Pontifícia Universidade Católica de Minas Gerais

Localidades Temporal e Espacial no Acesso à Memória

Gabriel Henrique, Lucas Araújo, Lucas de Oliveira, Pedro Miranda, Vincíus Simões

Introdução

Introdução

- O trabalho desenvolvido busca analisar o impacto de alterações na memória cache e virtual, explorando localidades temporal e espacial para otimização do desempenho.
- A eficiência dos sistemas computacionais modernos depende fortemente da otimização do tempo de acesso à memória, questões essas exploradas ao longo do projeto.
- Como ferramente foi utilizado o Simulador Amnesia.

Hierarquia de Memória

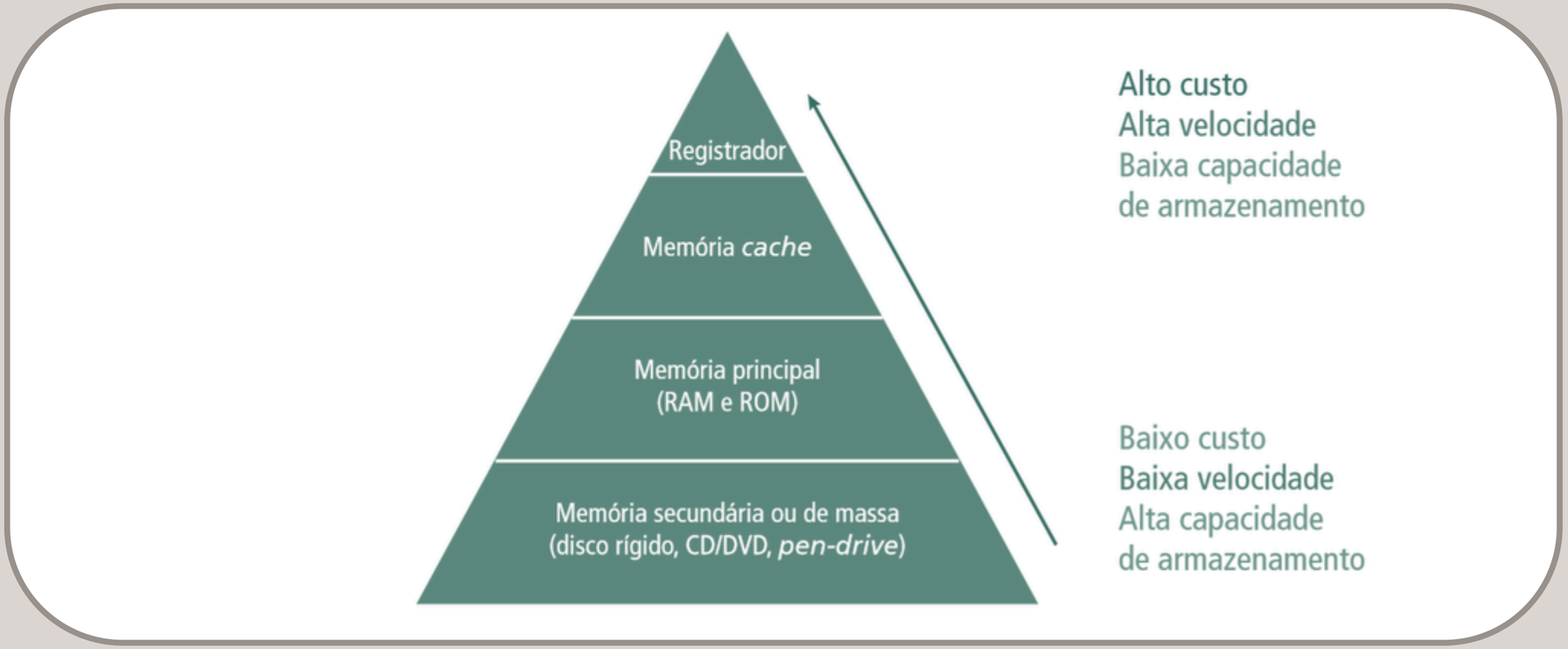
Conceito

A hierarquia de memória organiza diferentes tipos de memória em um sistema, priorizando a eficiência de acesso aos dados. Ela consiste em níveis com variações de velocidade, capacidade e custo, onde memórias mais rápidas, como cache, estão próximas ao processador, e memórias mais lentas, como discos, estão mais distantes.

Hierarquia de Piramide

A memória em pirâmide representa a hierarquia de memória em forma de camadas. No topo, estão as memórias rápidas e pequenas (registros e cache), seguidas pela RAM, com equilíbrio de velocidade e capacidade, e, na base, as memórias secundárias (discos), mais lentas, porém maiores.

