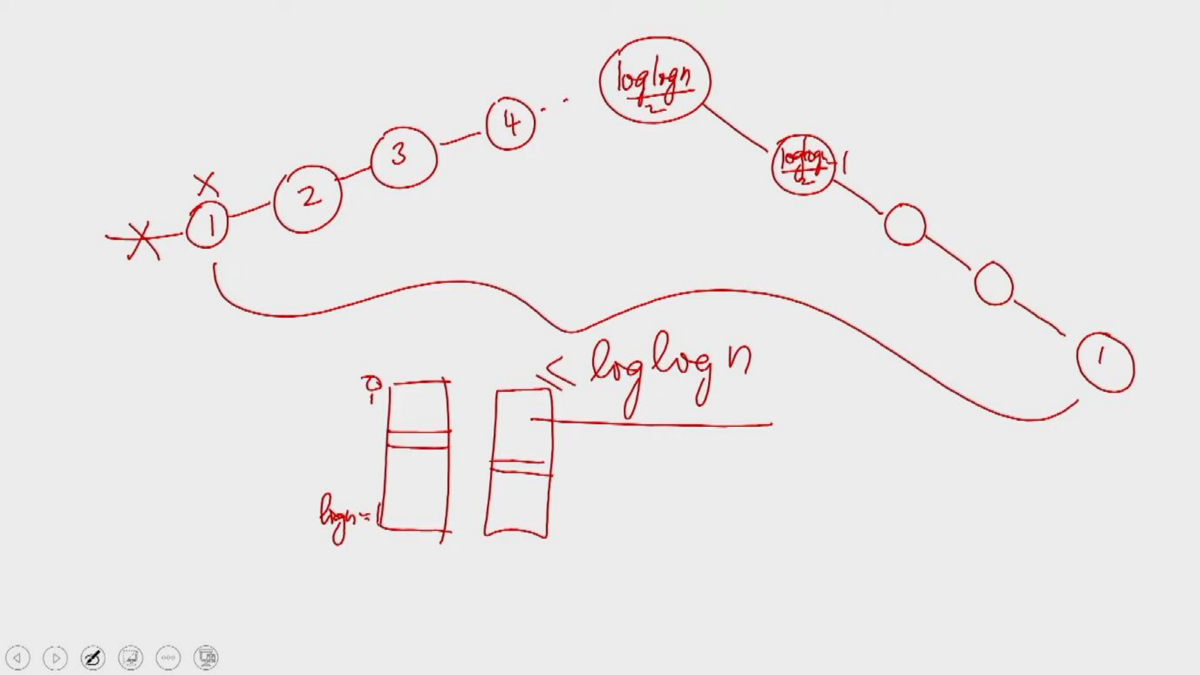
Optimal List ranking

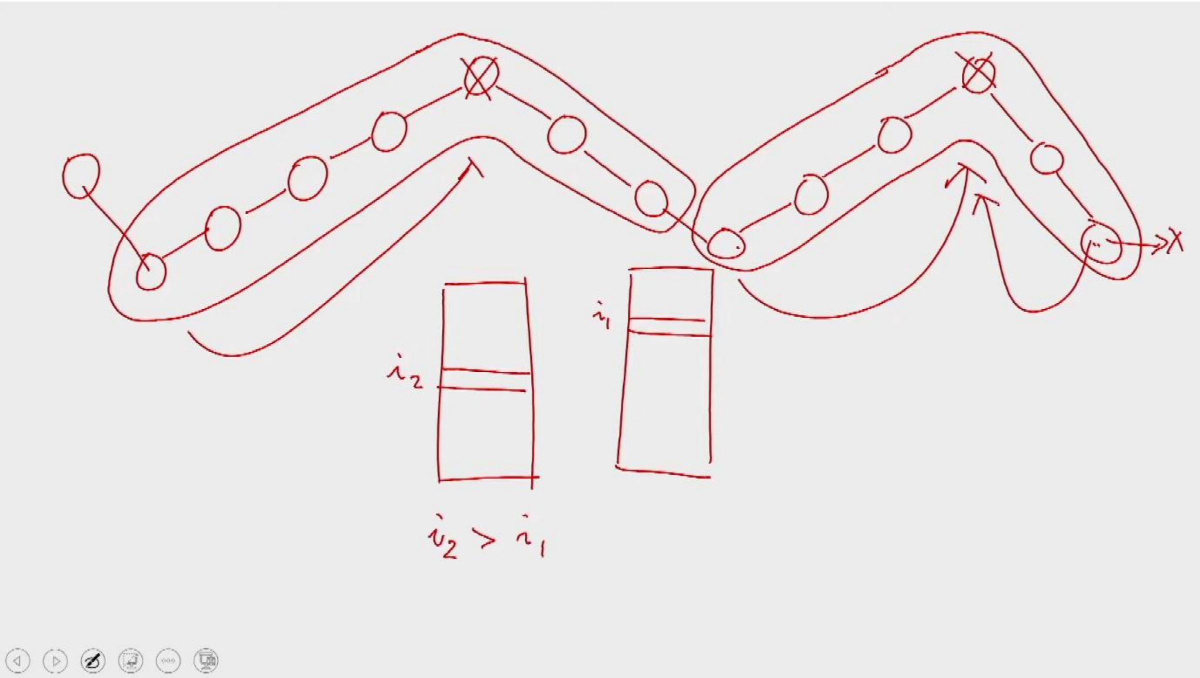
- For each processor pardo
  - if (ruler) remove a subject
    - if (that was the last subject) turn active
  - if (active)
    - if (isolated) remove self
    - else take part in the subject-ruler election

