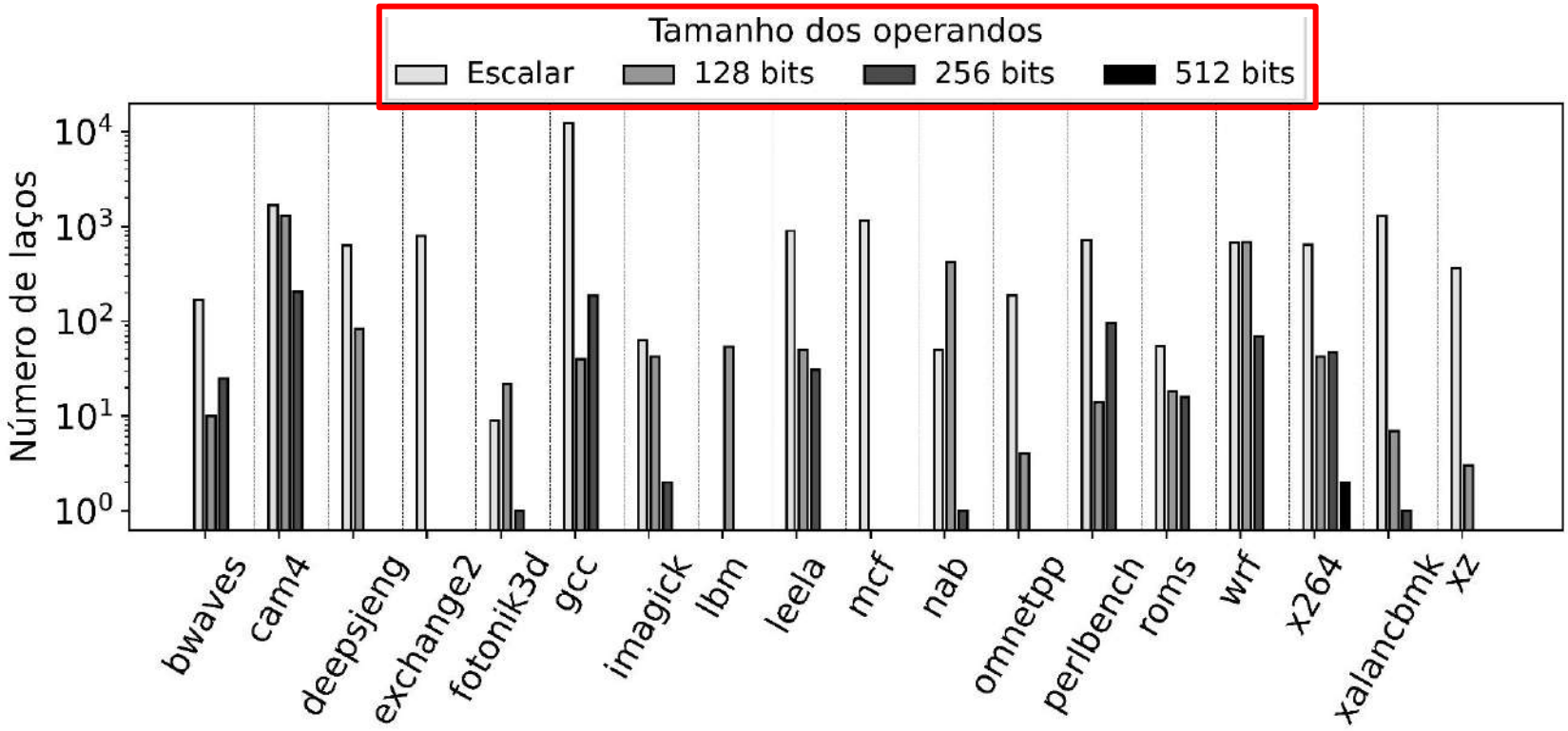
Relevância dos Laços



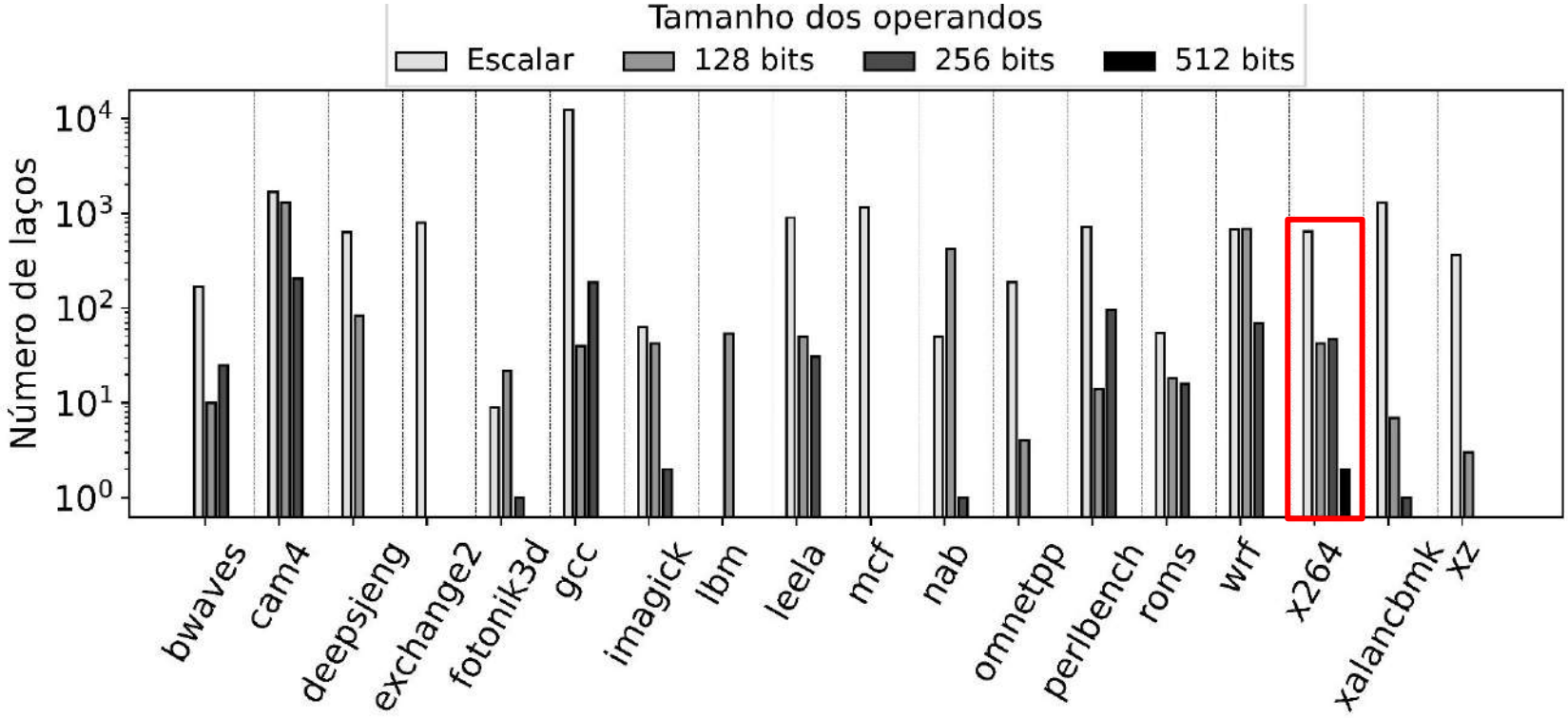
Relevância dos Laços



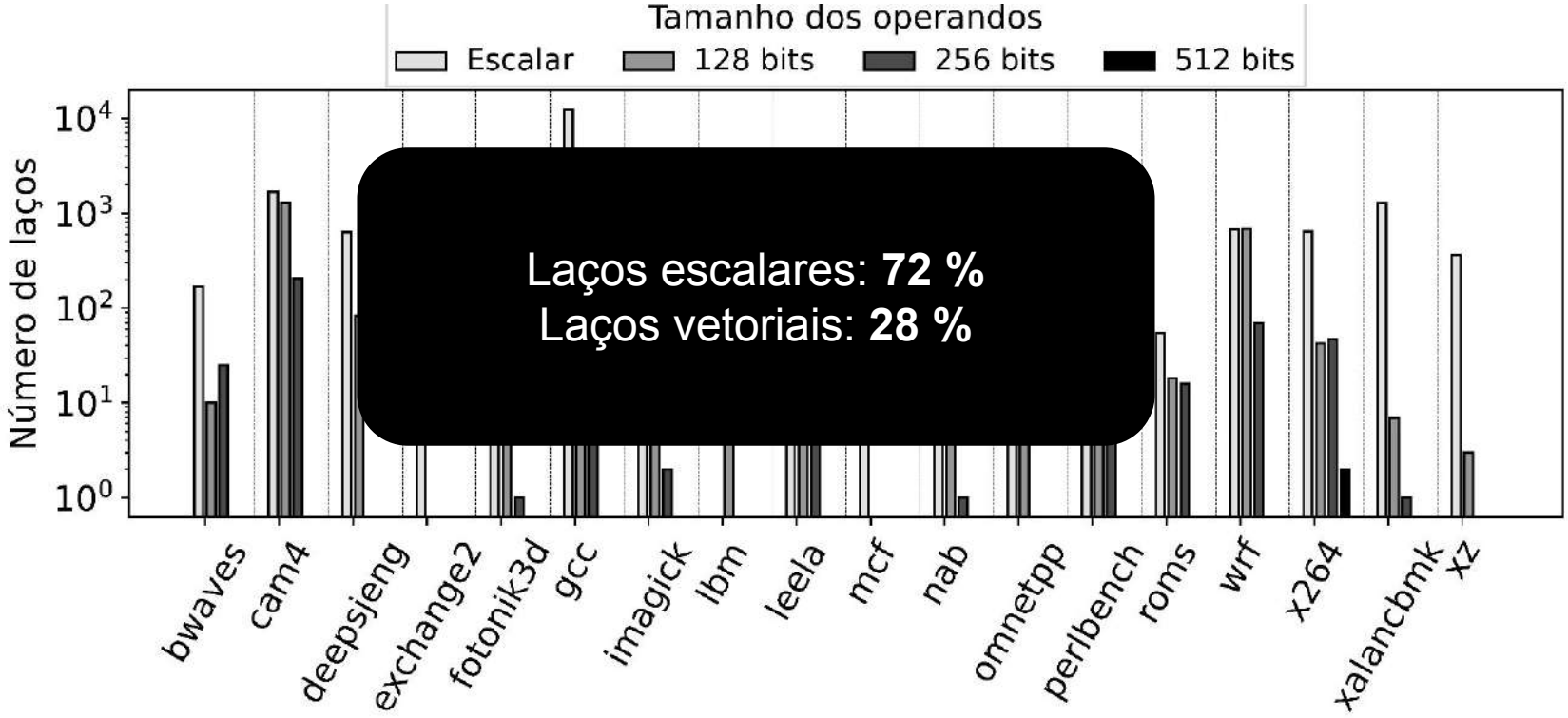
Relevância dos Laços



Relevância dos Laços



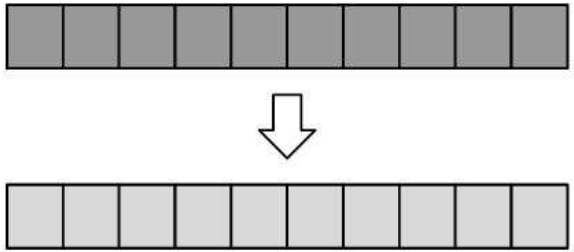
Relevância dos Laços



Inadequações em Laços - Padrões Avaliados

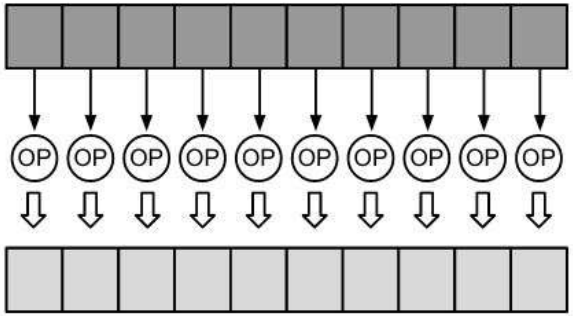
Cópia de memória

(I)



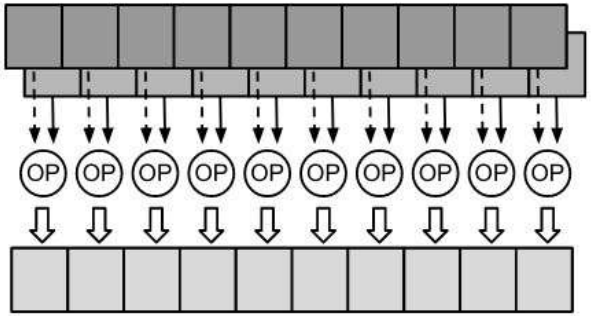
