External Selection

Classical algorithm

"Median of medians"

As seen previously:

- $O(n)$ I/O's
- $O(N)$ work

Asymptotically optimal.

Looking at *constants:*

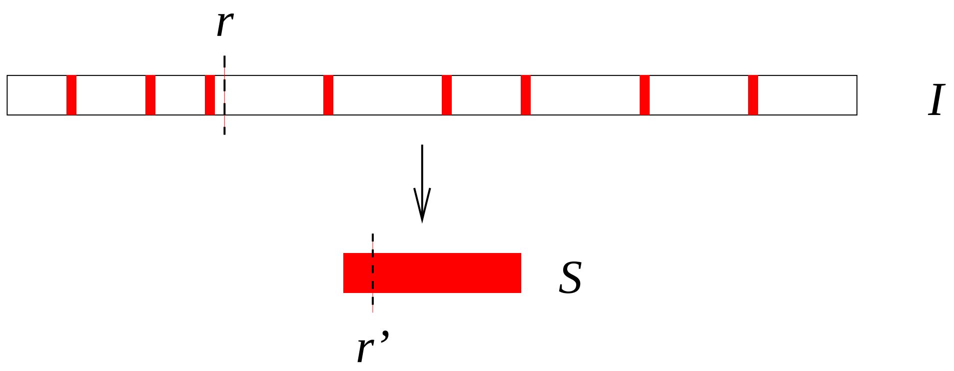
- Approx. $11 \cdot n$ I/O's

Worse than real life constants of MergeSort.

Sampling

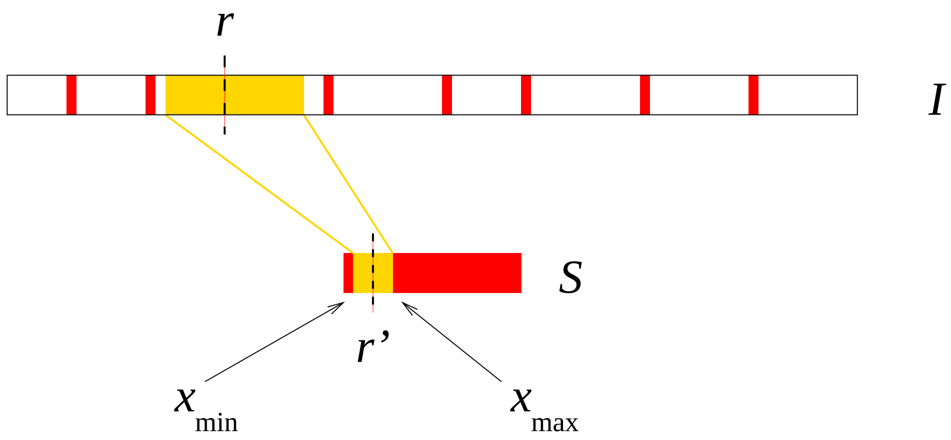
Looking for element of rank r:

Select $s = |S|$ elements out of $N = |I|$.



Good sample: element in S with rank $r' = s\frac{r}{N}$ has rank $\approx r$ in I.

Choose $x_{\min}, x_{\max} \in S$ with rank $r' \pm \Delta$ in S.



Generic algorithm:

1. Make sample S.
2. Find $x_{\min}$ and $x_{\max}$ by selection in S.
3. Set $I_{\text{new}} = \{x \in I \mid x_{\min} \leq x \leq x_{\max}\}$.
4. Recurse on I_{new} (if contains element with rank r).

Random Sampling

Easy [see paper] to choose s, Δ such that:

$$\boxed{\begin{array}{l} \text{The sampling is cheap } (s \text{ small}) \\ \text{Selection in } S \text{ is cheap } (s \text{ small}) \\ I_{\text{new}} \text{ contains sought element w.h.p.} \\ |I_{\text{new}}| \text{ is sublinear w.h.p.} \end{array}}$$

$$\Downarrow$$

Scanning (to find I_{new}) in first level of recursion dominates:

in terms of "$\leq$" (well-known)

in terms of I/O (Sibeyn [99])

Deterministic Sampling

Sibeyn [99]

Aim: minimize I/O.