Subject-Ruler Election

- (log log n)/2 colour the subgraph induced by the active nonisolated nodes
- Break the predecessor link to each vertex with local minimum colour
- Elect every vertex that is a local maximum on depth as a ruler
- Subjects that are local minima on depth, unless they are the last nodes of their sublists, will associate with the ruler in the forward direction



loglogn/2
loglogn/2 +1
1
2
3
4
1
≤ loglogn
i
logn-1



i1
i2
i2 > i1

each ruler gets at most loglogn subjects.

Analysis of the algorithm

Charging technique

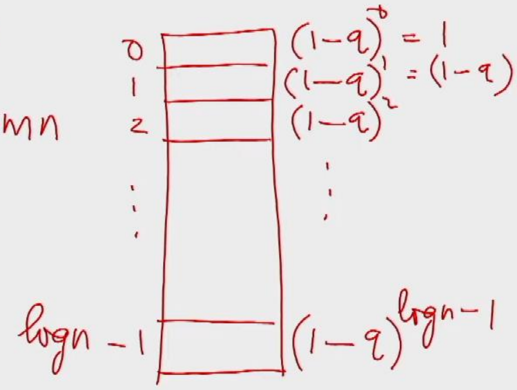
$q = 1/\log\log n$

every node in the list is given a charge.

initial distribution of the charges

for each column

$$q = \frac{1}{\log \log n}$$



The total of the initial charges

$$= \frac{n}{\log n} \sum_{j=0}^{\log n - 1} (1-q)^j$$

$$\leq \frac{n}{\log n} \left[1 + (1-q) + (1-q)^2 + \cdots \right]$$

$$= \frac{n}{\log n} \frac{1}{1-(1-q)} = \frac{n}{\log n}$$

Manipulation of the charges

① When an active isolated node removes itself

throw away the charge of the node

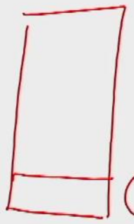
② An active node becomes a subject

throw away $\frac{1}{2}$ of the charge of the node

③ A ruler removes a subject

throw away the charge on the subject

After $O(\log n)$ iterations,
the total charge in the system
would be $\leq \frac{n}{\log n}(1-q)^{\log n}$



$(1-q)^{\log n - 1}$

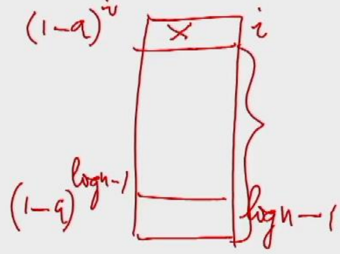
$$\frac{n}{\log n}(1-q)^{\log n - 1}$$

How much the charge in the system
reduces after every iteration.

