

## AEDS III

Módulo: Memória Secundária  
Aula 1: Localidade de Referência  
Prof: Olga Goussevskaia

### Conteúdo

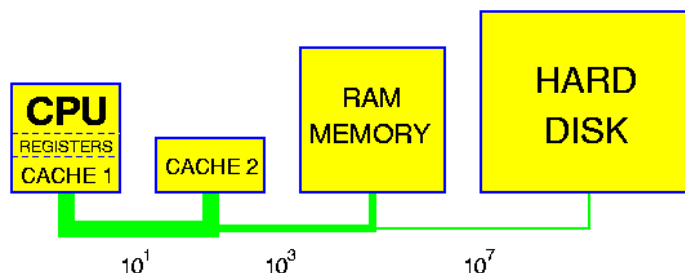
- Localidade de Referência
- Memória Virtual e Paginação
- Ordenação Externa
  - Intercalação balanceada
  - Seleção por substituição
  - Intercalação polifásica
  - Quicksort externo
- Árvores B, B\*

## Livro de Referência

- **Projeto de Algoritmos** com implementação em Pascal e C. Nivio Ziviani. 3a edição.
- Localidade de Referência (não está no livro)
- Memória Virtual e Paginação (6.1)
- Ordenação Externa (4.2)
  - Intercalação balanceada
  - Seleção por substituição
  - Intercalação polifásica
  - Quicksort externo
- Árvores B, B\* (6.2, 6.3)
- 2a PROVA: 10/10 (turma 2a/4a) e 11/10 (3a/5a)

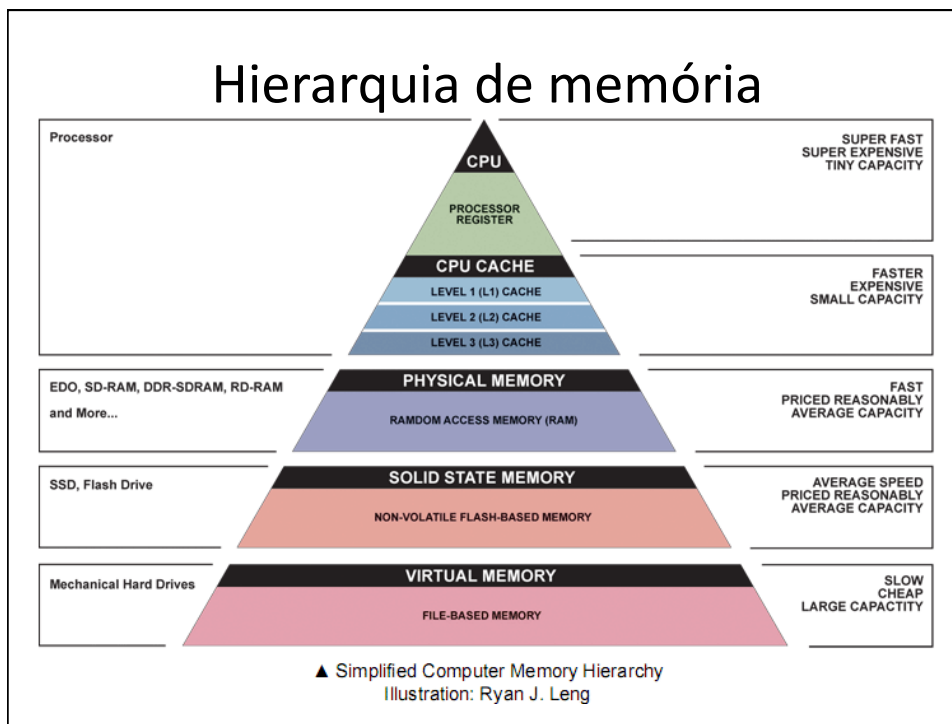
## Perspectiva Histórica

- Primeiros computadores: processador + fita
- Processadores modernos:
  - + Capacidade crescente: Cache: O(MB), RAM: O(GB), HD: O(TB)
  - + Latência crescente
  - Custo por BYTE decrescente



Indicated are approximate numbers of clock cycles to access the various elements of the memory hierarchy

## Hierarquia de memória



## Localidade de Referência

- Padrões de acesso a dados observados desde os primeiros sistemas de computação:
  - **Temporal**: se um dado é acessado uma vez, há uma probabilidade grande dele ser acessado novamente num futuro próximo.
  - **Espacial**: se um dado é acessado uma vez, há uma probabilidade grande do seu vizinho ser acessado em breve.

