Algorithms
Professor John Reif

ALG 2.1
Randomized Algorithms for Selection and Sorting:

(a) Randomized Sampling
(b) Selection by Randomized Sampling
(c) Sorting by Random Splitting: Quicksort and Multisample Sorts

Main Reading Selections:
CLR, Chapters 8, 10

Auxillary Reading Selections:
AHU-Design, Sections 3.5-3.7
BB, Sections 4.5, 4.6
AHU-Data, Section 8.3
Handout: "Derivation of Randomized Algorithms"

Comparison Problems

input
set X of N distinct keys
total ordering < over X

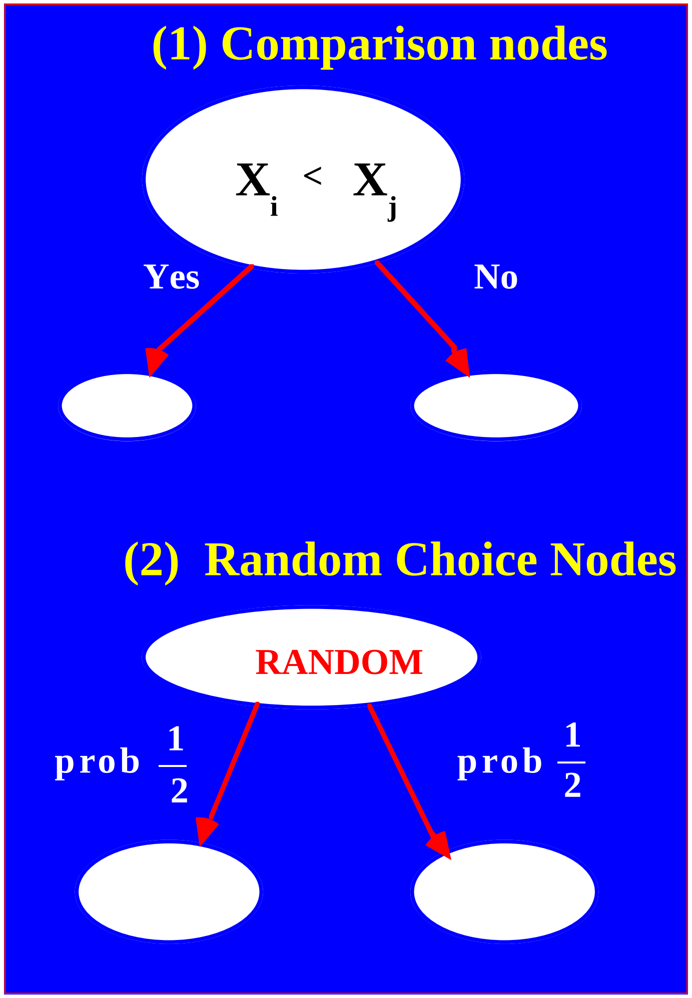
Problems

(1) for each key $x \in X$
rank$(x,X) = |\{x' \in X \mid x' < x\}| + 1$

(2) for each index $i \in \{1,\dots,N\}$
select (i,X) = the key $x \in X$ where i = rank (x,X)

(3) *sort* $(X) = (x_1, x_2, \dots, x_n)$
where x_i = select (i,X)

Randomized Comparison Tree Model



Algorithm samplerank $_s$(x,X)

begin

Let S be a random sample of X-{x} of size s

output $1 + \frac{N}{s}[\text{rank}(x,S) - 1]$

end

Lemma 1

The *expected value* of samplerank$_s$(x,X) is rank (x,X)

proof

Let k=rank(x,X)