Notation: N: input size

M: external memory size

s: parameter, $1 \leq s \leq M$

rank(x): no. of elms. $\leq x$

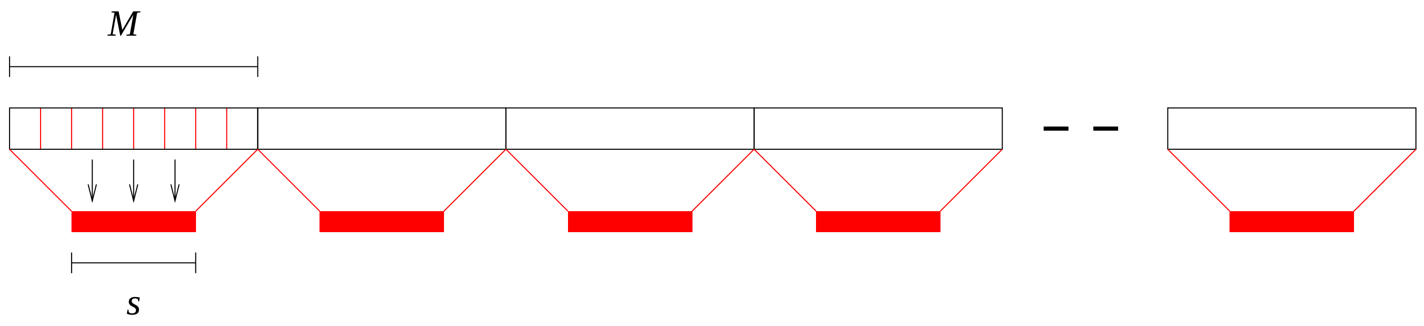
r: desired rank

(assume w.l.o.g. M, N, s are powers of 2)

Base Sampling

For each M elements of input:

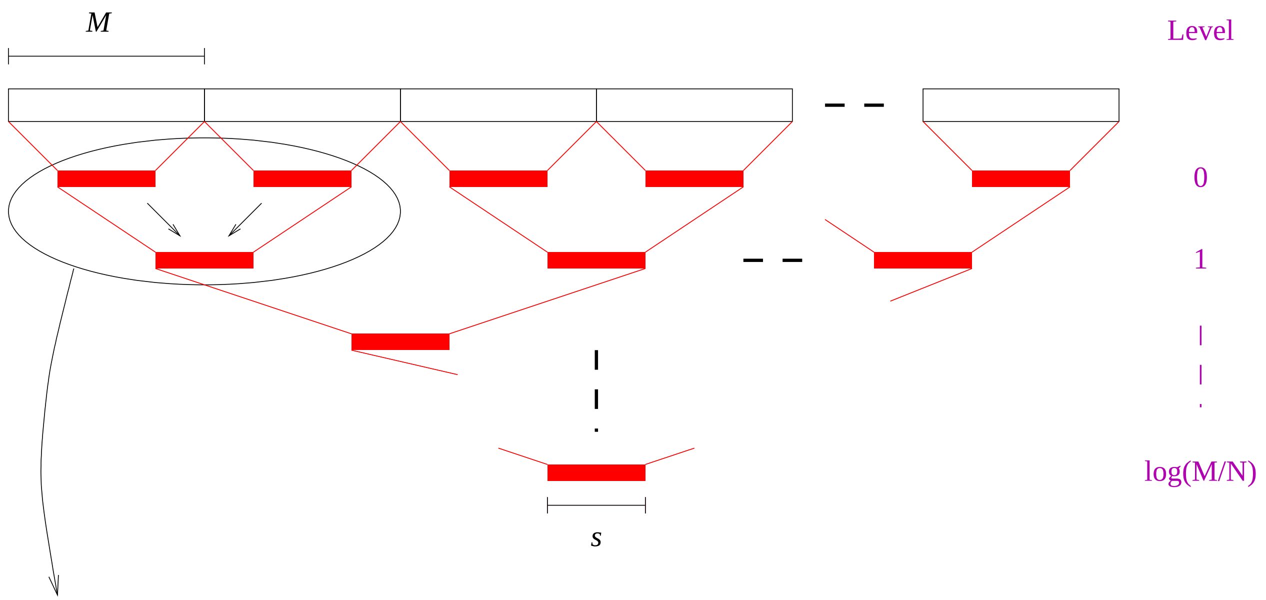
select the s elements with rank $i \cdot \frac{M}{s}$, $i = 1, 2, \ldots, s$.



