External Memory Algorithms and Data Structures

Winter 2004/2005

Riko Jacob
Peter Widmayer

Assignments: Yoshio Okamoto

External Memory Algorithms and Data Structures

Winter 2004/2005

Riko Jacob
Peter Widmayer

Assignments: Yoshio Okamoto

based on a lecture in Aarhus, DK, by
Gerth Stølting Brodal and
Rolf Fagerberg

Course

Lectures:

- Based on articles.
- Theoretical.
- New stuff: 1995-2004.
- Aim: General principles and methods.

Course

Lectures:

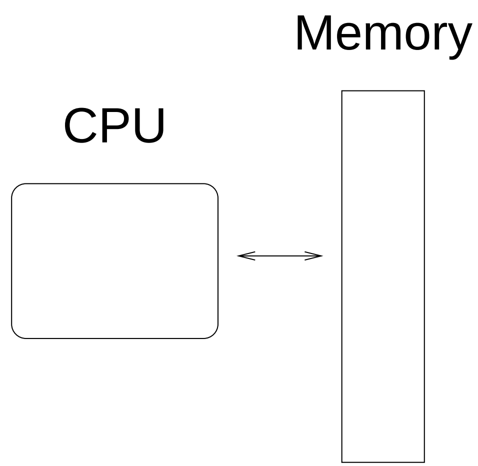
- Based on articles.
- Theoretical.
- New stuff: 1995-2004.
- Aim: General principles and methods.

Homepage:

- `www.ti.inf.ethz.ch/ew/courses/EMADS04/`

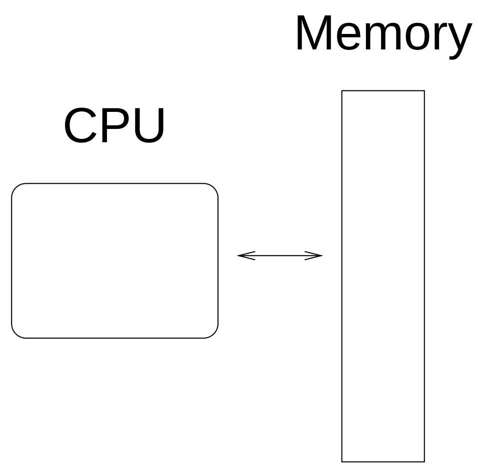
Analysis of algorithms

The standard model:



Analysis of algorithms

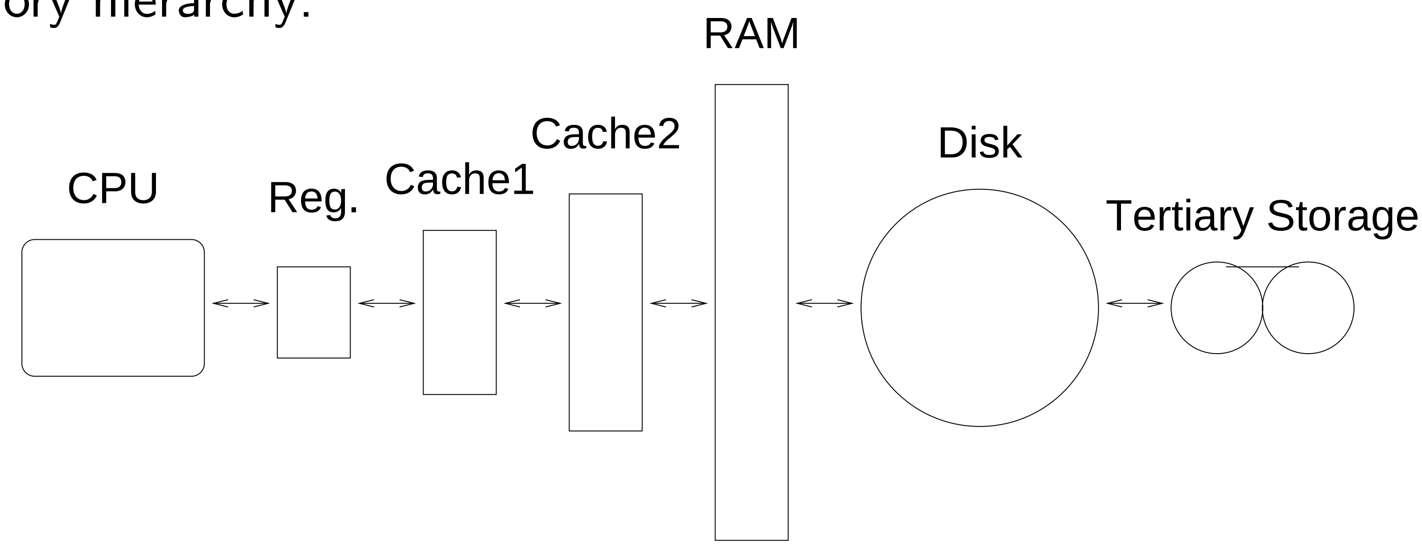
The standard model:



