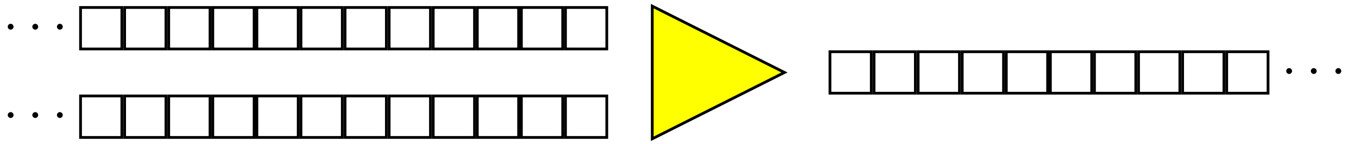
I/O Efficient Sorting

Upper and Lower bounds

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

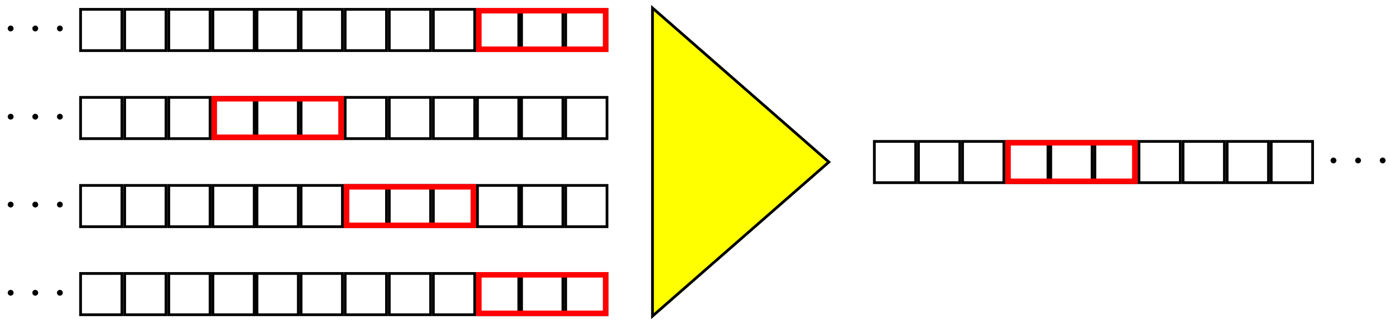
Standard MergeSort

Merge of two sorted sequences $\sim$ sequential access



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

Multiway Merge



- For I/O-efficient k-way merge of sorted lists we need:

$$M \geq B(k+1) \Leftrightarrow M/B - 1 \geq k$$

- Number of I/Os: $2N/B$.

Multiway MergeSort

- N/M times sort M elements internally $\Rightarrow$ N/M sorted *runs* of length M.
- Merge k runs at at time, to produce $(N/M)/k$ sorted runs of length kM.
- Repeat: Merge k runs at at time, to produce $(N/M)/k^2$ sorted runs of length k^2M, ...

At most $\log_k N/M$ phases, each using $2N/B$ I/Os.

Best k: M/B-1.

$\boxed{O(N/B \log_{M/B}(N/M)) \text{ I/Os}}$

Multiway MergeSort

$$1 + \log_{M/B}(x) = \log_{M/B}(M/B) + \log_{M/B}(x) = \log_{M/B}(x \cdot M/B)$$

$$\Downarrow$$

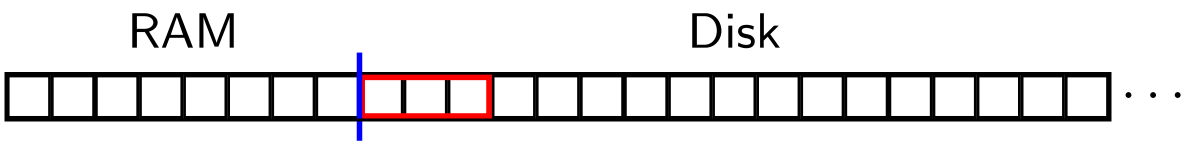
$$O(N/B \log_{M/B}(N/M)) = O(N/B \log_{M/B}(N/B))$$