Gives N/M sorted sample sets of size s at level 0:

$$S_1^0, S_2^0, \ldots, S_{N/M}^0$$

Merge Sample Sets



$$\boxed{S^k_{2j-1}, S^k_{2j} \longrightarrow S^{k+1}_j = \{x \in S^k_{2j-1} \cup S^k_{2j} \mid \mathrm{rank}(x) \text{ even}\}}$$

Note: S^k_j is a sample from a set $Q \subseteq I$ of size $M \cdot 2^k$.

Quality of S_j^k

Lemma: Let $e \in S_j^k$ have rank t in S_j^k. For the rank T of e in base set Q:

$$t \cdot \frac{M}{s} \cdot 2^k \leq T \leq (t + k/2) \cdot \frac{M}{s} \cdot 2^k$$

Proof: Induction on k.

Basis $(k = 0)$: OK by def. of base sampling.

Induction step

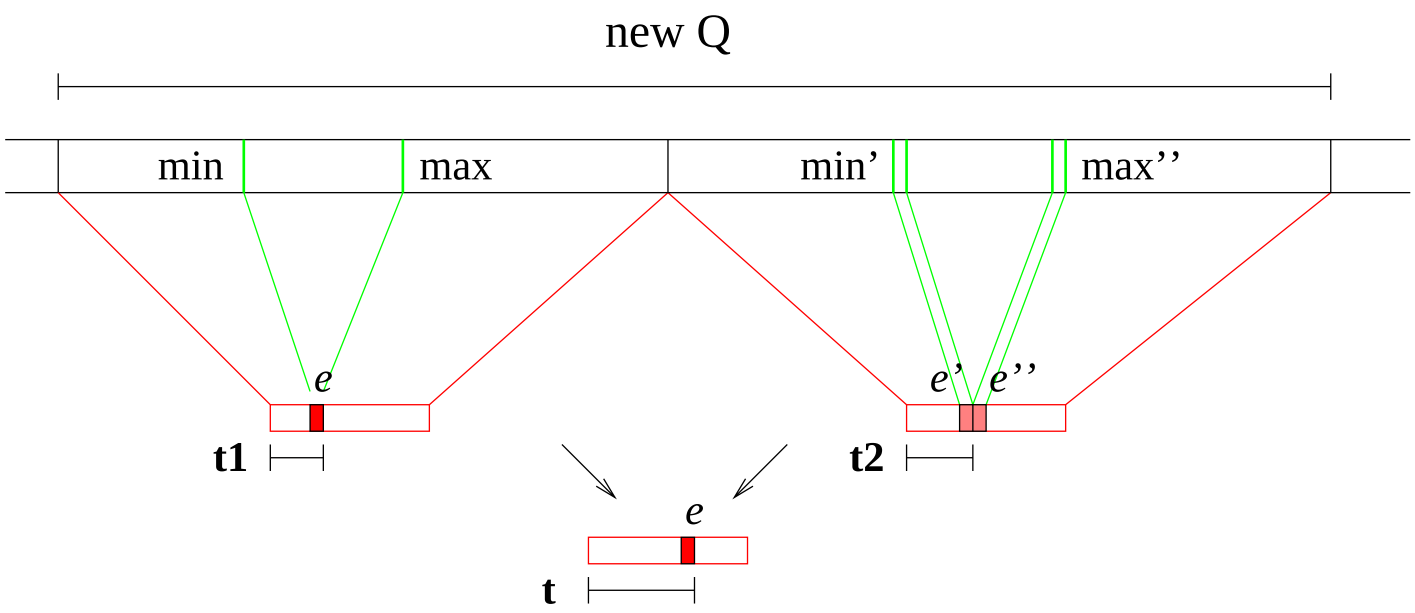
$$\boxed{S^k_{2j-1}, S^k_{2j} \longrightarrow S^{k+1}_j:}$$