Representação da Hierarquia de Piramide



Localidade de Memória

Localidade de memória é um princípio que descreve como os dados são acessados em um sistema computacional, visando melhorar o desempenho ao aproveitar padrões previsíveis de uso de memória

Localidade de memória pode ser dividida em duas categorias principais, **localidade temporal** e **localidade espacial**

Localidade Temporal

A localidade temporal baseia-se no princípio de que, uma vez que um dado é acessado, ele será provavelmente acessado novamente em um futuro próximo

Essa localidade sugere que armazenar dados acessados recentemente em cache pode reduzir o tempo de acesso subsequente, uma vez que esses dados não precisarão ser carregados novamente da memória principal,

Localidade Espacial

A localidade espacial, por sua vez, baseia-se no princípio de que, se um determinado dado foi acessado, é provável que os dados próximos a ele na memória também sejam acessados em breve.

Essa técnica de buscar dados próximos em antecipação ao uso é usada tanto no cache quanto na memória virtual, melhorando o desempenho ao reduzir o número de acessos à memória lenta

Arquitetura proposta

Características da Arquitetura

Dois Traços de Execução

- Exploram princípios de localidade (temporal e espacial).
- Endereços variam de 00 a 31 em hexadecimal.

Traço Temporal

- Foca no reuso de dados recentes para maximizar a localidade temporal.

Traço Espacial

Acessa dados próximos na memória para explorar a localidade espacial.

Associatividade do Cache

- Cache ajustado para diferentes níveis de associatividade (2, 4, 8, 16 vias).
- Reduz colisões e melhora o desempenho.

Cache de Múltiplos Níveis

- Sistema com cache L1 e L2 testado com variações de tamanho.
- Objetivo: reduzir acessos à memória principal e diminuir perdas de cache.

Resultados Esperados

Modificações na estrutura do cache devem reduzir a taxa de erros e melhorar o tempo de acesso à memória.

Metodologia

Para se explorar as localidades temporal e espacial foram cridas duas fases, tendo 3 cenarios cada, alem do cenerio base

Primeira Fase:
Memória Cache

Utilização de Memória Principal e de Memória Cache

Cenário 1: Alteração na quantidade de palavras por bloco

Cenário 2: Alteração na associatividade da Cache.

Cenário 3: Alteração nos níveis da Cache.

Segunda Fase:
Memória Virtual

Utilização de Memória Principal e Memoria Secundaria (Disco), além da TLB

Cenário 1: Alteração na quantidade de blocos por página na Tabela de Páginas

Cenário 2: Alteração no tamanho de armazenamento da TLB

Cenário 3: Não inclusão da TLB

Arquitetura Proposta Com Memória Cache (Primeira Fase)

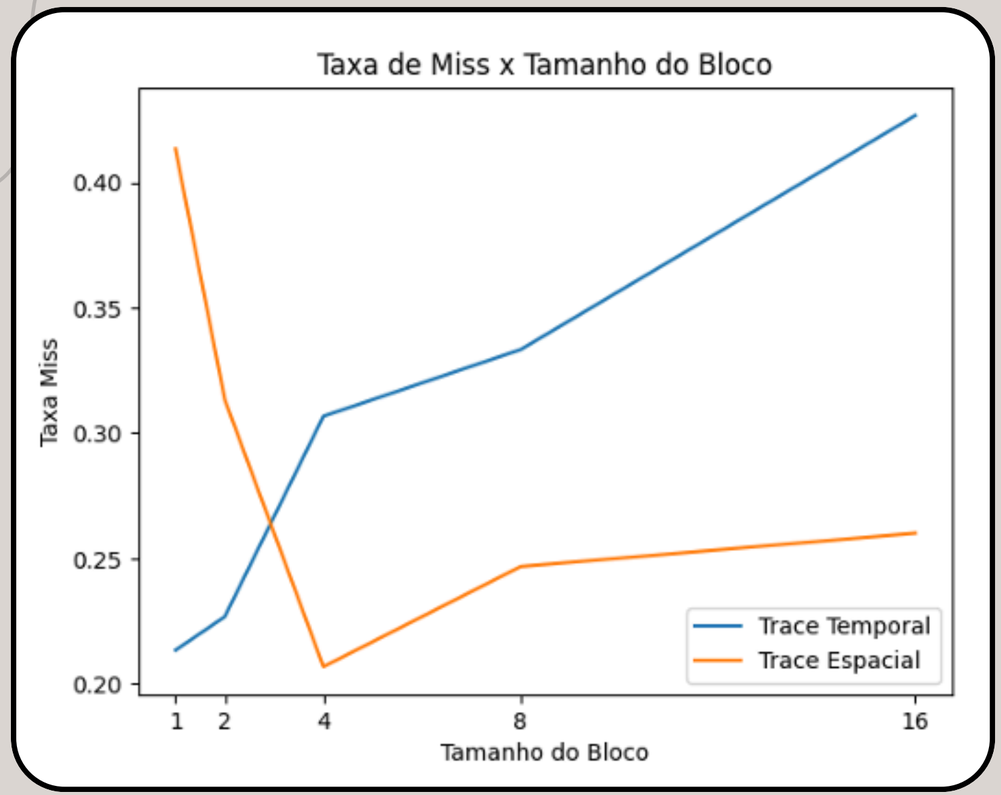
Memória Principal	**Cenário Base**	**Cenário 1**	**Cenário 2**	**Cenário 3**
Palavras por Bloco	1	**2, 4, 8, 16**	1	1
Tamanho de Dados (Bytes)	32	32	32	32
Ciclos por acesso para leitura	10	10	10	10
Ciclos por acesso para escrita	20	20	20	20
Tempo de ciclo (ms)	1	1	1	1
Memória Cache	Unificada	Unificada	Unificada	**Multinível**
Palavras por Bloco	1	**2, 4, 8, 16**	1	1
Tamanho de Dados (Bytes)	16	16	16	**L1 = 8 e L2 = 16** **L1 = 8 e L2 = 32** **L1 = 16 e L2 = 32**
Ciclos por acesso para leitura	1	1	1	1
Ciclos por acesso para escrita	2	2	2	2
Tempo de ciclo (ms)	1	1	1	1
Nº de vias	1	1	**2, 4, 8, 16**	1
Política de escrita	Write Through	Write Through	Write Through	Write Through
Algoritmo de substituição	LRU	LRU	LRU	LRU

