Lower bounds for comparison based sorting

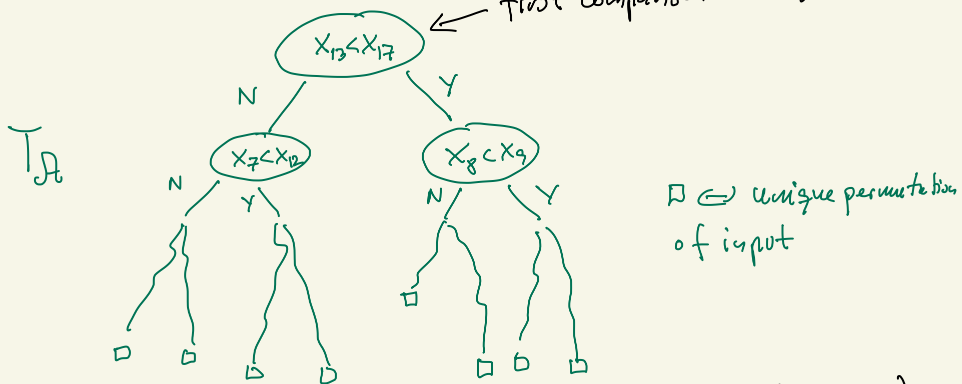
Based on Baase Section 2.4 and JBJ notes Section 7

Known from e.g. DM507: Mergesort uses $\Theta(n \log n)$ comparisons to sort n numbers

Decision trees

Every deterministic comparison based (sorting) algorithm A can be associated with a so-called decision tree:

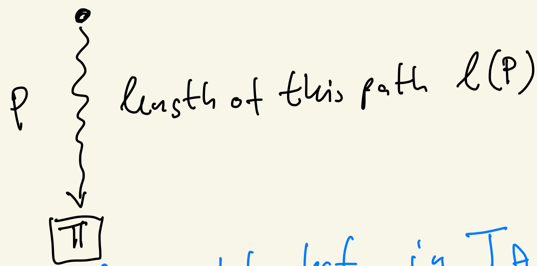
first comparison made by A



- A must be able to determine any permutation of $\{1, 2, \ldots, n\}$ as output (a leaf in T_A) $\Rightarrow$ T_A has at least $n!$ leaves
- T_A is a binary tree so at most 2^h leaves when T_A has height h. Thus $2^h \geq n! \Rightarrow h \geq \log n! \sim n \log n - cn$
- Each path P from root of T_A to a leaf corresponds to comparisons made by A to sort some input and # comparisons = length of P so A uses $\geq n \log n - cn$ comp. on some input

Lower bound for average # comparisons via decision trees

\# comparisons for a fixed permutation π:

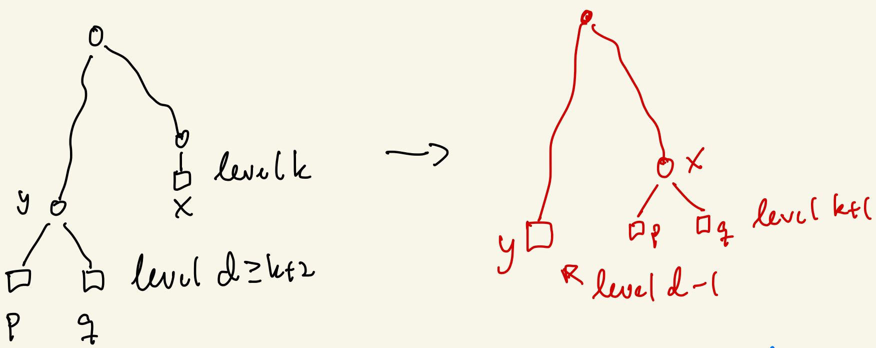


$\mathcal{P}$ set of all paths from root to leaf in T_A

<u>Def</u> $epl = \sum_{P \in \mathcal{P}} \text{length}(P)$ epl ~ external path length

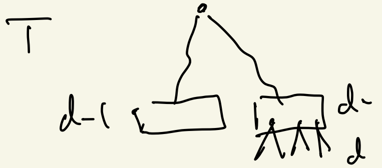
<u>Lemma</u> 2.8 epl is minimized when T is almost balanced

P:



Change in epl: $(2(k+1)-k) + (d-1-2d) = k+1-d < 0$ as $d \geq k+2$

Conclusion epl is minimum when T

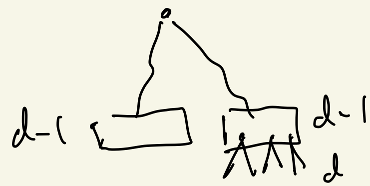


Lemma 2.9 min epl in binary tree with ℓ leaves is

$$\ell\lfloor\log \ell\rfloor + 2\left(\ell - 2^{\lfloor\log \ell\rfloor}\right)$$