## Localidade de Referência

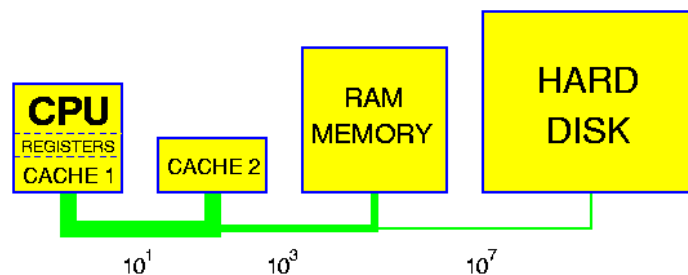
- Estrutura mais clássica de programação:
  - Loop:
 

```
for (i = 0; i < n; i++) {
    vetor[i] = ...;
}
```
- Localidade de referência espacial:
  - vetor[0], vetor[1], ..., vetor[n-1]
- Localidade de referência temporal:
  - As instruções compoendo o corpo do loop
- Mais um exemplo:
  - Árvore: os nós são sempre acessados pela raiz

## Localidade de Referência Temporal

- Quanto maior, melhor!
- Uma vez trazido para a o topo da hierarquia, o custo de acessos seguintes a um dado é bem mais baixo.

### MEMORY HIERARCHY

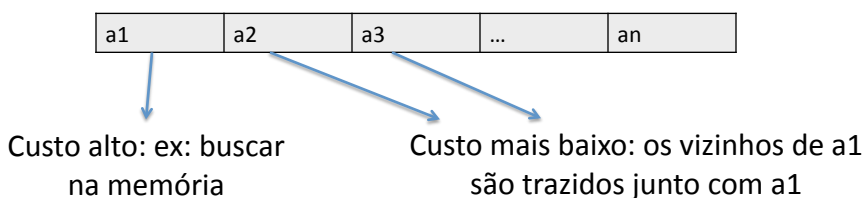


Indicated are approximate numbers of clock cycles to access the various elements of the memory hierarchy

## Localidade de Referência Espacial

- Como pode ser explorada?
  - Estruturas baseadas em paginação e segmentação
    - Em hardware, a partir da arquitetura 386 (segmentation fault)
  - Paginação: Acessos em blocos, de tamanho:
    - L1: 128 bytes
    - L2: um pouco maior
    - RAM: 4 a 8 KB
    - Disco: 4 a 8 KB

Isso torna os sistemas de memória modernos eficientes!



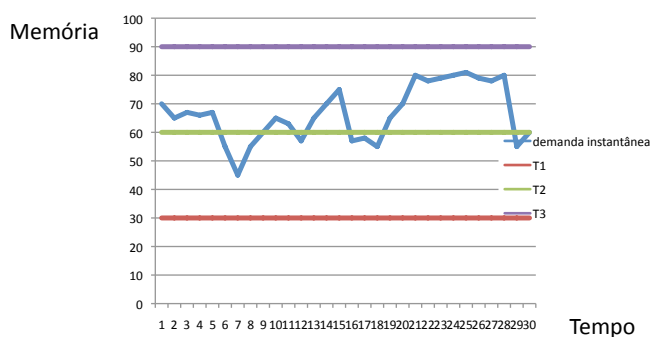
## Localidade de Referência Espacial

- Importância (compromisso) do bloco trazido:
  - Se for grande demais, dados inúteis podem ocupar espaço na cache (fragmentação)
  - Se for pequeno demais, acessos desnecessários a níveis mais baixos da memória (com custo muito alto)
- Exemplo: google maps
  - Arrastar o mapa um pouquinho carrega rápido, arrastar muito demora bem mais
  - O engenheiro do maps deve tomar uma decisão sobre quantos dados “em volta” do ponto clicado/requisitado trazer no primeiro clique
- Fabricantes de hardware
  - Devem analisar esses tradeoffs muito bem para produzirem memórias eficientes



## Localidade de Referência: Quantificando

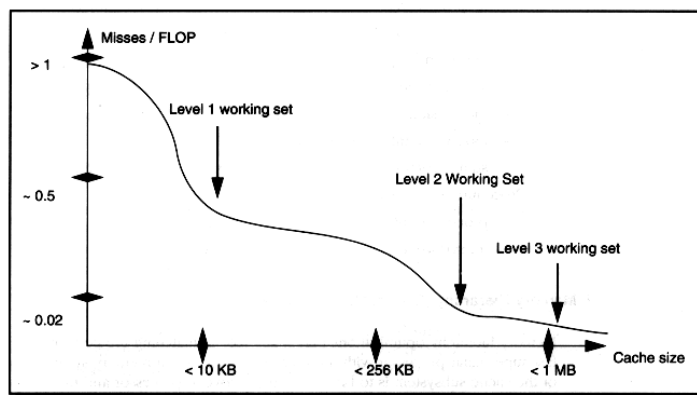
- Conceito 1: “working set”: quantidade de dados referenciada por um processo em um intervalo de tempo.



- Pergunta: qual o tamanho de memória ideal? T1, T2 ou T3?

## Localidade de Referência: Quantificando

- Conceito 2: Problema de dimensionamento



- Pergunta: o que determina essa curva?