For a random y ε X,

$$\text{Prob}(y < x) = \frac{k-1}{N}$$

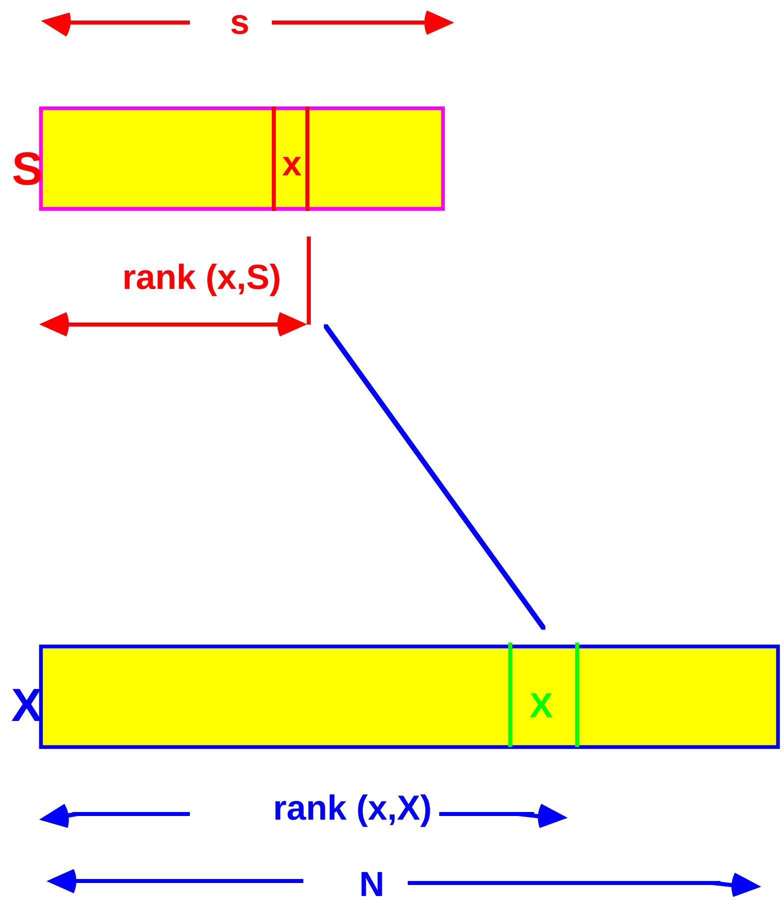
Hence E (rank(x,S)) = $s \cdot \frac{k-1}{N} + 1$

Solving for k, we get

$$\text{rank}(x,X) = k = 1 + \frac{N}{s} E[\text{rank}(x,S) - 1]$$

$$= E(\text{samplerank}(x,X))$$

S is random sample of X of size s



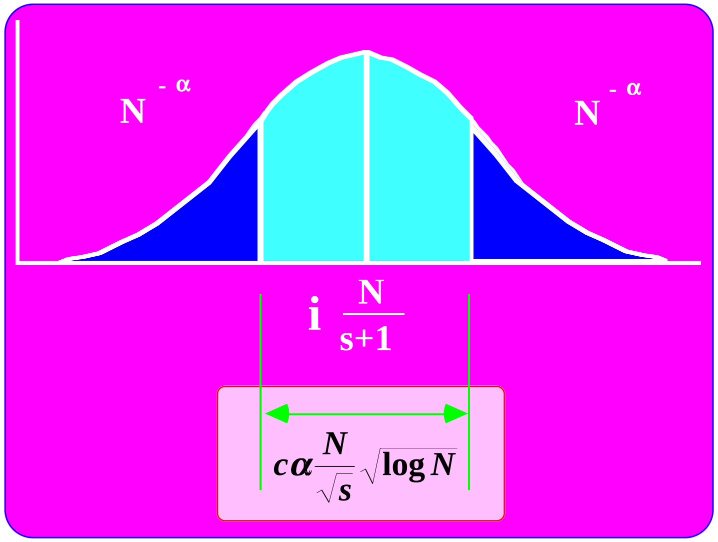
More Precise Bounds on Randomized Sampling

Let S be a random sampling of X

Let r_i = rank(select (i,S),X)

Lemma 2

$$\text{Prob}\left(\left| r_i - i \frac{N}{s+1} \right| > c\,\alpha \frac{N}{\sqrt{s}} \sqrt{\log N} \right) < N^{-\alpha}$$



proof

We can bound r_i by a Beta distribution, implying

$$mean\ (r_i) = i \frac{N}{s+1}$$

$$Var\ (r_i) \le \frac{i\,(s-i+1)}{(s+1)^2\,(s+2)} N^2$$

Weak bounds follow from *Chebychev inequality*

The Tighter bounds follow from Chernoff Bounds

Subdivision by Random Sampling

Let S be a random sample of X of size s

Let $k_1, k_2, \ldots, k_s$ be the elements of S in sorted order

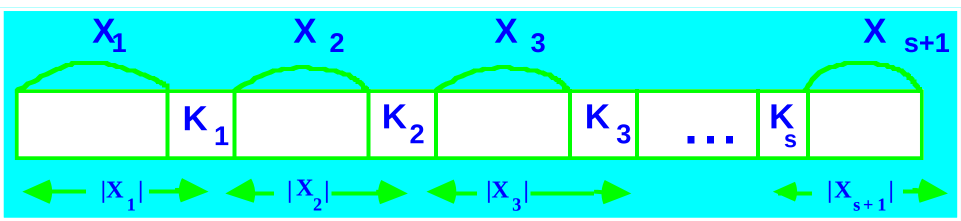
These elements *subdivide* X into s+1 subsets $x_1 = \{x \; \varepsilon \; X \mid x \leq k_1\}$

$$X_2 = \{x \; \varepsilon \; X \mid k_1 < x \leq k_2\}$$

$$X_3 = \{x \; \varepsilon \; X \mid k_2 < x \leq k_3\}$$

$$\vdots$$

$$X_{s+1} = \{x \; \varepsilon \; X \mid x > k\}$$



How even are these subdivisions?

