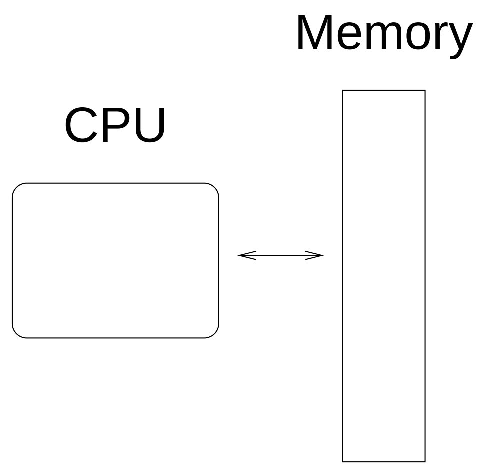
The I/O-Model

- Aggarwal and Vitter, *The Input/Output Complexity of Sorting and Related Problems*. Communications of the ACM, 31(9), p. 1116-1127, 1988.

Analysis of algorithms

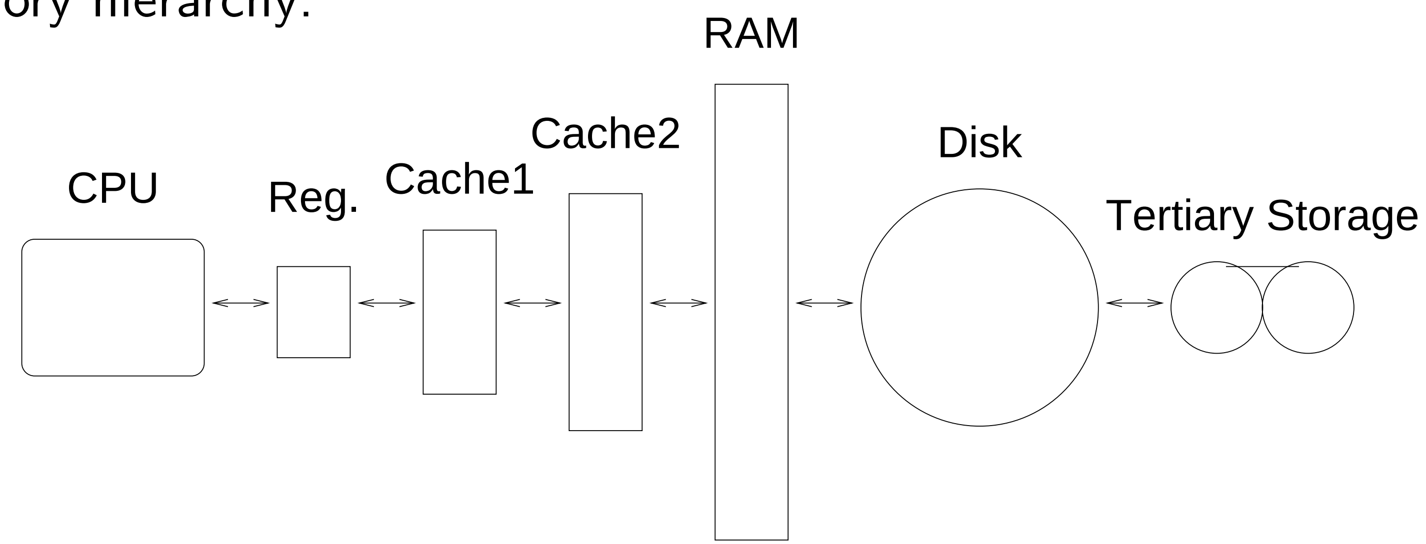
The standard model:



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

Reality

Memory hierarchy:



	Access time	*Volume*
Registers	1 cycle	1 Kb
Cache	5 cycles	512 Kb
RAM	50 cycles	256 Mb
Disk	2,000,000 cycles	80 Gb

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

I/O bottleneck

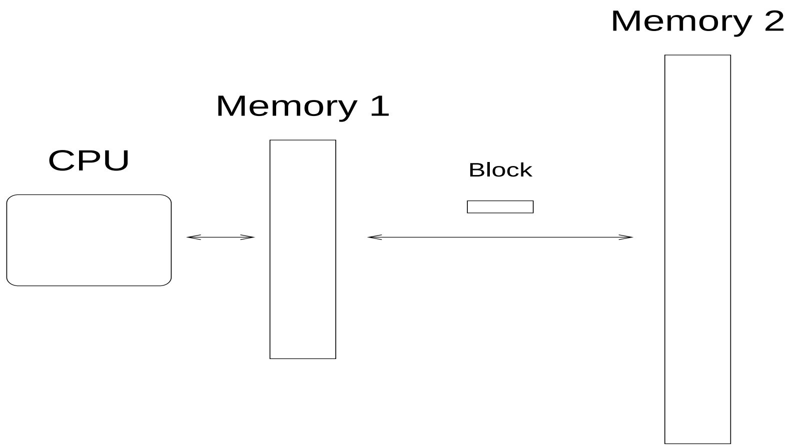
I/O is the bottleneck

$\Downarrow$

I/O should be optimized (not instruction count)

Analysis of algorithms

New **I/O-model:**



Parameters:

$$N \quad = \quad \text{no. of elements in problem.}$$

$$M \quad = \quad \text{no. of elements that fits in RAM.}$$

$$B \quad = \quad \text{no. of elements in a block on disk.}$$

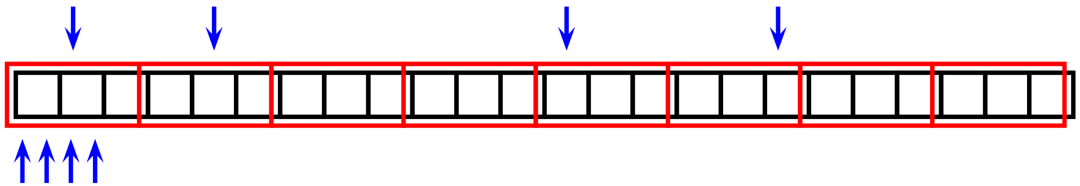
$$D \quad = \quad \text{no. of disks (copies of Memory 2)}$$

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.



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

Typically, $B \sim 10^3$.

Specific Examples

Two classic sorting algorithms:

QuickSort $\sim$ sequential access

vs.

HeapSort $\sim$ random access

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

Other examples:

- Hashing, radixsort (random access)
- Scanning, queues (sequential access)