- ADD: 1 unit of time
- MULT: 1 unit of time
- BRANCH: 1 unit of time
- MEMACCESS: 1 unit of time

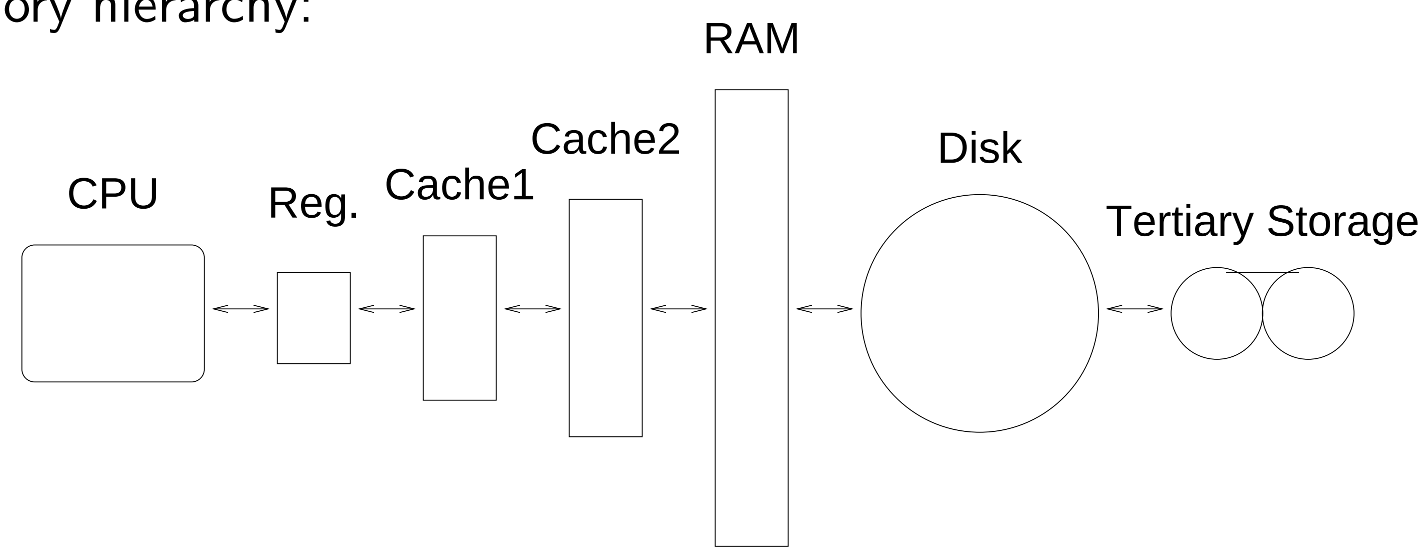
Reality

Memory hierarchy:



Reality

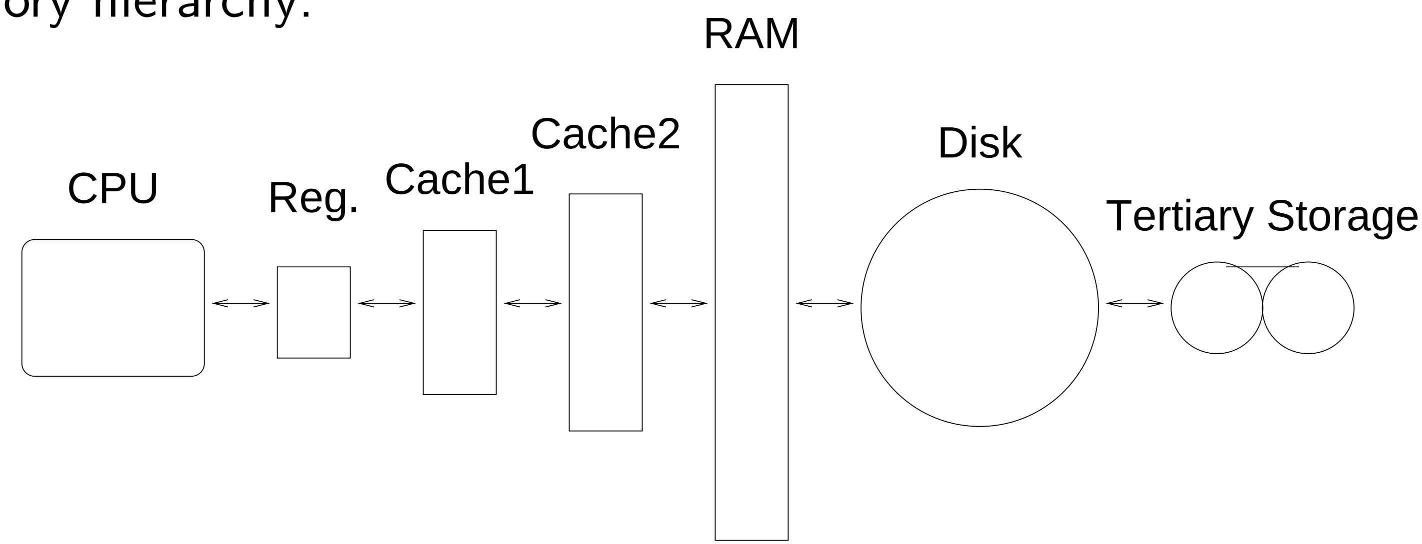
Memory hierarchy:



	Access time	*Volume*
Registers	1 cycle	1 Kb
Cache	5 cycles	512 Kb
RAM	50 cycles	512 Mb
Disk	20,000,000 cycles	80 Gb

Reality

Memory hierarchy:



	Access time	*Volume*
Registers	1 cycle	1 Kb
Cache	5 cycles	512 Kb
RAM	50 cycles	512 Mb
Disk	20,000,000 cycles	80 Gb

CPU speed improves faster than RAM access time and much faster than disk access time

Reality

Many real-life problems of **Gigabyte**, **Terabyte**, and even **Petabyte** size:

- Databases
  - weather
  - geology/geograpy
  - astrology
  - financial
  - WWW
  - phone companies
- Geographic Information Systems (maps).
- Computer graphics, animation.
- VLSI design.

I/O bottleneck

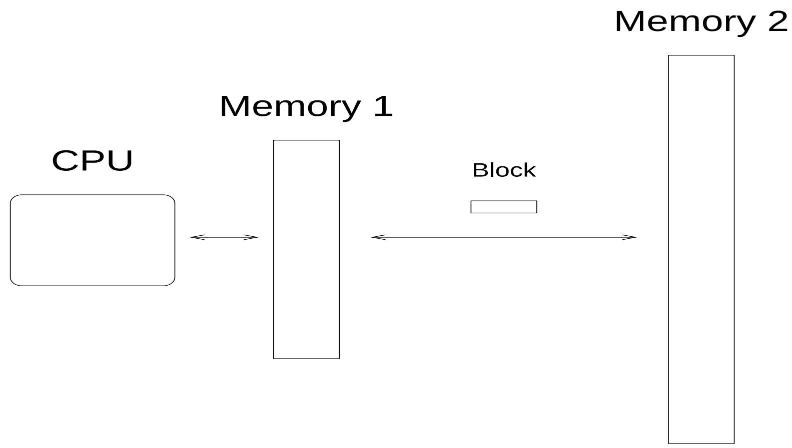
I/O is the bottleneck

$\Downarrow$

I/O should be optimized (not instruction count)