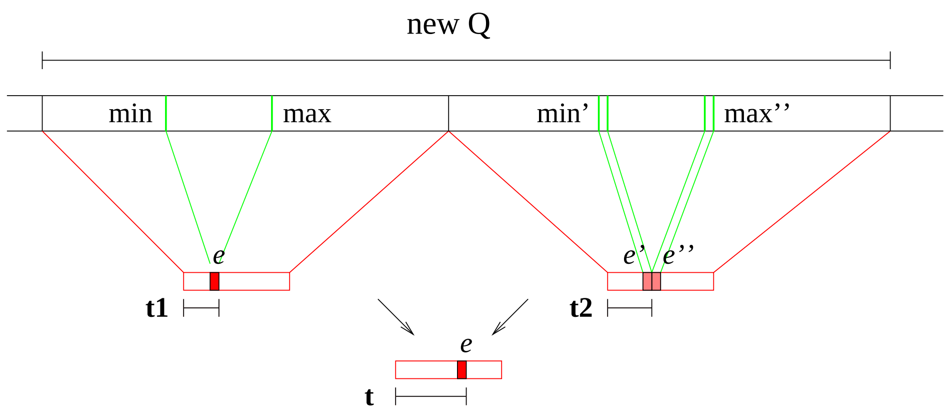
Assume w.l.o.g. $e \in S^k_{2j-1}$

t_1 = rank of e in S^k_{2j-1}

t_2 = rank of e in S^k_{2j}

$t = (t_1 + t_2)/2$





In Q:

$$
\begin{aligned}
\min + \min' &\leq T \leq \max + \max'' \\
t_1 \cdot \frac{M}{s} \cdot 2^k + t_2 \cdot \frac{M}{s} \cdot 2^k &\leq T \leq (t_1 + k/2) \cdot \frac{M}{s} \cdot 2^k \\
& \qquad\qquad + (t_2 + 1 + k/2) \cdot \frac{M}{s} \cdot 2^k \\
t \cdot \frac{M}{s} \cdot 2^{k+1} &\leq T \leq (t + (k+1)/2) \cdot \frac{M}{s} \cdot 2^{k+1}
\end{aligned}
$$

$\square$

Corollary

The last level is $K = \log(N/M)$. For the single sample set S_1^K on this level:

Corollary: It is possible to find $x_{\min}, x_{\max} \in S_1^K$ such that in I:

$$\mathrm{rank}(x_{\min}) \leq r \leq \mathrm{rank}(x_{\max})$$

$$\mathrm{rank}(x_{\max}) - \mathrm{rank}(x_{\min}) \leq (1+K/2)\frac{M}{s}2^K \cdot 2 = \Theta(\log(N/M)N/s)$$

Choice of s

Assume e.g. $N \in 2^{O(M^{1/2-\alpha})}$ for some $\alpha > 0$.

Choose $s = \Theta(M^{1/2})$.

Then $|N_{\text{new}}| = \Theta(\log(N/M)N/s) = O(N/M^{\alpha})$.

Thus, *first level of recursion dominates* the running time and the I/O.

I/0 for first round

Base sampling: N/B

Find N_{new}, given $x_{\min}$and $x_{\max}$: N/B

Merge samples: at most $2N/M \cdot s/B =$ lower order.

Total: $2N/B +$ low order terms

[CPU work: $N/M \cdot M \cdot \log(s) = N \log(s)$]