Advanced Data Structures

(Random topics Gerth finds interesting...)

Gerth Stølting Brodal

Formalities

- Versions of course:
  - ”**normal**”, 3 implementation projects, groups 2-3 people
  - ”**honours**”, 4 theoretical projects, individual
- **Exam**: Individual discussion about projects (Jan.)
- **Literature**: Research papers
- **Lectures**: High level discription of ideas

Problem

Input: Unordered list $(x_1, x_2, \ldots x_n)$ and $y_1 < y_2 < \cdots < y_k$

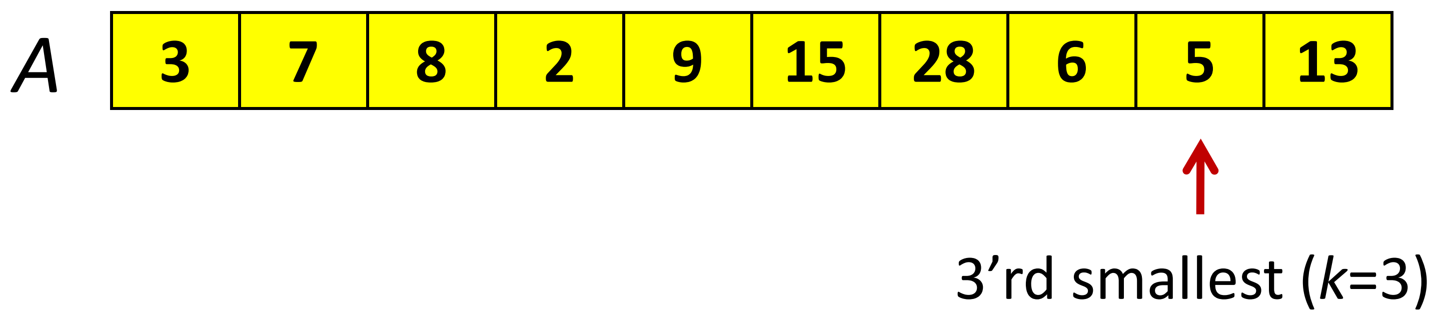
Output: For each y_i determine if y_i is in the list

*Comparison model

Selection

[T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, *Introduction to Algorithms,* 2001, Chapter 9.3]
Original [M. Blum, R.W. Floyd, V. Pratt, R. Rivest and R. Tarjan, *Time bounds for selection*, J. Comp. Syst. Sci. 7 (1973) 448-461]

Problem: Given an array *A* of *n* elements and an integer *k,* find the k'th smallest element in *A*



Randomized Selection Algorithm

[C. A. R. Hoare: Algorithm 65: find. Commun. ACM 4(7): 321-322 (1961)]

Algorithm QuickSelect(A,k)

p = random element from A

$A_< = \{ e \mid e \in A \text{ and } e < p \}$

$A_> = \{ e \mid e \in A \text{ and } e > p \}$

if $|A_<| = k$-1 **then** return p

if $|A_<| > k$-1 **then** return QuickSelect($A_<$,k)

return QuickSelect($A_>$, $k-|A_<|-1$)

Thm QuickSelect runs in expected $O(n)$ time

Proof $O(\sum_{i=0}^{\infty} n(3/4)^i) = O(n)$ □

Deterministic Selection Algorithm

[M. Blum, R.W. Floyd, V. Pratt, R. Rivest and R. Tarjan, *Time bounds for selection*, J. Comp. Syst. Sci. 7 (1973) 448-461]

Algorithm Select(A,k)

if $|A| = 1$ **then** $A[1]$

A | 2 4 1 3 7 **8 6 4 2 10** 9 11 3 2 12 ... | *n*/5 groups

5 5 5

A' | 3 **6** 9 ... | *n*/5 medians, one for each group

$p = \text{Select}(A', |A'|/2)$

$A_< = \{ e \mid e \in A \text{ and } e < p \}$

$A_> = \{ e \mid e \in A \text{ and } e > p \}$

if $|A_<| = k-1$ **then** return p

if $|A_<| > k-1$ **then** return Select($A_<$,k)

return Select($A_>$, $k-|A_<|-1$)