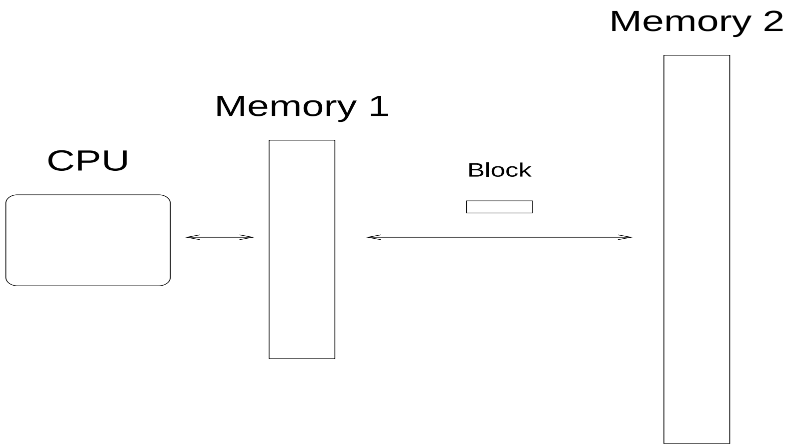
Analysis of algorithms

New **I/O-model:**



Analysis of algorithms

New **I/O-model:**

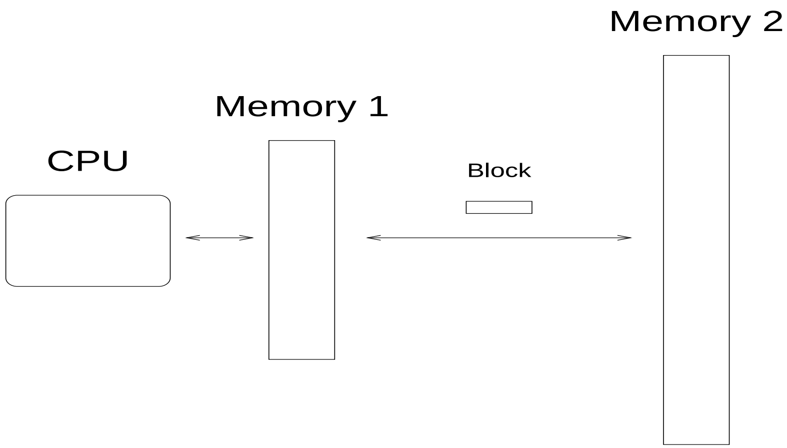


Parameters:

N	$=$	no. of elements in problem.
M	$=$	no. of elements that fit in RAM.
B	$=$	no. of elements in a block on disk.
D	$=$	no. of disks (copies of Memory 2)

Analysis of algorithms

New **I/O-model:**



Parameters:

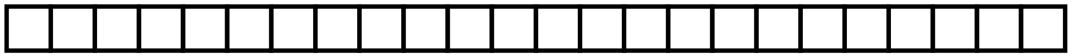
$$\begin{aligned} N &= \text{no. of elements in problem.} \\ M &= \text{no. of elements that fit in RAM.} \\ B &= \text{no. of elements in a block on disk.} \\ D &= \text{no. of disks (copies of Memory 2)} \end{aligned}$$

Cost: Number of I/O's (block transfers) between Memory 1 and Memory 2.

Generic Example

Consider two $O(n)$ algorithms:

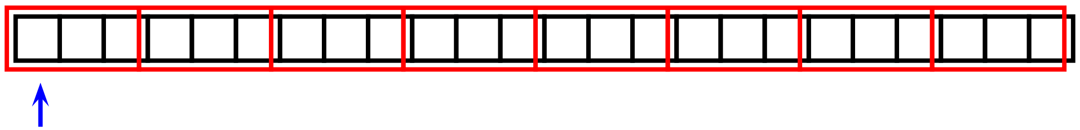
1. Memory accessed randomly $\Rightarrow$ page fault at each memory access.
2. Memory accessed sequentially $\Rightarrow$ page fault every B memory accesses.