Defining $\boxed{\mathsf{n} = \mathsf{N}/\mathsf{B}}$ and $\boxed{\mathsf{m} = \mathsf{M}/\mathsf{B}}$ we get

$$\boxed{\text{Multiway MergeSort:} \quad O(n \log_m(n))}$$

Sorting Lower Bound

Model of memory:



- Comparison based model: elements may be compared in internal memory. May be moved, copied, destroyed. Nothing else.
- Assume $M \geq 2B$.
- May assume I/Os are block-aligned, and that at start, input contiguous in lowest positions on disk.
- Adversary argument: adversary gives order of elements in internal memory (chooses freely among consistent answers).
- Given an execution of a sorting algorithm: S_t = number of permutations consistent with knowledge of order after t I/Os.

Adversary Strategy

After an I/O, adversary must give new answer, i.e. must give order of elements currently in RAM.

If number of possible (i.e. consistent with current knowledge) orders is X, then there exist answer such that

$$S_{t+1} \geq S_t / X.$$

This is because any single answer induces a subset of the S_t currently possible permutations (consisting of the permutations consistent with this answer), and the X such subsets clearly form a partition of the S_t permutations. If no subset has size S_t/X, the subsets cannot add up to S_t permutations.

Adversary chooses answer fulfilling the inequality above.

Possible X's

Type of I/O	Read untouched block	Read touched block	Write
X	$\binom{M}{B}B!$	$\binom{M}{B}$	1

Note: at most N/B untouched blocks read.

From $S_0 = N!$ and $S_{t+1} \geq S_t/X$ we get

$$S_t \geq \frac{N!}{\binom{M}{B}^t (B!)^{N/B}}$$

Sorting algorithm cannot stop before $S_t = 1$. Thus,

$$1 \geq \frac{N!}{\binom{M}{B}^t (B!)^{N/B}}$$

for any correct algorithm making t I/Os.

Lower Bound Computation

$$1 \geq \frac{N!}{\binom{M}{B}^t (B!)^{N/B}}$$

$$t \log \binom{M}{B} + (N/B) \log(B!) \geq \log(N!)$$

$$3tB \log(M/B) + N \log B \geq N(\log N - 1/\ln 2)$$

$$3t \geq \frac{N(\log N - 1/\ln 2 - \log B)}{B \log(M/B)}$$

$$t = \Omega(N/B \log_{M/B}(N/B))$$

Lemma was used:

a) $\log(x!) \geq x(\log x - 1/\ln 2)$

b) $\log(x!) \leq x \log x$

c) $\log \binom{x}{y} \leq 3y \log(x/y)$ when $x \geq 2y$

Proof of Lemma

Lemma:

a) $\log(x!) \geq x(\log x - 1/\ln 2)$

b) $\log(x!) \leq x \log x$

c) $\log \binom{x}{y} \leq 3y \log(x/y)$ when $x \geq 2y$

$$\boxed{\text{Stirlings formula: } n! = \sqrt{2\pi n} \cdot (n/e)^n \cdot (1 + O(1/12n))}$$

Proof (using Stirling):

a) $\log(x!) \geq \log(\sqrt{2\pi x}) + x(\log x - 1/\ln 2) + o(1)$

b) $\log(x!) \leq \log(x^x) = x \log x$

c) $\log \binom{x}{y} \leq \log(\frac{x^y}{(y/e)^y}) = y(\log(x/y) + \log(e))$

$\leq 3y \log(x/y)$ when $x \geq 2y$

The I/O-Complexity of Sorting

Defining

$$\mathsf{n} = \mathsf{N}/\mathsf{B}$$
$$\mathsf{m} = \mathsf{M}/\mathsf{B}$$
$$N/B \log_{M/B}(N/B) = \mathrm{sort}(N)$$

we have proven

I/O cost of sorting:

$$\boxed{\begin{aligned} &\Theta(N/B \log_{M/B}(N/B)) \\ &= \Theta(n \log_m(n)) \\ &= \Theta(\mathrm{sort}(N)) \end{aligned}}$$