Operação sobre vetor

(II)



Operação entre vetores

(III)



Dificuldades de Conversão

- Acessos externos à memória
  - *Load*: **99 %**
  - *Store*: **98 %**

```
for (...) {
  Vc[i] = Va[i] + Vb[i];
  Vc[3] += 1;
}
```

	LOAD adicional indevido	*STORE* adicional indevido	Desvio de fluxo indevido	Leitura indevida de registrador
bwaves	98 %	98 %	98 %	7 %
cam4	95 %	96 %	91 %	23 %
deepsjeng	100 %	100 %	100 %	48 %
exchange2	96 %	93 %	94 %	23 %
fotonik3d	100 %	100 %	100 %	21 %
gcc	100 %	97 %	97 %	40 %
imagick	99 %	97 %	97 %	33 %
lbm	100 %	100 %	100 %	0 %
leela	100 %	100 %	94 %	36 %
mcf	100 %	100 %	100 %	48 %
nab	98 %	98 %	98 %	17 %
omnetpp	100 %	100 %	100 %	46 %
perlbench	100 %	100 %	98 %	33 %
roms	99 %	99 %	91 %	21 %
wrf	100 %	100 %	64 %	29 %
x264	100 %	99 %	83 %	49 %
xalancbmk	99 %	99 %	98 %	43 %
xz	100 %	99 %	100 %	75 %