P: clear if $\ell = 2^k$ for some k

assume ℓ is not a power of 2



$d = \lceil\log_2 \ell\rceil \qquad d-1 = \lfloor\log_2 \ell\rfloor$

leaves at level d : $2(\ell - 2^{d-1})$

so $epl = \ell(d-1) + 2(\ell - 2^{d-1})$

$= \ell\lfloor\log \ell\rfloor + 2\left(\ell - 2^{\lfloor\log \ell\rfloor}\right)$

Lemma 2.10 The average path length in a binary tree with ℓ leaves is at least $\lfloor\log \ell\rfloor$

P: The min average path length is

$$\frac{\ell\lfloor\log \ell\rfloor + 2\left(\ell - 2^{\lfloor\log \ell\rfloor}\right)}{\ell} = \lfloor\log \ell\rfloor + \varepsilon$$

$0 \leq \varepsilon < 1$ as $\ell - 2^{\lfloor\log \ell\rfloor} < \ell/2$

Thm 2.11 The average # comparisons done by any algorithm to sort n numbers by comparisons is at least $\lfloor\log n!\rfloor = \Omega(n\log n)$

Adversary lower bounds for sorting

Adversary one (very powerfull)

- always maintain a list $\mathcal{S}$ with all permutations of input consistent with answers given so far

  $\mathcal{S}_i$ = permutations alive (still possible) after i'th comparison

  $|\mathcal{S}_0| = n!$

- When answering $x<y$? in step i answer such that $\frac{|\mathcal{S}_i|}{|\mathcal{S}_{i-1}|} \geq \frac{1}{2}$

- It will take at least $\log_2 n!$ comparisons before input is sorted $(|\mathcal{S}_k| = 1)$

  so A must do at least $n\log n - cn$ comparisons on some input

Problem: argument is correct but in each step (comparison) the adversary has to construct $\mathcal{S}_i$ from $\mathcal{S}_{i-1}$ which takes time proportional to $|\mathcal{S}_{i-1}|$

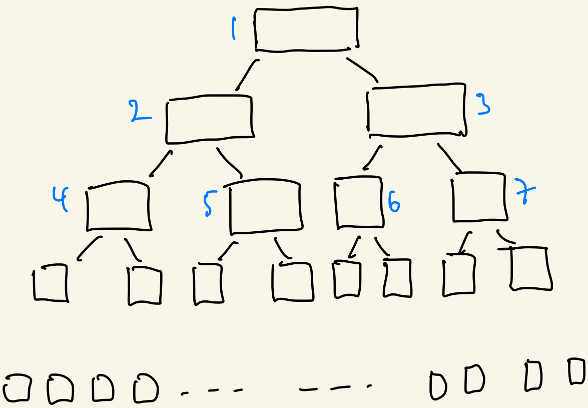
Can we make a more efficient adversary?

want the answer to $x<y$? fast

while maintaining consistent answers

Adversary 2 (much less powerfull) assume $n=2^k$

Maintain a tree structure

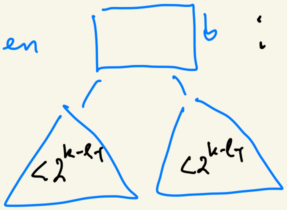


level	size of each bag	# bags at level
0	$n=2^k$	1
1	$n/2=2^{k-1}$	2
2	$n/4=2^{k-2}$	4
3	$n/8=2^{k-3}$	8
$\vdots$		
ℓ	$n/2^\ell=2^{k-\ell}$	2^ℓ

- Initially the root bag contains all n elements
- Subtree $T(b)$ rooted at bag b at level ℓ contains at most $\frac{n}{2^\ell}=2^{k-\ell}$ elements and in total there are n elements (so MANY bags are empty)
- While the adversary answer queries by A the elements move down through T until they are all in the n leaf bags of size 1 at level k

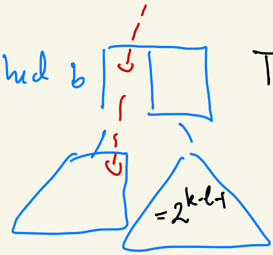
States of a bag b at level ℓ

b is open



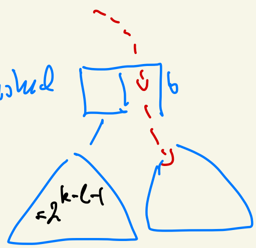
: Each of the left and right subtrees of b have less than $2^{k-\ell-1}$ elements

b is left flushed



The left subtree of b has $2^{k-\ell-1}$ elements

b is right flushed



The right subtree of b has $2^{k-\ell-1}$ elements

Idea! adversary only moves an element x to a lower bag if

1. x is in a new comparison or
2. the box that x enters is flushed

$\Rightarrow$ adversary can force $\frac{n}{2}\log_2 n$ comparisons

How to perform the strategy?

