External Memory I

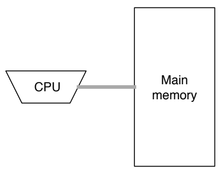
- Computational Models
- Scanning
- Sorting
- Searching

Philip Bille

External Memory I

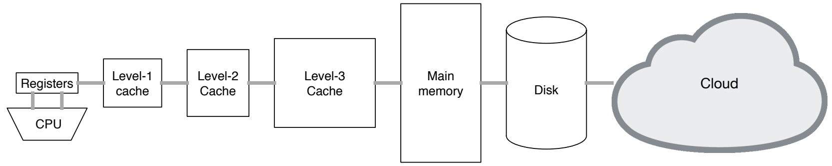
- Computational Models
- Scanning
- Sorting
- Searching

Computational Models



- (word) RAM Model
  - Infinite memory of w-bit memory cells
  - Instructions: Memory access, arithmetic operations, boolean operations, control-flow operations, etc.
- Complexity model.
  - Time = number of instructions.
  - Space = number of memory cells used.

Computational Models



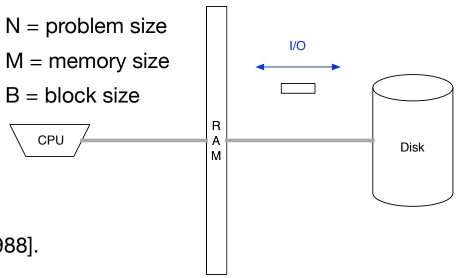
- iMac (late 2017)
  - CPU: 3.5 Ghz Core i5 (4 cores)
  - Registers: ?
  - L1 cache: ?
  - L2 cache: 256k per core
  - L3 cache: 6 MB shared
  - Memory: 8 GB
  - Disk: 1 Tb, (32 Gb SSD + 1Tb hard drive)
  - Instructions: Memory access, arithmetic operations, boolean operations, control-flow operations, etc.
- Complexity?

Computational Models

N = problem size

M = memory size

B = block size



- I/O model [Aggarwal and Vitter 1988].
  - Limited memory, Infinite disk
  - Instructions: Disk I/O operations, memory access, arithmetic operations, boolean operations, control-flow operations, etc.
- Complexity model.
  - I/Os = Number of disk I/Os
  - Computation is free (!)

External Memory I

- Computational Models
- Scanning
- Sorting
- Searching

Scanning

33	4	25	28	45	18	7	12	36	1	47	42	50	16	···

- Scanning. Given an array A of N values stored in N/B blocks and a key x, determine if x is in A.
- I/Os. O(N/B).

External Memory I

- Computational Models
- Scanning
- Sorting
- Searching

Sorting

33	4	25	28	45	18	7	12	36	1	47	42	50	16	31

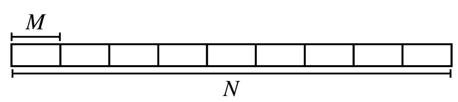
1	4	7	12	16	18	25	28	31	33	36	42	45	47	50

- Sorting. Given array A of N values (stored in N/B consecutive blocks), output the values in increasing order.

External Merge Sort

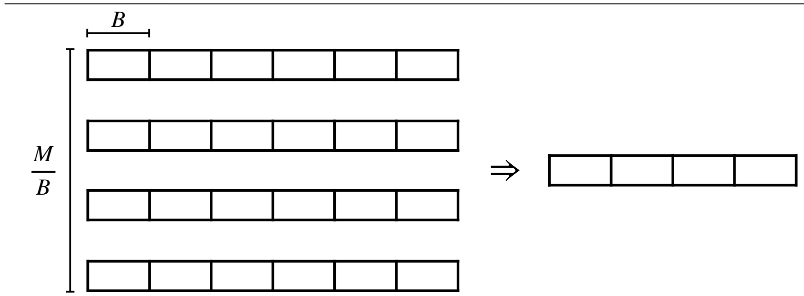
- Goal. Sorting in $O(N/B \log_{M/B} (N/B))$ I/Os.
- Solution in 3 steps.
  - Base case.
  - External multi-way merge.
  - External merge sort.

External Merge Sort



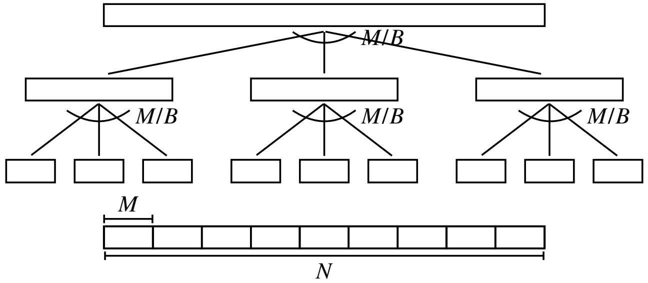
- Base case.
  - Partition N elements into N/M arrays of size M.
  - Load each into memory and sort.
- I/Os. O(N/B)

External Merge Sort



- Multiway merge algorithm.
  - N elements in M/B arrays.
  - Load M/B first blocks into memory and sort.
  - Output B smallest elements.
  - Load more blocks into memory if needed.
  - Repeat
- I/Os. O(N/B).