Dificuldades de Conversão

- Acessos externos à memória
  - *Load*: 99 %
  - *Store*: 98 %
- Desvio de fluxo: **93 %**

```
for (...) {
  if (i % 2)
    Vc[i] = Va[i] + Vb[i];
}
```

	LOAD adicional indevido	*STORE* adicional indevido	Desvio de fluxo indevido	Leitura indevida de registrador
bwaves	98 %	98 %	98 %	7 %
cam4	95 %	96 %	91 %	23 %
deepsjeng	100 %	100 %	100 %	48 %
exchange2	96 %	93 %	94 %	23 %
fotonik3d	100 %	100 %	100 %	21 %
gcc	100 %	97 %	97 %	40 %
imagick	99 %	97 %	97 %	33 %
lbm	100 %	100 %	100 %	0 %
leela	100 %	100 %	94 %	36 %
mcf	100 %	100 %	100 %	48 %
nab	98 %	98 %	98 %	17 %
omnetpp	100 %	100 %	100 %	46 %
perlbench	100 %	100 %	98 %	33 %
roms	99 %	99 %	91 %	21 %
wrf	100 %	100 %	64 %	29 %
x264	100 %	99 %	83 %	49 %
xalancbmk	99 %	99 %	98 %	43 %
xz	100 %	99 %	100 %	75 %

Dificuldades de Conversão

- Acessos externos à memória
  - *Load*: 99 %
  - *Store*: 98 %
- Desvio de fluxo: 93 %
- Leitura de registradores da conversão: **41 %**

```
for (...) {
  tmp = Va[i];
  Vc[i] = tmp + Vb[i];
  x = x + tmp;
}
```

	LOAD adicional indevido	*STORE* adicional indevido	Desvio de fluxo indevido	Leitura indevida de registrador
bwaves	98 %	98 %	98 %	7 %
cam4	95 %	96 %	91 %	23 %
deepsjeng	100 %	100 %	100 %	48 %
exchange2	96 %	93 %	94 %	23 %
fotonik3d	100 %	100 %	100 %	21 %
gcc	100 %	97 %	97 %	40 %
imagick	99 %	97 %	97 %	33 %
lbm	100 %	100 %	100 %	0 %
leela	100 %	100 %	94 %	36 %
mcf	100 %	100 %	100 %	48 %
nab	98 %	98 %	98 %	17 %
omnetpp	100 %	100 %	100 %	46 %
perlbench	100 %	100 %	98 %	33 %
roms	99 %	99 %	91 %	21 %
wrf	100 %	100 %	64 %	29 %
x264	100 %	99 %	83 %	49 %
xalancbmk	99 %	99 %	98 %	43 %
xz	100 %	99 %	100 %	75 %

Conclusões

Conclusões

- A transferência de dados entre a CPU e a Memória é um problema crescente
- Dispositivos PIM, como a VIMA, podem atenuar esse problema

Conclusões

- A transferência de dados entre a CPU e a Memória é um problema crescente
- Dispositivos PIM, como a VIMA, podem atenuar esse problema
- Pergunta de pesquisa:
  - Auto-vetorização em hardware para processamento em memória pode levar a ganhos de desempenho?

Conclusões

- Memory to CPU data transfer is a growing problem
- PIM devices, as VIMA, can attenuate this problem
- Pergunta de pesquisa:
  - Auto-vetorização em hardware para processamento em memória pode levar a ganhos de desempenho?
- SAPIVe: *Simple AVX to PIM Vectorizer*
  - Obteve uma aceleração de até 5 vezes sobre uma implementação com AVX-512
  - Capaz de obter o potencial de aceleração das *intrinsics* em sistemas futuros
- Apresentamos 4 desafios para futuros mecanismos de conversão dinâmica

Publicações

- Sokulski, R., & Alves, M. (2022). **Proposta de conversão automática em hardware de instruções para execução em memória**. In *Anais da XXII Escola Regional de Alto Desempenho da Região Sul*
- Sokulski, R., Santos, S., & Alves, M. (2022). **On the SPEC-CPU 2017 opportunities for dynamic vectorization possibilities on PIM architectures**. In *Anais do XXIII Simpósio em Sistemas Computacionais de Alto Desempenho* [Menção honrosa]
- Sokulski, R. M., Santos, P. C., dos Santos, S. R., & Alves, M. A. Z. (2022). **SAPIVe: Simple AVX to PIM Vectorizer**. In 2022 XII Brazilian Symposium on Computing Systems Engineering (SBESC) [Entre os *best papers*]

Conversão dinâmica de instruções para processamento vetorial em memória

Rodrigo Machniewicz Sokulski
Orientador: Marco Antonio Zanata Alves