Generic Example

Consider two $O(n)$ algorithms:

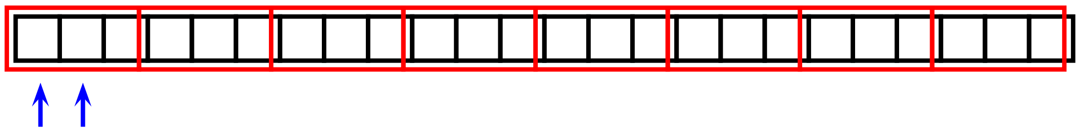
1. Memory accessed randomly $\Rightarrow$ page fault at each memory access.
2. Memory accessed sequentially $\Rightarrow$ page fault every B memory accesses.



Generic Example

Consider two $O(n)$ algorithms:

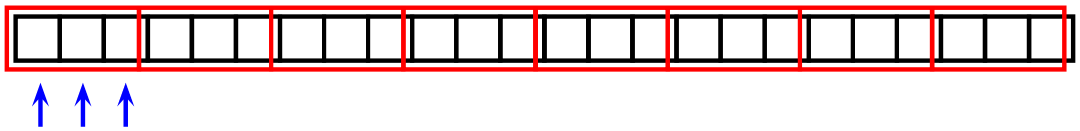
1. Memory accessed randomly $\Rightarrow$ page fault at each memory access.
2. Memory accessed sequentially $\Rightarrow$ page fault every B memory accesses.



Generic Example

Consider two $O(n)$ algorithms:

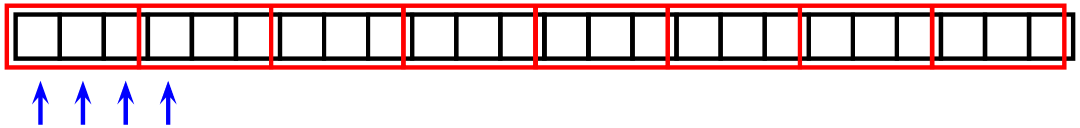
1. Memory accessed randomly $\Rightarrow$ page fault at each memory access.
2. Memory accessed sequentially $\Rightarrow$ page fault every B memory accesses.



Generic Example

Consider two $O(n)$ algorithms:

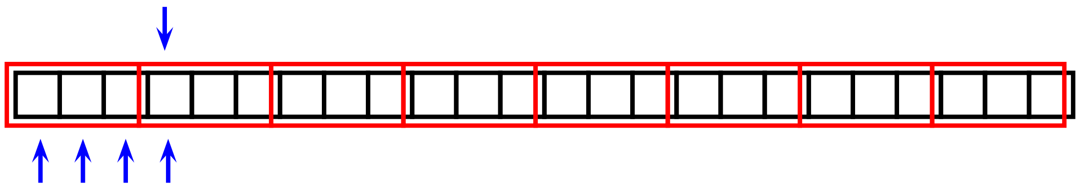
1. Memory accessed randomly $\Rightarrow$ page fault at each memory access.
2. Memory accessed sequentially $\Rightarrow$ page fault every B memory accesses.



Generic Example

Consider two $O(n)$ algorithms:

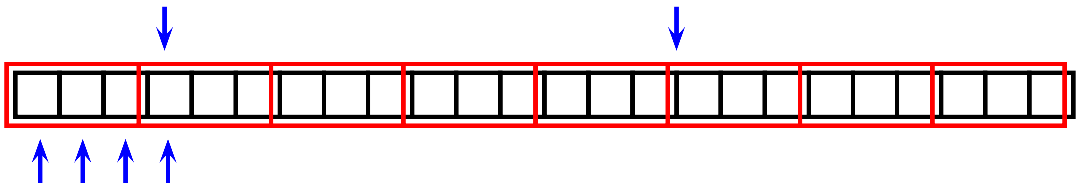
1. Memory accessed randomly $\Rightarrow$ page fault at each memory access.
2. Memory accessed sequentially $\Rightarrow$ page fault every B memory accesses.



