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

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 Merge sort 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:

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

$$\text{ cost per level } = \frac{N}{B}$$

$$T(N) = \frac{N}{B} \log_{M/B} \frac{N}{B}.$$
Note: For simplicity I'm removing "$1^+".

Note: The parallel mergesort 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: Sorty LB.

$N!$ permutations, consult with info.
each company rules out at most half.

need $\log (N!) = \Omega (n \log n)$

Again.
Lower Bound on Sorting for $P=1$

**Thm:** External sorting requires $\Omega(\frac{N}{B} \log_{\frac{N}{B}} \frac{N}{B})$ I/O's in comparison - 1/0 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 $t = \frac{1}{2} (\frac{N}{B} \log_{\frac{N}{B}} \frac{N}{B})$ 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 the flow of data and knowing the order vs. not knowing the order]
\# possible orderings in memory

\[ \leq (B!) \left( \frac{M-B+B}{(M-B)!B!} \right) \]
\hfill order in block \# interleavings (\#s and \#s)
\hfill \leq (B!) \left( \binom{M}{B} \right).

If \( S \) denotes \# possible orderings of \( N \) elments before input,

\exists one of \( (B!) \binom{M}{B} \) orderings within memory, such that

\# remaining orderings still consistent is

\[ \geq \frac{S}{(B!) \binom{M}{B}}. \]

After \( t \) inputs of case 1: \# remaining orderings

\[ \geq \frac{S}{(B! (\binom{M}{B})^t}. \]

**CASE 2:** Order of records in both main memory and input block already known (e.g. input block was output previously).

\# possible orderings in memory

\[ \leq \binom{M}{B}. \]
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

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

orderings are consistent with available information.

Goal: Narrow down possible orderings to 1:

\( \# \text{1/0's, } t \), must satisfy

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

Useful formulae:

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

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

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

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

$$t \log\left(\frac{M}{B}\right) + \frac{N}{B} \log(B!) \geq \log(N!)$$

$$t B \log\left(\frac{M}{B}\right) + \frac{N}{B} B \log B \geq \Omega(N \log N)$$

$$t B \log\left(\frac{M}{B}\right) \geq \Omega(N \log \frac{N}{B})$$

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

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

---

Notation from I/O efficient alg $\mathbb{E}$:

$$m = M/B$$

$$n = N/B.$$ 

$$\Rightarrow \quad \Omega(n \log mn).$$
Lower Bound on External Permuting

Thm: Rearranging $N$ elements according to a given permutation requires $\Omega(\min(N, \frac{N}{B} \log \frac{N}{B}))$ 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.

\[
\begin{array}{ccccccc}
\text{NB} & \text{H} & \text{M/B} & \text{I}
\end{array}
\]

\[
\begin{array}{ccccccc}
\text{I} - 8 - 1
\end{array}
\]
Def: Permutation = order of elements in extended array (ignore spaces)

Claim: Assumptions $\Rightarrow$ exactly one permutation at all

Idea: Bound # permutations for $t$ clos.

Initially: $t$ permutation

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

2 cases:
1) Virgin block, new
   $N(B!)(M/B)^x \cdot (# \text{permutations already})$

2) Already-read block: $N(M/B)^x \cdot (# \text{perms already})$

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

N target blocks to output. B elects to pick.

Claim: After $t$ vos $\leq (B!)^{N/B} (N \binom{N}{B})^t$

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

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

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

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

Case 1: \( \log N \leq B \log \frac{N}{B} \)

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

Case 2: \( \log N > B \log \left( \frac{N}{B} \right) \Rightarrow B \ll \sqrt{N} \)

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

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

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

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

Nonexample → Simple: remove all 1/0s not present in final perm.

Example

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??

[Diagram of multiway merge process]