External Merge Sort



- Algorithm.
  - Partition N elements into N/M arrays of size M. Load each into memory and sort.
  - Apply M/B way external multiway merge until left with single sorted array.
- I/Os.
  - Sort N/M arrays: O(N/B) I/Os
  - Height of tree $O(\log_{M/B}(N/M))$
  - Cost per level: O(N/B) I/Os.

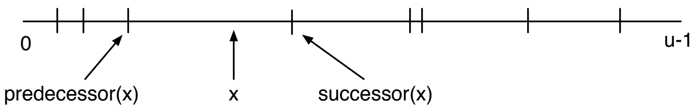
Total I/Os: $O\left(\frac{N}{B}\log_{M/B}\frac{N}{M}\right) = O\left(\frac{N}{B}\log_{M/B}\frac{N}{B}\right)$

External Memory I

- Computational Models
- Scanning
- Sorting
- Searching

Searching

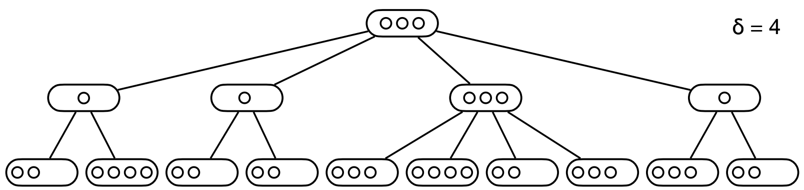
- Searching. Maintain a set $S \subseteq U = \{0, ..., u-1\}$ supporting
  - member(x): determine if $x \in S$
  - predecessor(x): return largest element in $S \leq x$.
  - successor(x): return smallest element in $S \geq x$.
  - insert(x): set $S = S \cup \{x\}$
  - delete(x): set $S = S - \{x\}$



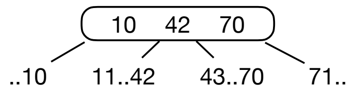
Searching

- Applications.
  - Relational data bases.
  - File systems.

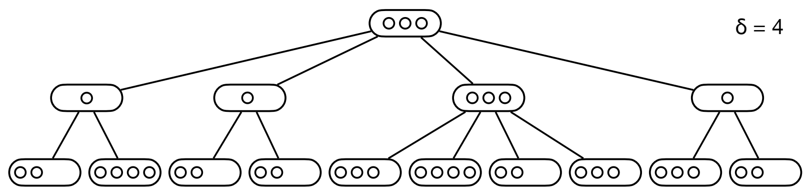
B-tree



- B-tree of order $\delta = \Theta(B)$ with N keys.
  - Keys in leaves. Routing elements in internal nodes.
  - Degree between $\delta/2$ and δ.
  - Root degree between 2 and δ.
  - Leaves store between $\delta/2$ and δ keys.
  - All leaves have the same depth.
- Height. $\Theta(\log_\delta (N/B)) = \Theta(\log_B N)$

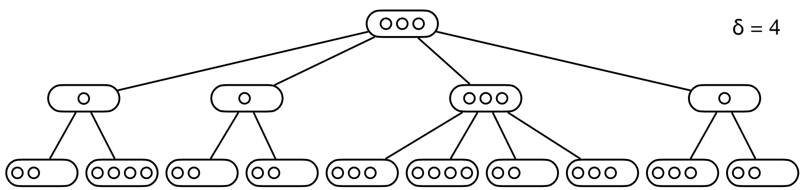


B-tree

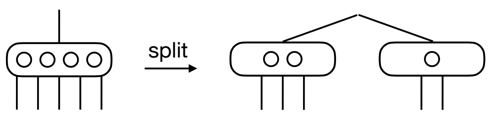


- Searching.
  - Find leaf using routing elements.
- I/Os. $O(\log_B N)$.

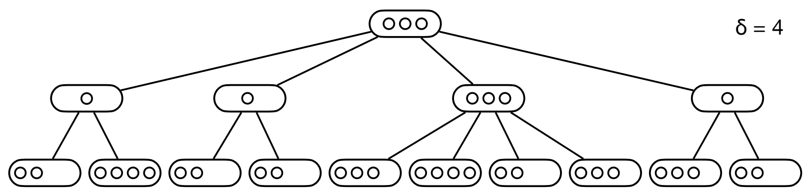
B-tree



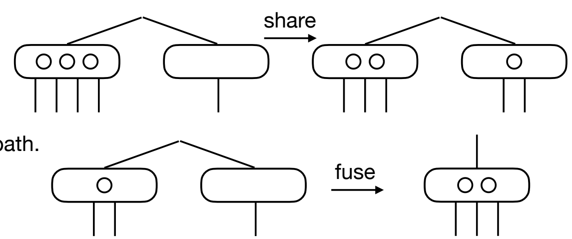
- Insertion.
  - Find leaf.
  - Insert key.
  - Split nodes on path.
- I/Os. $O(\log_B N)$.



B-tree



- Deletion.
  - Find leaf.
  - Delete key.
  - Share or fuse nodes on path.
- I/Os. $O(\log_B N)$.



Basic Bounds

	Internal	External
Scanning	O(N)	$scan(N) = O(N/B)$
Sorting	O(Nlog N)	$sort(N) = O((N/B)\log_{M/B} (N/B))$
Searching	O(log N)	$search(N) = O(\log_B(N))$

External Memory I

- Computational Models
- Scanning
- Sorting
- Searching