## Localidade de Referência: Quantificando

- O que determina a curva de “misses”?
  - O fluxo do programa:
    - 1o grau: vetor mais acessado do programa
    - 2o grau: 2o vetor mais acessado do programa
    - Etc.
- Como medir localidade de referência?
  - Temporal
  - Espacial
  - Popularidade
    - Dados muito pop. têm uma localidade temporal inerente que independe da ordem de acessos.
    - Pergunta: dado um log de acessos, como distinguir boa localidade temporal de popularidade?

## Localidade de Referência Temporal: Quantificando

- **Distância de Pilha**
- Ex: dados: A, B, C, D
- Sequência de acessos:
 

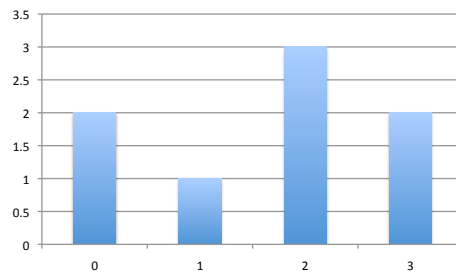
A, B, D, B, A, C, C, B, D, D, A, B

					C	C	B	D	D	A	B
		D	B	A	A	A	C	B	B	D	A
	B	B	D	B	B	B	A	C	C	B	D
A	A	A	A	D	D	D	D	A	A	C	C
-	-	-	1	2	-	0	2	3	0	3	2

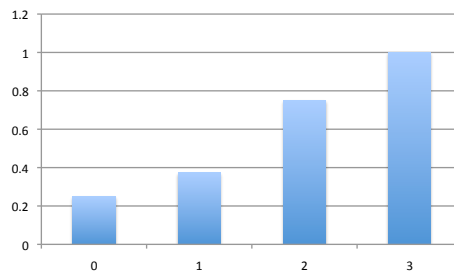
Temporal

## Localidade de Referência Temporal: Quantificando

- Histograma



- Histograma cumulativo



- Pergunta: localidade boa ou ruim?
- Dimensionamento:
  - Quanto custa cada tamanho?
  - O programa pode ser reestruturado para melhorar a localidade?

## Localidade de Referência Espacial: Quantificando

- Distância de acessos

– Ex. armazenamento de dados na memória:

A	B	C	D
---	---	---	---

- Sequência de acessos:

A, B, D, B, A, C, C, B, D, D, A, B

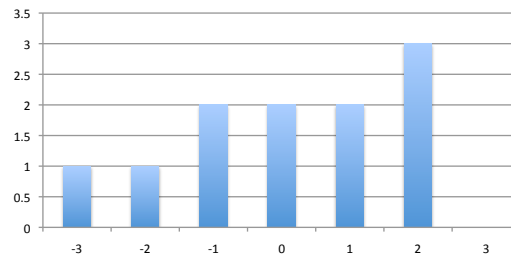
	1	2	-2	-1	2	0	-1	2	0	-3	1
--	---	---	----	----	---	---	----	---	---	----	---

↖  
Espacial



## Localidade de Referência Espacial: Quantificando

- Histograma de disposição espacial



- Pergunta: qual é o melhor esquema de organização da memória, i.e., número de registros por página?

## Localidade de Referência Espacial: Quantificando

- Pergunta: qual é o melhor esquema de organização da memória, i.e., número de registros por página?

– 1 registro por página: só o caso 0 é satisfeito



– 2 registros por página: alguns casos -1 e +1 serão satisfeitos



– Calcula média ponderada das distâncias de pilha

- Temporal:  $(2 \times 0 + 1 \times 1 + 3 \times 2 + 2 \times 3) / 8 = 13/8$ , (alta=> ruim)
- Espacial: soma deve ser por módulo

## Localidade de Referência: Analisando

- Pergunta: dado o fluxo de execução do programa, qual a melhor disposição dos dados na memória?
  - A B C D ou C B D A?
- Ex: dada uma árvore, o programa faz busca em largura ou profundidade?
- Ex: armazenamento: ABCD, seq. acessos: ADADAD
  - Localidade Temporal: 1,1,1... (boa),
  - Espacial: 3,-3,3,-3,... (n-1, ruim)
- Ex: armazenamento: ABCD, seq. acessos: ABCDCBAB
  - Localidade Temporal: 1,2,3,2..., (n-1, ruim)
  - Espacial: 1,1,1,-1,-1,-1,1,1,1,.. (boa)

## Localidade de Referência: Analisando

- Histograma Bidirecional (temporal x espacial)

3	1					1	
2			2		1		
1		1					
0				2			
	-3	-2	-1	0	1	2	3

- Localidade temporal pode ser boa em detrimento da espacial e vice-versa

## Localidade de Referência: Recapitulando...

- O segredo para a hierarquia de memória funcionar bem é a localidade de referência:
  - Temporal: caches mais rápidos
  - Espacial: paginação
- É essencial entender a natureza da aplicação
  - Distância de pilha
  - Distância de acesso
- Moral da história:
  - Um programa com complexidade  $O(n \log n)$  e localidade de referência ruim pode ser mais lento que um programa  $O(n^2)$  com localidade boa.