- MOVE(u)

  $b(u)$ current bag of u

  1. If $b(u)$ is open Return
  2. if $b(u)$ is left flushed

     $b(u) \leftarrow b^L(u)$

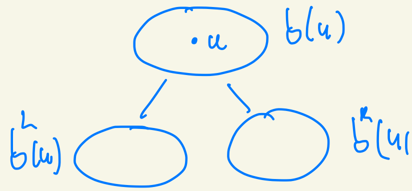
     MOVE(u)

     Return
  3. If $b(u)$ is right flushed

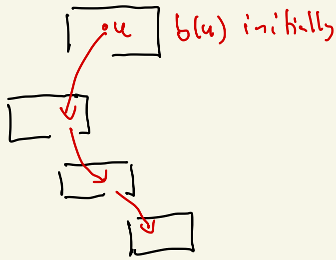
     $b(u) \leftarrow b^R(u)$

     MOVE(u)

     Return



u may move down several levels when we call MOVE(u)



CHECK(b) b open bag at level ℓ

If $|T[b^R]| = 2^{k-\ell-1}$ (full) then

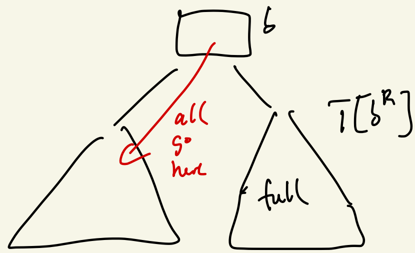
- Mark b as left-flushed
- $\forall u \in b$ MOVE(u)

Return

If $|T[b^L]| = 2^{k-\ell-1}$ then

- mark b as right-flushed
- $\forall u \in b$ MOVE(u)

Return



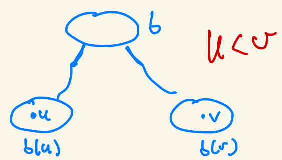
Strategy: while answering queries by $\mathcal{A}$

Move 2, 1 or zero elements down in T with one exception:

After each move of an element from b we call CHECK(b) which could move up to $2^{k-\ell-1}$ elements from b and leave it empty

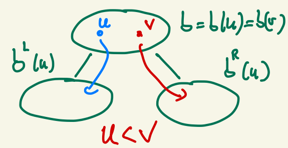
Answer to query $u < v$?

1. let b = least common ancestor of $b(u), b(v)$ in T



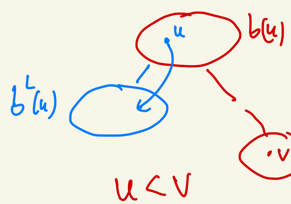
2. If $b \neq b(u), b(v)$
   If $u \in T[b^L]$ answer $u < v$
   Else answer $u > v$

3 Else (wlog $b = b(u)$, else rename u, v)

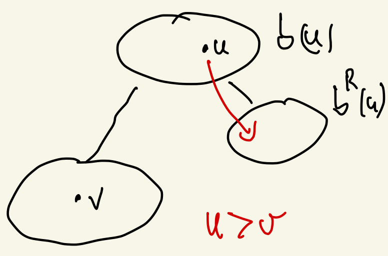


(a) if $b(u) = b(v)$ then
   • answer $u < v$ and $b(u) \leftarrow b^L(u)$; $b(v) \leftarrow b^R(u)$
   MOVE(u); MOVE(v); CHECK(b)

(b) Else if $b(v)$ in $T[b^R(u)]$ then
   answer $u < v$; $b(u) \leftarrow b^L(u)$
   MOVE(u); CHECK(b)



Else
   answer $u > v$;
   $b(u) \leftarrow b^R(u)$
   MOVE(u)
   CHECK(b)



Theorem 7.1 Using the strategy above the adversary can force any comparison based sorting algorithm $\mathcal{A}$ to make $\Omega(n \log n)$ comparisons

P: • Initially all elements are at level 0 in the root bag
• At termination all elements are in their own leaf bag at level $k = \log n$
• Between these events each bag becomes non-empty and then empty again at least once

• Claim: we can associate at least $2^{k-\ell-1}$ comparisons made by $\mathcal{A}$ privately to each bag at level ℓ

P: Let b be any bag at level ℓ
associate all comparisons $u < v$ when $|b \cap \{u,v\}| \geq 1$ and $u,v \in T(b)$
There are at least $2^{k-\ell-1}$ such as b is not flushed before we have made at least $2^{k-\ell-1}$ comparisons of the type above
(these are the only comparisons that move elements out of b)

Conclusion at level ℓ we can associate at least

$$2^{\ell} \cdot 2^{\ell-k-1} = 2^{k-1} = n/2 \text{ comparisons}$$

There are $\log n$ levels so in total adversary forces $\geq \frac{n}{2} \log_2 n$ comparisons

D.

Corollary 7.2 The adversary's strategy can be performed in time $O(n \log n)$

P: read yourselves

Idea: we don't need to construct T

- represent the 2^ℓ bags at level ℓ by

  $b_{\ell,0}, b_{\ell,1} \; \ldots \; b_{\ell,2^\ell-1}$

- For each element u maintain

  $\ell(u)$ = level of bag containing u

  $b(u)$ with $0 \le b(u) \le 2^{\ell(u)}-1$ is the index of the bag at level $\ell(u)$ which contains u.