Sorting and Permuting on Sequential and Parallel Devices.

Notation:
- \( N \): # records to sort
- \( M \): # records that fit in internal memory
- \( B \): # records in a transfer block
- \( P \): # blocks transferred concurrently.

DAM Model (Aggarwal, Vitter 88)

Cost of block transfer = 1
- \( P \) blocks transferred concurrently, explicitly managed.
- Goal: minimize # memory transfers.

Compare With Cache-Oblivious Model:
- \( P = 1 \), other parameters unknown
- System manages memory
- Shoulders burden of memory management
Results

Theorem: The average-case and worst-case cost to sort $N$ records is

$$\Theta\left(\frac{N}{PB} \frac{\log(1+N/B)}{\log(1+M/B)}\right).$$

Theorem: The average-case and worst-case cost to permute $N$ records is

$$\Theta\left(\min\left\{\frac{N}{P}, \frac{N}{PB} \frac{\log(1+N/B)}{\log(1+M/B)}\right\}\right).$$
Parallel Mergesort for $P = 1$.

Total # Mem transfers:

$$T(N) = \frac{N}{B} + \frac{M}{B} T\left(\frac{N}{M/B}\right)$$

$$T(B) = 1$$

Solution:

$$\frac{N}{B} \cdot \frac{M}{B} = \frac{N}{B}$$

**# levels**

$$\text{height} = \log_{M/B} N - \log_{M/B} B = \log_{M/B} \frac{N}{B}.$$ 

Cost per level = $N/B$

$$T(N) = \frac{N}{B} \log_{M/B} \frac{N}{B}.$$
Note: For simplicity I'm removing "t+".
Note: The parallel merge sort doesn't immediately work for a non-constant \( P \), but can be made to work...
Permuting for $P = 1$:

2 choices:

1) sort $\Rightarrow$ same bounds as before

2) put each element directly in its destination $\Rightarrow$ 1 memory transfer per element
Reminder: Sorry LB.

N! permutations are consistent with info.
each company rules out at most half.

need $\log(n!) \geq \Omega(n \log n)$
   computer.
Lower Bound on Sorting for $P=1$

Thus: External sorting requires $O(N/B \log_{N/B} N/B)$ I/O's in comparison - $40$ model (comparison is only allowed up in internal memory).

Proof: Information-theoretic argument.

At beginning of computation, $N!$ possibilities available for correct ordering based on available information (none).

After each input we learn through comparisons, narrowing down possible number of orders.

Show that need $t = O(N/B \log_{N/B} N/B)$ inputs to learn enough that only one consistent order left.

(narrow down possibilities)

Two cases:

Case 1: We know order of elements in internal memory but not order of block $B$ being input.

[Diagram showing flow of information and memory blocks]

know order

do not know order
If $S$ denotes the number of possible orderings of $N$ elements before input, then one of $\binom{M}{B}$ orderings within memory, such that the remaining orderings still consistent is

$$\geq \frac{S}{\binom{M}{B}}.$$
Claim: # times we can read a block of B records that have not been together in memory: $n/B$.

Lemma: After $t$ input operations, at least

$$\frac{N!}{(M)^t (B!)^{n/B}}$$

orderings are consistent with available information.

Goal: Narrow down possible orderings to 1:

\[ \#1/0's, t, \text{ must satisfy} \]

$$\frac{N!}{(M)^t (B!)^{n/B}} \leq 1.$$ 

Useful formulae:

- $\log(x!) = \Theta(x \log x)$ (stirling)  \[ n! \approx \sqrt{2\pi n} \left(\frac{n}{e}\right)^n \]
- $\log\left(\frac{M}{B}\right) = \Theta(B \log \frac{M}{B})$