① An active isolated node removes
this node is at depth of i

W_1 : the initial wt of the queue

$$W_1 = \sum_{j=i}^{\log n - 1} (1-q)^j$$



$$W_2 = \sum_{j=i+1}^{\log n - 1} (1-q)^j$$

$$= (1-q) \underbrace{\sum_{j=i}^{\log n - 2} (1-q)^j}$$

$$< (1-q)\, W_1 < \underline{\underline{(1-q/4)\, W_1}}$$

② An active node becomes a subject

Ruler's ~~index~~ depth is i_1

subjects $i_2, \ldots, i_K$

$$i_1 > i_2, \ldots, i_K$$

$$(1-q)^{i_1} < (1-q)^{i_2}, \ldots, (1-q)^{i_K}$$

$$W_2 = W_1 - \frac{1}{2}\sum_{j=2}^{k} (1-q)^{i_j}$$

$$W_1 < \sum_{j=1}^{k} \frac{(1-q)^{i_j}}{q} \leq \frac{2}{q}\sum_{j=2}^{k} (1-q)^{i_j}$$

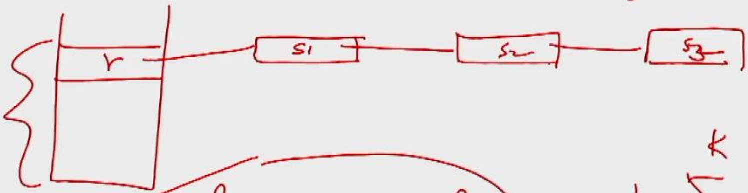
$$(1-q)^{i_1} < (1-q)^{i_2} \dots (1-q)^{i_k}$$

$$\frac{q}{4} W_1 < \underbrace{\frac{1}{2}\sum_{j=2}^{k} (1-q)^{i_j}}$$

$$W_2 = W_1 - \frac{1}{2}\sum_{j=2}^{k} (1-q)^{i_j} < W_1 - \frac{q}{4} W_1$$

$$= W_1 (1 - q/4)$$

③ A ruler removes a subject



$$W_1 = \sum_{\ell=i_1}^{\log n-1} (1-q)^{\ell} + \frac{1}{2}\sum_{j=2}^{k} (1-q)^{i_j}$$

ruler's column

Wlg, assume that the subject with the largest weight goes (otherwise, redistribute the wt)

$$W_1 < \frac{(1-q)^{i_1}}{q} + \left(\frac{k-1}{2}\right)(1-q)^{i_2}$$

✓

$$W_2 < \left(1-\frac{q}{3}\right)W_1 < \underline{\underline{\left(1-\frac{q}{4}\right)W_1}}$$

In each iteration, the total wt reduces by $(1-q/4)$

$$\frac{n}{\log n} \; \frac{1}{q} \; (1-q/4)^t$$

$$t = 5 \log n$$

$$\frac{n}{\log n}\,\frac{1}{q}\left(1-\frac{q}{4}\right)^{5\log n} = \frac{n}{\log n}\,\frac{1}{q}\left(1-\frac{1}{4/q}\right)^{(4/q)\left(\frac{5}{4}\right)q\log n}$$

$$\le \frac{n}{\log n}\,\frac{1}{q}\left(e^{-1}\right)^{1.25\,q\log n} \qquad \left(1-\frac{1}{m}\right)^{m} < e^{-1}$$

$$= \frac{n}{\log n}\,\frac{1}{q}\left(e^{-q}\right)^{1.25\log n}$$

$$\le \frac{n}{\log n}\,\frac{1}{q}\left(1-q\right)^{\left(\frac{1}{q}-1\right)q\;1.25\log n} \qquad \left(1-\frac{1}{m}\right)^{m-1} > e^{-1}$$

$$= \frac{n}{\log n}\,\frac{1}{q}\left(1-q\right)^{(1-q)\,1.25\log n}$$

$$= \frac{n}{\log n}\left(1-q\right)^{\log n}\left[\frac{(1-q)^{(0.25-1.25q)\log n}}{q}\right]$$

As $n \to \infty$ with $q = 1/\log\log n$

$$[\quad] \longrightarrow 0$$

the remaining wt in the system

$$\le \underline{\frac{n}{\log n}(1-q)^{\log n}} < \frac{n}{\log n}(1-q)^{\log n - 1}$$

$\#$ nodes in the system $< \dfrac{n}{\log n}$

Pointer Jumping : $O\left(\log \frac{n}{\log n}\right) = O(\log n)$

$O(\log n)$ time $n/\log n$ processors

EREW PRAM

optimal algorithm