Lemma 3

If random sample S in X is of size s and X is of size N, then S divides X into subsets each of

$$size \leq \alpha \frac{(N-1)}{s} \, ln(N) \text{ with prob } \geq 1 - N^{-\alpha}$$

proof

The number of *(s+1) partitions of X* is

$$\binom{N-1}{s} \sim \frac{(N-1)^s}{s!}$$

The *number of partitions of X* with *one* block of *size ≥ v* is

$$\binom{N-v-1}{s} \sim \frac{(N-v-1)^s}{s!}$$

So the probability of a random (s + 1) partition having a block size ≥ v is

$$\frac{\binom{N-v-1}{s}}{\binom{N-1}{s}} \sim \left(\frac{N-v-1}{N-1}\right)^s = \left(1-\tfrac{1}{Y}\right)^s \text{ for } Y = \frac{N-1}{v}$$

$$\leq \left(1-\tfrac{1}{Y}\right)^{Y\left(\frac{s}{Y}\right)} \sim e^{-\frac{s}{Y}} = e^{-\frac{s\,v}{N-1}}$$

$$\leq N^{-\alpha} \text{ if } v = \alpha\frac{(N-1)}{s}\ln N$$

since $\left(1-\frac{1}{Y}\right)^{Y} < e^{-1}$

Randomized Algorithms for Selection

"canonical selection algorithm"

Algorithm

can select (i,X)
input set X of N keys
index i ε {1,...,N}

[0] if N=1 then *output* X

[1] select a bracket B of X, so that select (i,X) ε B with high prob.

[2] Let i_1 be the number of keys less than any element of B

[3] *output* can select (i-i_1, B)

Note: **B found by *random sampling* (also must cover cases of low prob to *always get correct output*)**

Hoar's Selection Algorithm

lgorithm **Hselect (i,X) where 1≤ i ≤N**

begin
if X = {x} *then output* x *else*
choose a *random splitter* k ε X
let B = {x ε X | x < k}

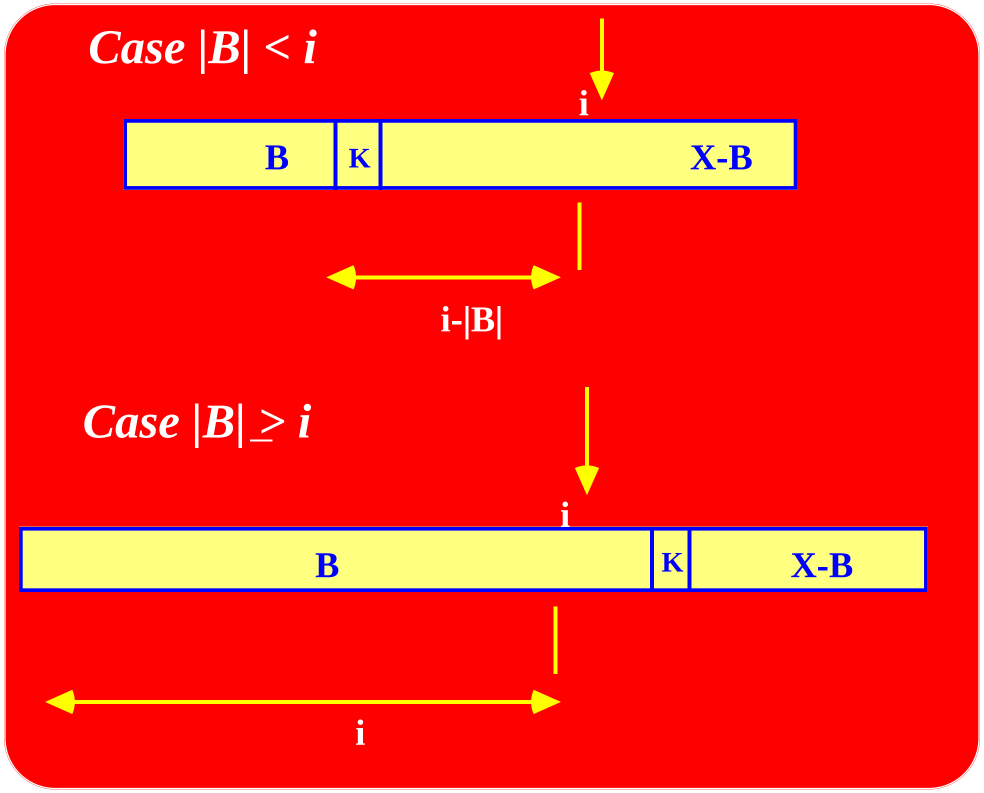
if |B| ≥ i *then output* Hselect (i,B)
else output Hselect (i-|B|, X-B)
end