Proof:\[ \left(\frac{M}{B}\right)^B \leq (M)^B \leq \left(\frac{eM}{B}\right)^B. \]
Solve for \( t \):

\[
\frac{N!}{(\frac{N}{B})^t (B!)^{N/B}} \leq 1
\]

\[
(\frac{N}{B})^t (B!)^{N/B} \geq N!
\]

\[
t \log(\frac{N}{B}) + \frac{N}{B} \log(B!) \geq \log(N!)
\]

\[
t B \log(\frac{N}{B}) + \frac{N}{B} B \log B \geq \Omega(N \log N)
\]

\[
t B \log(\frac{N}{B}) \geq \Omega(N \log \frac{N}{B})
\]

\[
t \geq \Omega\left(\frac{N}{B} \frac{\log(N/B)}{\log(M/B)}\right)
\]

\[
t = \Omega\left(\frac{N}{B} \log_{M/B} (N/B)\right).
\]

Notation from I/O efficient algs:

\[m = \frac{M}{B}\]
\[n = \frac{N}{B}.
\]

\[\Rightarrow \Omega(n \log m n).
\]
Lower Bound on External Permuting

Thm: Rearranging $N$ elements according to a given permutation requires $\Omega \left( \min \left( N, \frac{N}{b} \log \frac{N}{b} \frac{N}{b} \right) \right)$ I/O operations.

Pf:

Model Assumptions:

1) External Memory comprised of $N$ blocks of size $B$ (size $NB$). An I/O moves a single block.

2) I/O's are simple. Transfer of elements only allowed operation—no new elements or duplicates.

3) Main memory + disk viewed as big extended array.

\[ \begin{array}{c}
\hline
1 & NB & 2 & \cdots & M/B & 1 \\
\vdots
\end{array} \]
Def: Permutation = order of elements in extended array (ignore spaces)

Claim: Assumptions ⇒ exactly one permutation at all times.

Idea: Bound # permutations for t = 10s.

Initially: 1 permutation

Require: N! permutations
Input:
- choice of $N$ blocks to input
- after loading one block, $B_B$ can put $\leq B$ elements between $\leq M-B$ locations in memory.

⇒

2 cases:
1) Virgin block, new
   \[ N(B!)(M)! \] (#permutations already)

2) already-read block:
   \[ N(M)! \] (#perms already)

Claim: case (1) can happen $\leq N/B$ times.
Output:

$N$ target blocks to output. $B$ elements to pick.

Claim: After $t \leq c_0 s \leq (\beta!)^{N/B} (N/\beta)^t$

perms attainable.
\[(B!)^{n/B} \left[ \left( \frac{n}{B} \right)^N \right]^t \geq N! \]

\[\frac{N}{B} \log (B!) + t \left[ \log \left( \frac{n}{B} \right) + \log N \right] \geq \log (N!) \]

\[N \log B + t \left[ B \log \left( \frac{n}{B} \right) + \log N \right] \geq \frac{\Omega}{B} (N \log N) \]

\[t \geq \frac{\frac{N \log \left( \frac{n}{B} \right)}{B \log \left( \frac{n}{B} \right) + \log N}}{2} \]
2 cases:

Case 1: \( \log N \leq B \log N/B \)

\[ \Rightarrow t \geq \Omega \left( \frac{N}{B} \log_{N/B} N/B \right) \]

Case 2: \( \log N > B \log (N/B) \Rightarrow B \ll N \)

\[
\frac{N \log N/B}{2 \log N} = \frac{N \log N - N \log B}{2 \log N}
\]

\[ = \frac{1}{2} \left[ N - N \frac{\log(B)}{\log(N)} \right] \]

\[ = \frac{1}{2} \left( N - \frac{1}{2} N \right) \]

\[ = \Omega(N) \]
Model Justification:

Nonexample -> Simple: remove all you not present in final paper.

[Note]

Size assumption -> no reason to have blocks that are empty.
Informal: Cache-Oblivious Sorting

Cost of merge sort:

\[ T(N) = 2T(N/2) + N/B \]

\[ T(B) = 1 \]

\[ T(N) = \Theta\left(\frac{N}{B} \log_2 N\right). \]

Need multiway merge! But how big??