Generic Example

Consider two $O(n)$ algorithms:

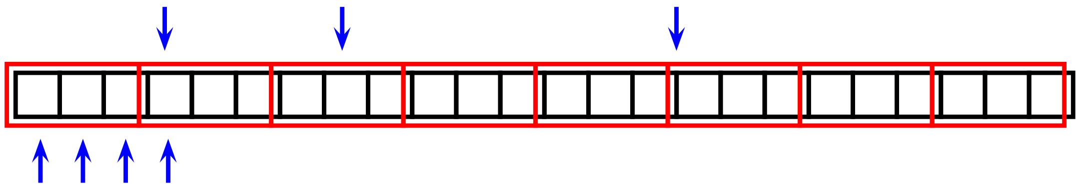
1. Memory accessed randomly $\Rightarrow$ page fault at each memory access.
2. Memory accessed sequentially $\Rightarrow$ page fault every B memory accesses.



Generic Example

Consider two $O(n)$ algorithms:

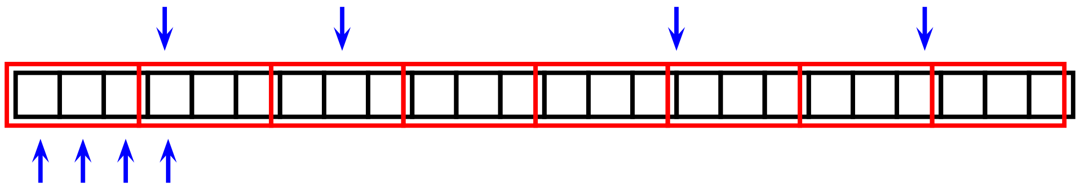
1. Memory accessed randomly $\Rightarrow$ page fault at each memory access.
2. Memory accessed sequentially $\Rightarrow$ page fault every B memory accesses.



Generic Example

Consider two $O(n)$ algorithms:

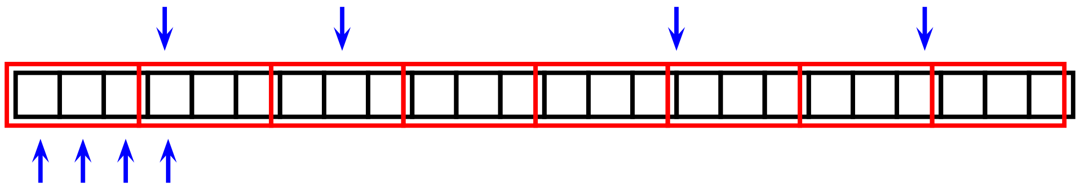
1. Memory accessed randomly $\Rightarrow$ page fault at each memory access.
2. Memory accessed sequentially $\Rightarrow$ page fault every B memory accesses.



Generic Example

Consider two $O(n)$ algorithms:

1. Memory accessed randomly $\Rightarrow$ page fault at each memory access.
2. Memory accessed sequentially $\Rightarrow$ page fault every B memory accesses.



$O(N)$ I/Os *vs.* $O(N/B)$ I/Os

Typically, $B \sim 10^3$.

Specific Examples

QuickSort $\sim$ sequential access

vs.

HeapSort $\sim$ random access

QuickSort:	$O(N \log_2(N/M)/B)$
HeapSort:	$O(N \log_2(N/M))$

Course

Contents (approximate):

- The I/O model(s).
- Algorithms, data structures, and lower bounds for basic problems:
  - Permuting
  - Sorting
  - Searching
- I/O efficient algorithms and data structures for problems from
  - computational geometry,
  - strings,
  - graphs.

Course

Contents (approximate):

- The I/O model(s).
- Algorithms, data structures, and lower bounds for basic problems:
  - Permuting
  - Sorting
  - Searching
- I/O efficient algorithms and data structures for problems from
  - computational geometry,
  - strings,
  - graphs.

Along the way I: Generic principles for designing I/O-efficient algorithms.

Along the way II: Hands-on experience via projects.