sequential time bound T(i,N) has mean

$$\overline{T}(i,N) = N + \frac{1}{N}\left[\sum_{j=1}^{i}\overline{T}(i-j,\ N-j) + \sum_{j=i+1}^{N}\overline{T}(i,j)\right]$$

$$= 2N + \min(i,\ N-i) + o(N)$$

random splitter k ε X Hselect(i,X) has two cases



Inefficient: each recursive call requires *N* *comparisons*, but only

reduces problem size by average $\frac{1}{2}$ N

Improved Randomized Selection

by Floyd and Rivest

Algorithm
FRselect(i,X)

```
begin
    if X = {x} then output x else
    Choose k1, k2 ε X  such that k1 < k2
    let r1 = rank(k1, X),  r2 = rank(k2, X)
    if r1 > i then FRselect(i, {x ε X | x < k1} )
else if r2 > i then FRselect(i-r1, {x ε X | k1≤x≤k2}}
    else  FRselect(i-r2, {x ε X | x > k2})
end
```

problem:

We must choose k_1, k_2 so that with *high likelihood,*

$$k_1 \le select(i,X) \le k_2$$

Choose *random sample* $S \le X$ size s

Define:

$$k_1 = \text{select}\left(i\frac{(s+1)}{(N+1)} - \delta,\ S\right)$$

$$k_2 = \text{select}\left(i\frac{(s+1)}{(N+1)} + \delta,\ S\right)$$

where $\delta = \lceil \sqrt{d\,\alpha\, s \log N} \rceil$, d=constant

Lemma 2 implies:

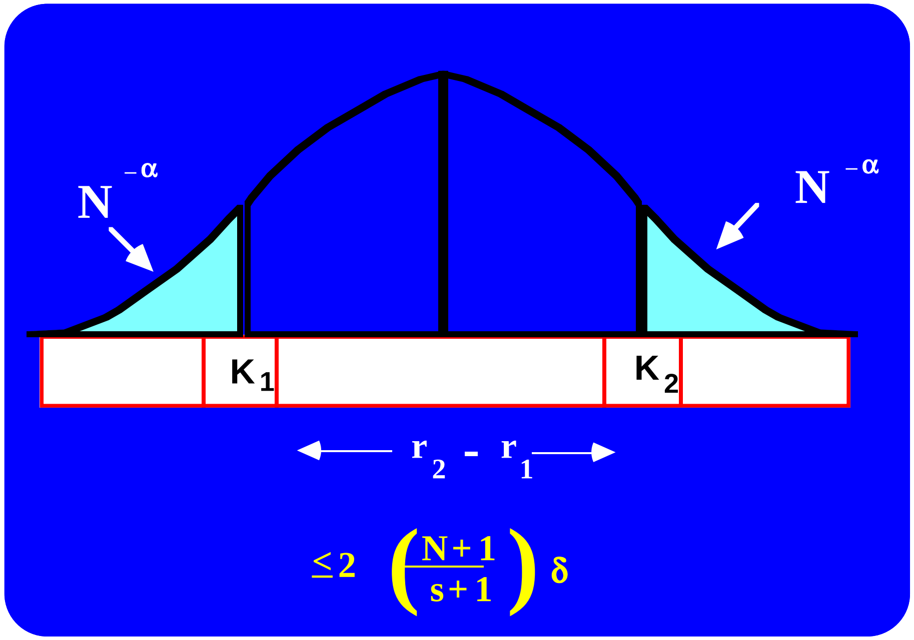
$$Prob\ (r_1 > i) < N^{-\alpha}$$

and

$$Prob\ (r_2 < i) < N^{-\alpha}$$

where $r_1 = rank\ (k_1, X)$

$r_2 = rank\ (k_2, X)$



Expected Time Bound

$$\overline{T}(i,N) \le N + 2\overline{T}(-,s)$$
$$+ \text{Prob}(r_1 > i) \cdot \overline{T}(i, r_1)$$
$$+ \text{Prob}(i > r_2) \cdot \overline{T}(i - r_1, N - r_2)$$
$$+ \text{Prob}(r_1 \le i \le r_2) \cdot \overline{T}(i - r_1, r_2 - r_1)$$