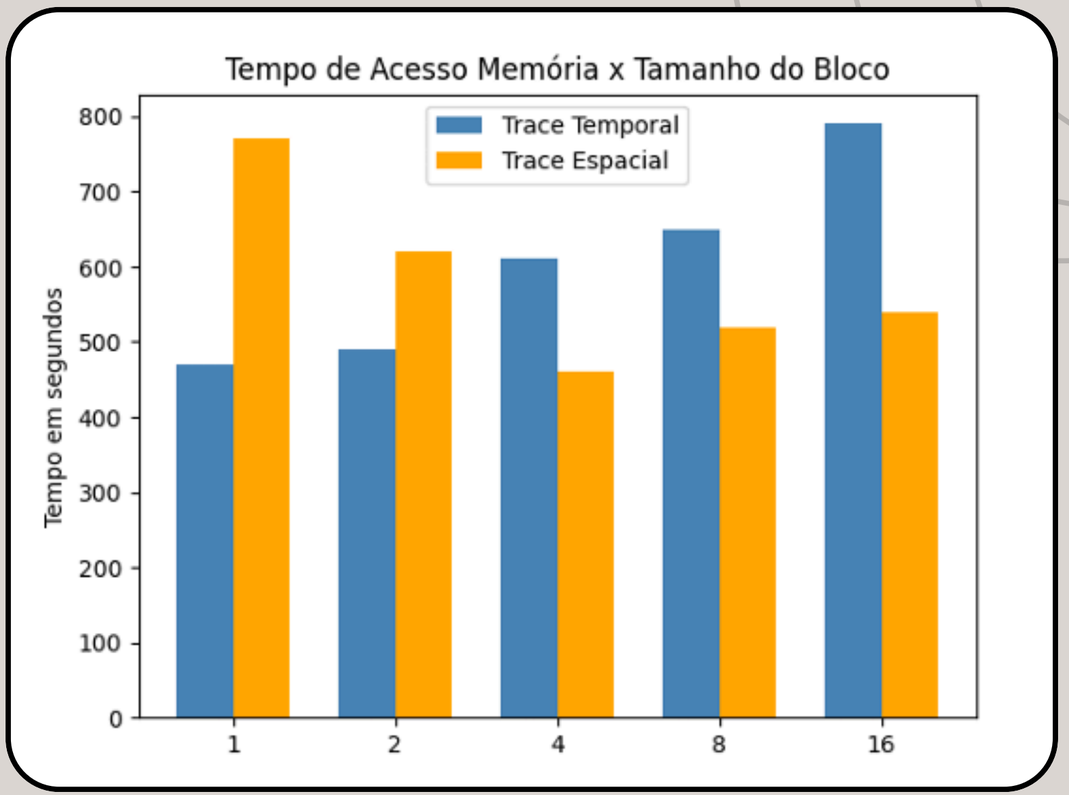
Memória Principal	**Cenário Base**	**Cenário 1**	**Cenário 2**	**Cenário 3**
Palavras por Bloco	1	1	1	1
Tamanho de Dados (Bytes)	16	16	16	16
Ciclos por acesso para leitura	10	10	10	10
Ciclos por acesso para escrita	20	20	20	20
Tempo de ciclo (ms)	1	1	1	1
Blocos por página na Tabela de Páginas	2	**4, 8, 16**	2	2
Memória Secundária	-			
Palavras por Bloco	1	1	1	1
Tamanho de dados (Bytes)	32	32	32	32
Ciclos por acesso para leitura	100	100	100	100
Ciclos por acesso para escrita	200	200	200	200
Tempo de ciclo (ms)	1	1	1	1
Algoritmo de substituição	FIFO	FIFO	FIFO	FIFO
TLB	Sim	Sim	Sim	**Não**
Tamanho de dados (Bytes)	8	8	**4, 6, 10, 12**	**-**
Ciclos por acesso para leitura	1	1	1	**-**
Ciclos por acesso para escrita	2	2	2	**-**
Tempo de ciclo (ms)	1	1	1	**-**
Algoritmo de substituição	LRU	LRU	LRU	**-**

Arquitetura Proposta Com Memória Virtual (Segunda Fase)

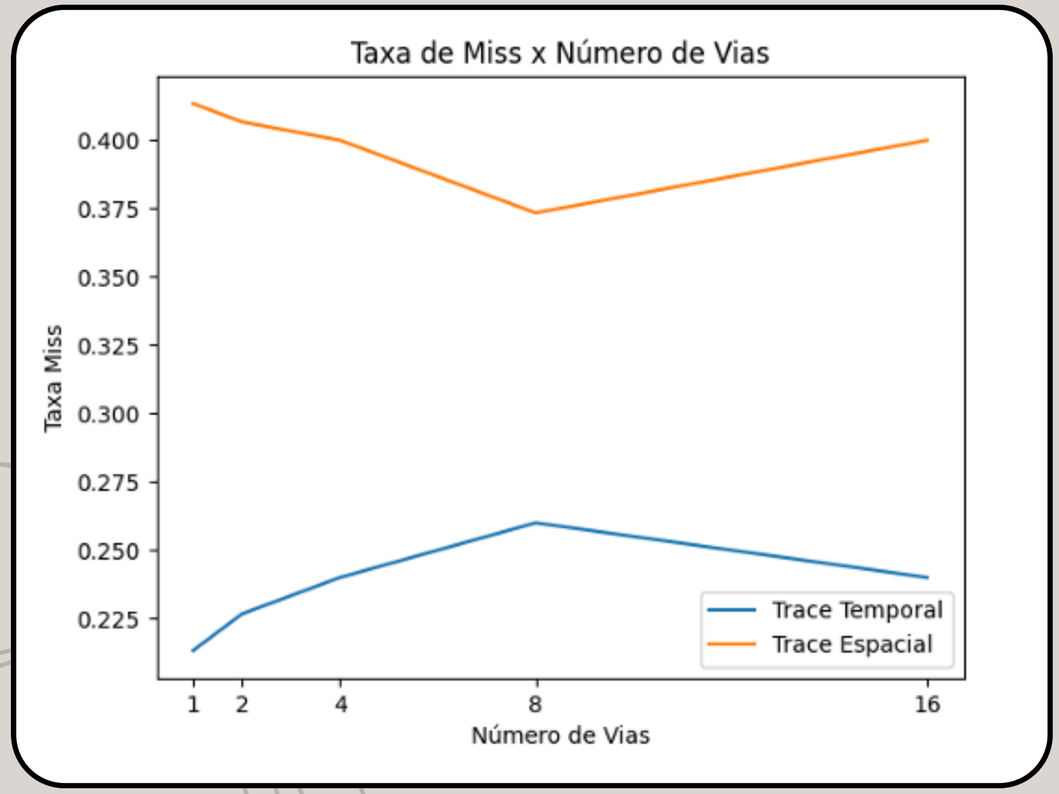
Resultados

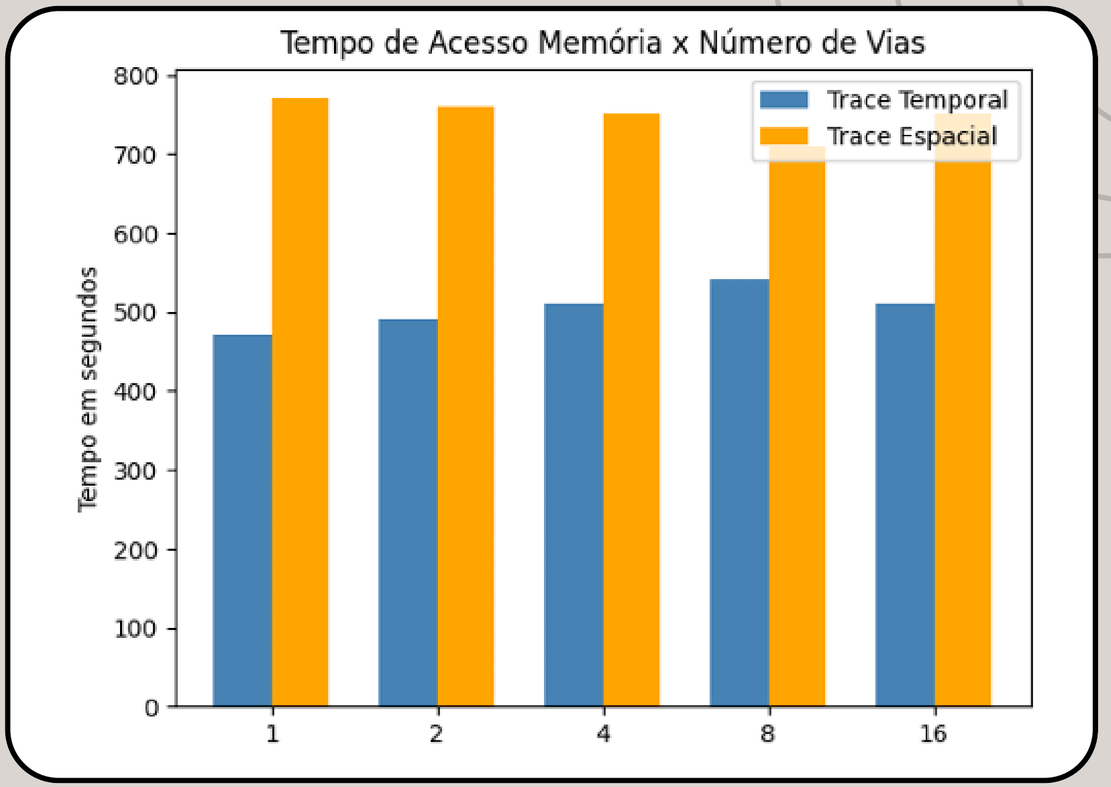
Primeira Fase - Cenário 01



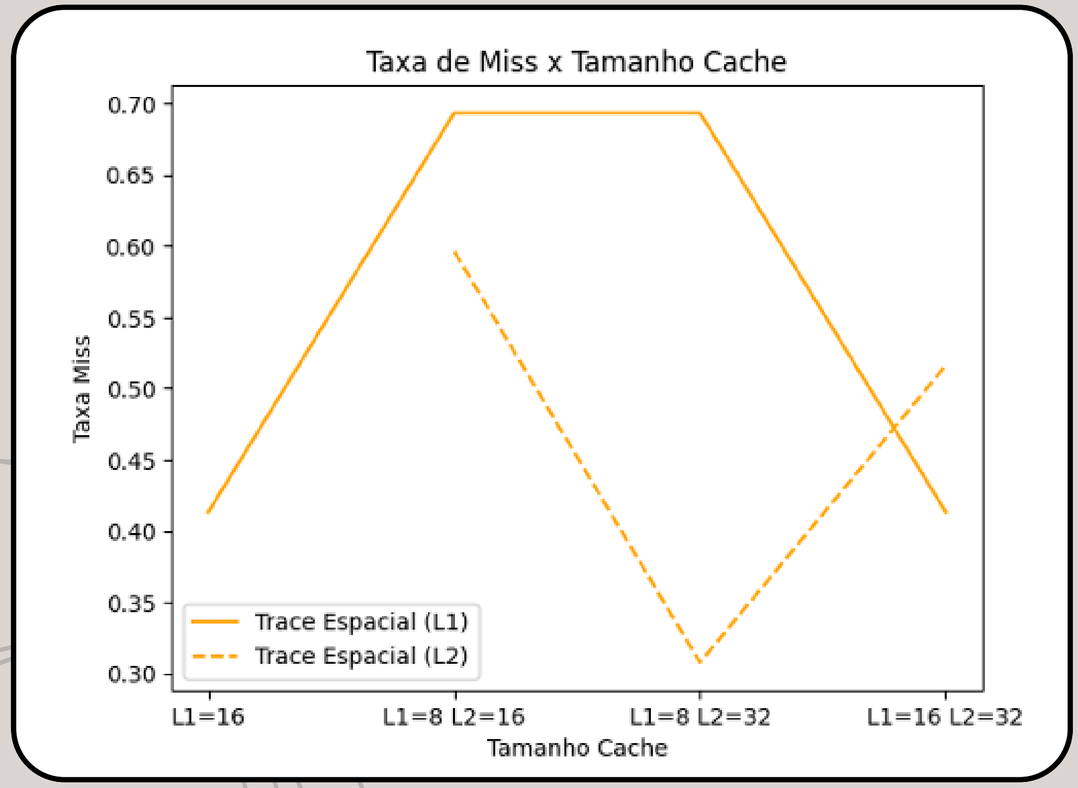


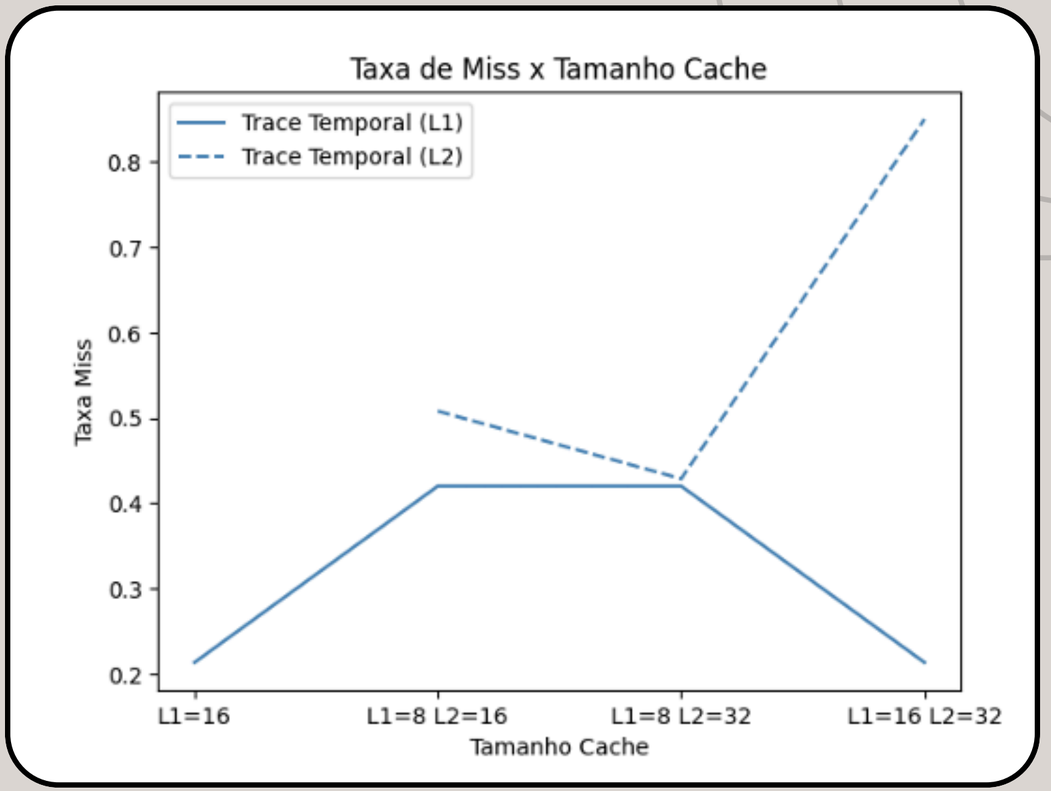
Primeira Fase - Cenário 02