$$\le N + 2\overline{T}(-,s) + 2N^{-\alpha} \cdot N + \overline{T}\left(i, 2\left\lceil \frac{N+1}{s+1} \delta \right\rceil\right)$$

$$\le N + \min(i, N-i) + o(N)$$

if we set $\delta < \dfrac{3}{\alpha}$ and $s = N^{\frac{2}{3}} \log N$

$$= o(N)$$

note

with prob $\ge 1 - 2N^{-\alpha}$ each

recursive call costs only $O(s)$ $= o(N)$

rather than N in previous algorithm

Randomized Sorting Algorithms

"canonical sorting algorithm"

Algorithm **cansort(X)**
begin
if x = {x} *then output* X *else*
choose a *random sample* S of X of size s
Sort S
S subdivides X into s+1 subsets
$X_1, X_2, \ldots, X_{s+1}$
output cansort(X_1) cansort(X_2) ... cansort(X_{s+1})
end

Problem:
must subdivide X into subsets of *nearly equal size* to minimize number of comparisons

Solution: random sampling!

Hoar's Randomized Sorting Algorithm

uses ***sample size s=1***

Algorithm **quicksort(X)**
begin
if |X|=1 *then output* X else
choose a *random splitter* k ε X
output quicksort({x ε X|x<k}) · (k) · quicksort({x ε X|x>k})
end

Expected Time Cost

$$\overline{T}(N) \le N-1 + \frac{1}{N}\sum_{i=1}^{N}\left(\overline{T}(i-1) + \overline{T}(N-i)\right)$$

$$\le 2\ N \log N$$

inefficient:
need to divide problem size by $\frac{1}{2}$ with *high likelihood!*

Better choice splitter is

$$k = \text{sample select}_S\ (\lfloor N/2 \rfloor, N)$$

Algorithm $\text{samplesort}_s(X)$

begin

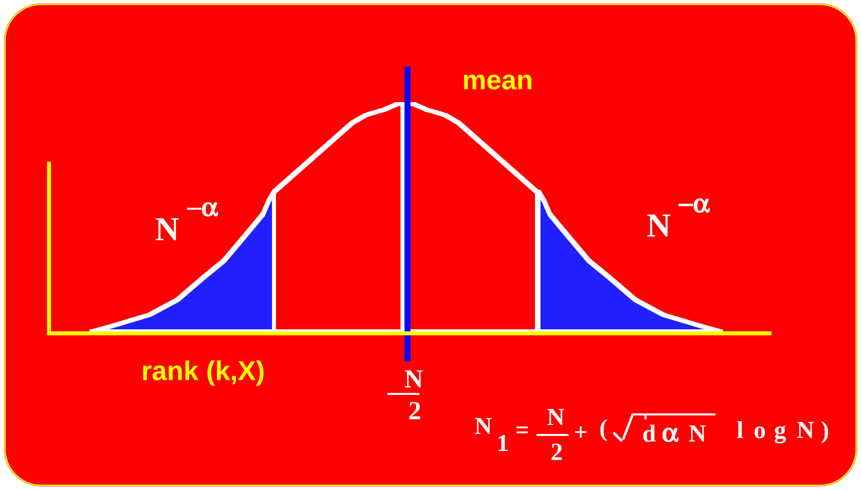
if $|X|=1$ *then output* X
choose a *random subset* S of X *size*
s=N/log N
$k \leftarrow \text{select}(\lfloor s/2 \rfloor, S)$ cost time $o(N)$

output
$\text{samplesort}_S(\{x\ \varepsilon X | x<k\}) \cdot (k) \cdot \text{samplesort}_S(\{x\ \varepsilon X | x>k\})$

end

By Lemma 2, rank(k,X) is *very nearly the mean:*

$$\text{Prob}\left(\left|\text{rank}(k,X) - \frac{N}{2}\right| > \sqrt{d\ \alpha N}\ \log N\right) < N^{-\alpha}$$



Expected Time Bounds

$$\overline{T}(N) \le 2\,\overline{T}(N_1) + N^{-\alpha}\ \overline{T}(N) \cdot N + o(N) + N - 1$$

$\approx$ **log (N!) is** ***optimal for comparison trees!***

Open Problems in Selection and Sorting

(1) ***Improve*** **Randomized Algorithms to** ***exactly match lower bounds*** **on number of comparisons**

(2) Can we ***de*** **randomize these algorithms - i.e., give** ***deterministic algorithms*** **with the same bounds?**