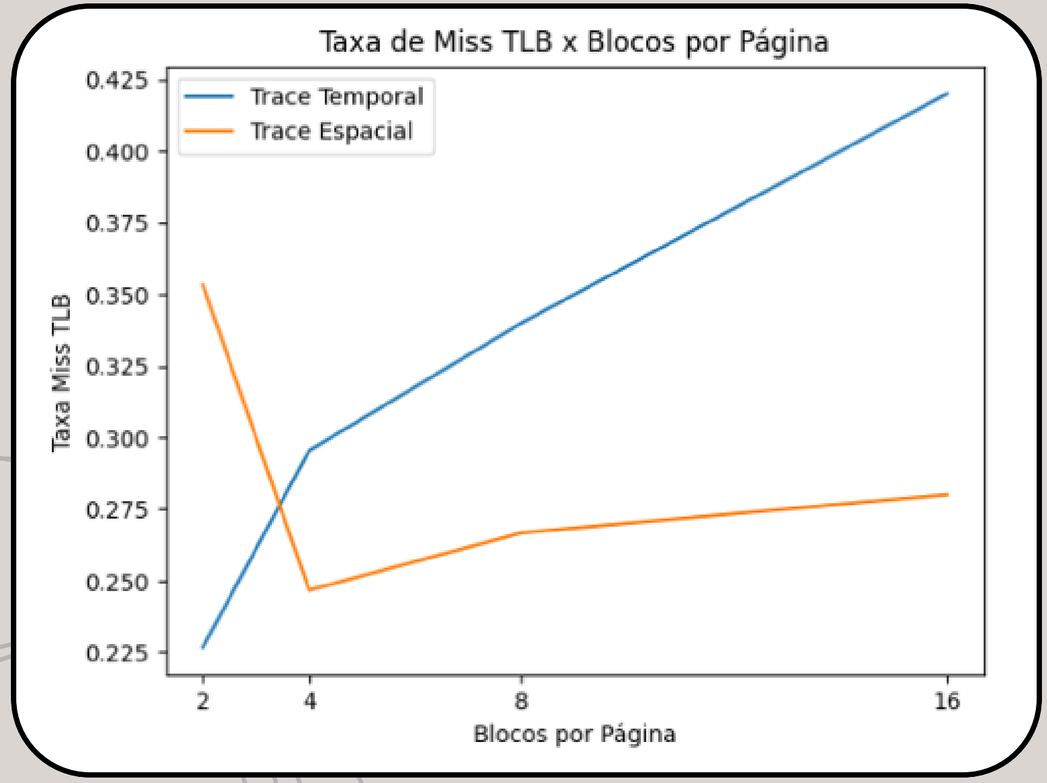


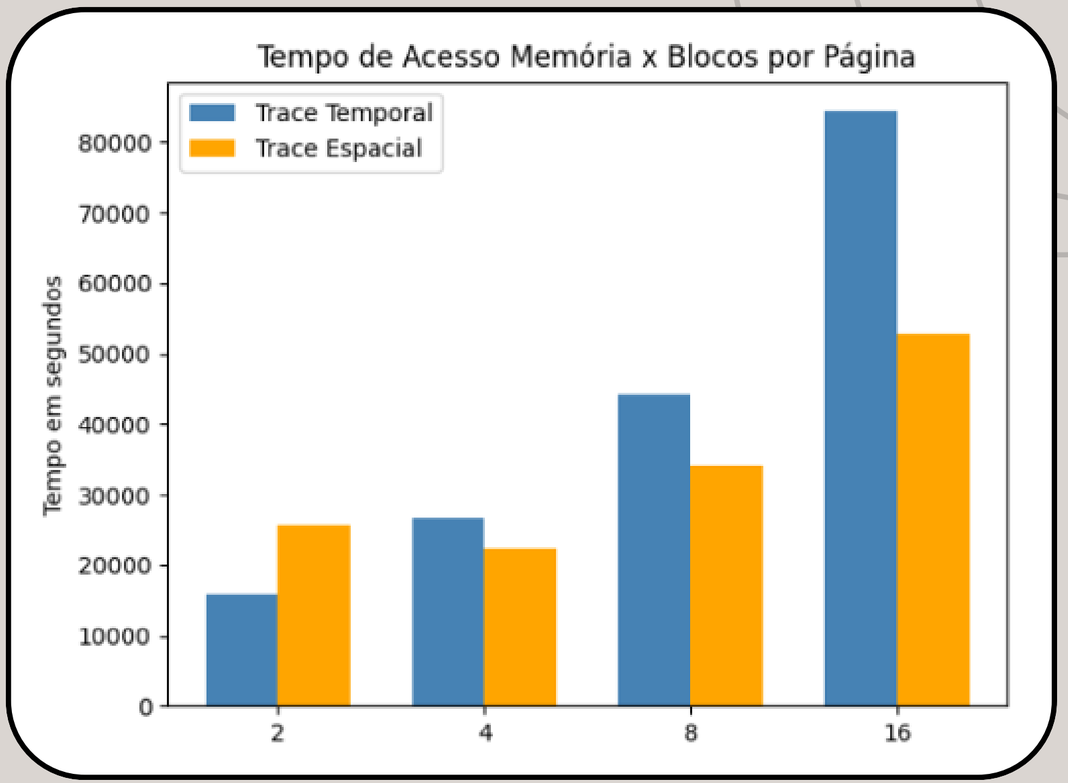
Primeira Fase - Cenário 03



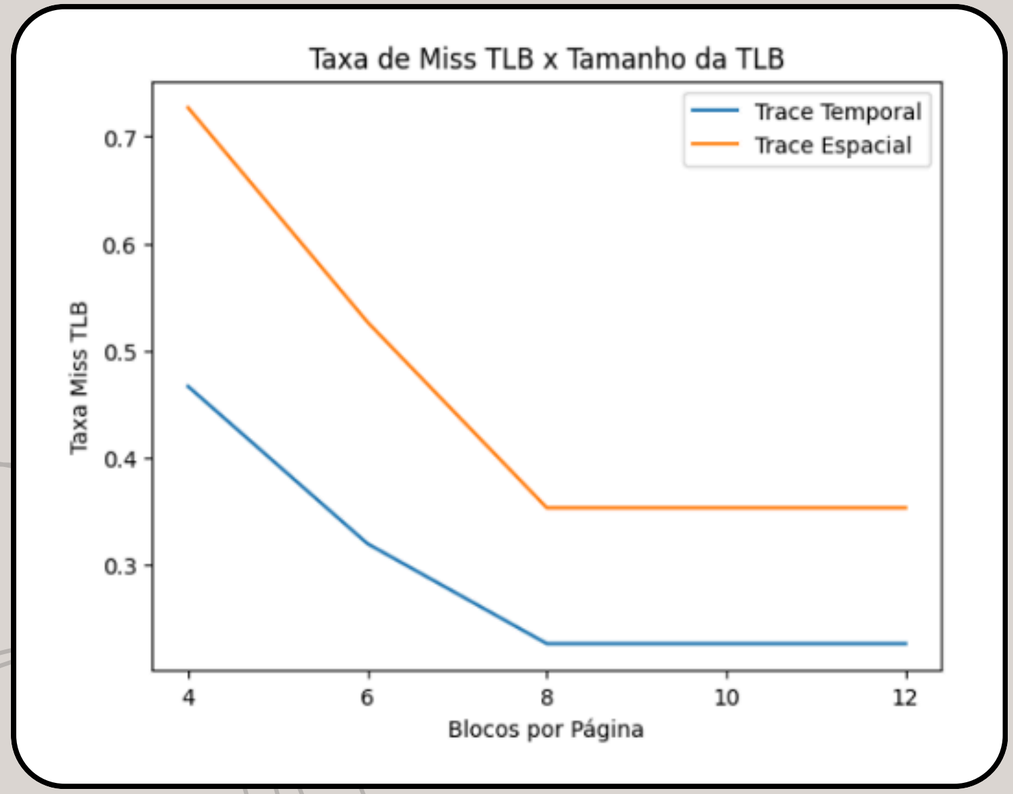


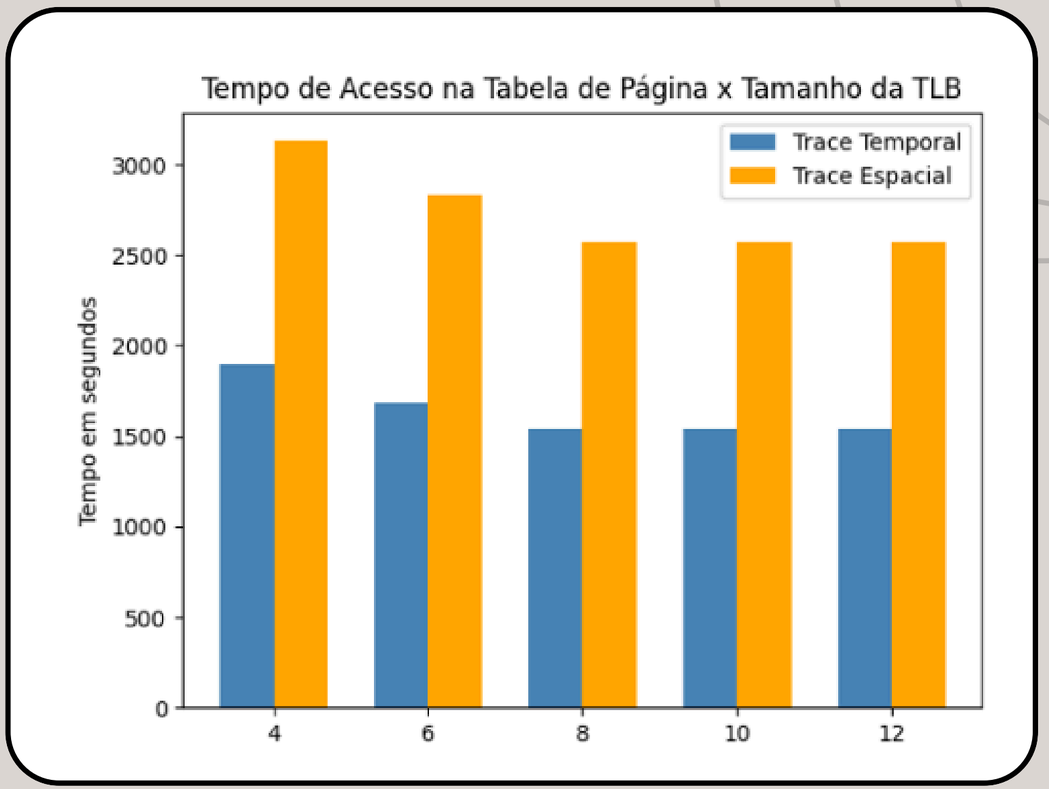
Segunda Fase - Cenário 01





Segunda Fase - Cenário 02





Segunda Fase - Cenário 03

	Sem TLB		Com TLB	
	Trace Espacial	**Trace Temporal**	**Trace Espacial**	**Trace Temporal**
Disco	102	60	102	60
Memória Principal	810	600	712	484
Tabela de Página	150	150	53	34
TLB	-	-	150	150
	Número de Acessos			

Conclusões

Neste estudo, exploramos como otimizar o acesso à memória em sistemas de computação, focando nos princípios de localidade temporal e espacial.

Ao aumentar a TLB, percebemos que a eficiência na tradução de endereços melhora, mas chega um momento em que isso se estabiliza.

Além disso, ao ajustar os tamanhos dos blocos, notamos uma redução nas taxas de perda de cache. No entanto, aumentar demais os blocos podem prejudicar a localidade temporal e aumentar a chance de perda de dados.

Em resumo, entendemos que uma hierarquia de memória bem planejada é crucial para garantir um acesso eficiente aos dados e um desempenho superior do sistema.

Obrigado!