Deterministic Selection Algorithm

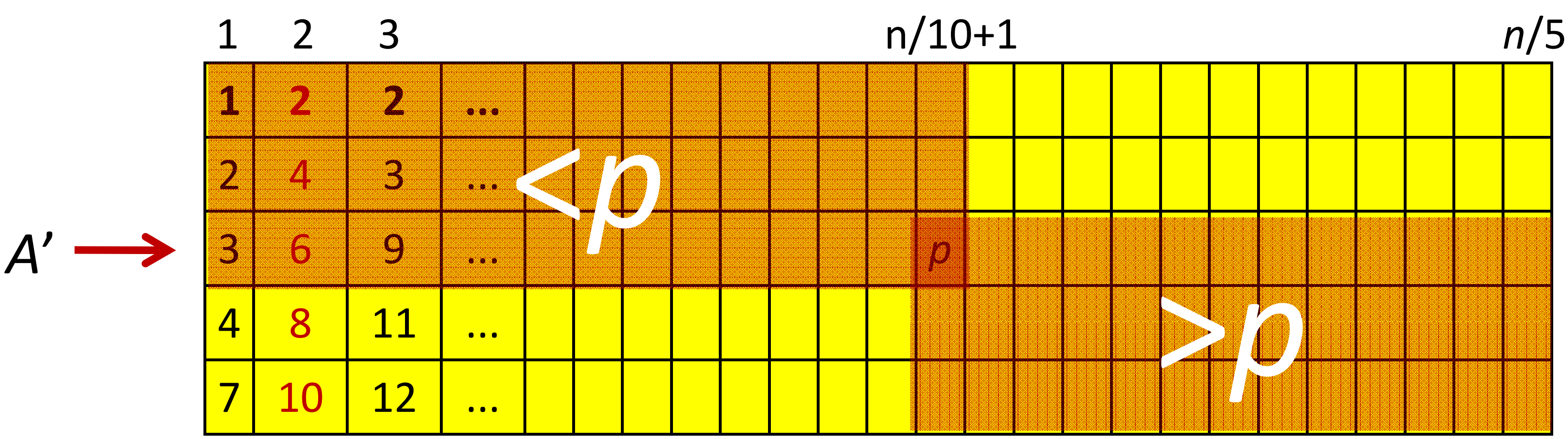
Thm Select runs in worst-case $O(n)$ time

Proof

$$T(n) = \begin{cases} 1 & n = 1 \\ n + T(n/5) + T(7n/10) & \text{otherwise} \end{cases}$$

$$= O(n)$$

□

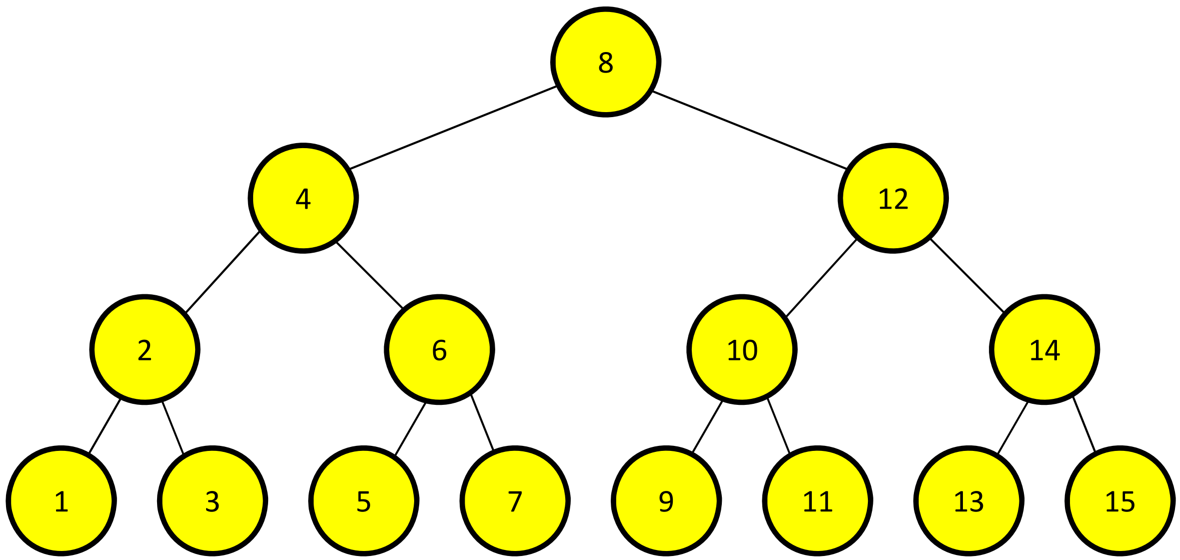


consider each group as a sorted column

Application:
Lazy Construction of Search Trees

[Y.-T. Ching, K. Mehlhorn, M.H.M. Smid: Dynamic Deferred Data Structuring. Inf. Process. Lett. (IPL) 35(1):37-40 (1990)]

1	**10**	**3**	**6**	**2**	**15**	**12**	**7**	**14**	**13**	**8**	**4**	**5**	**11**	**9**

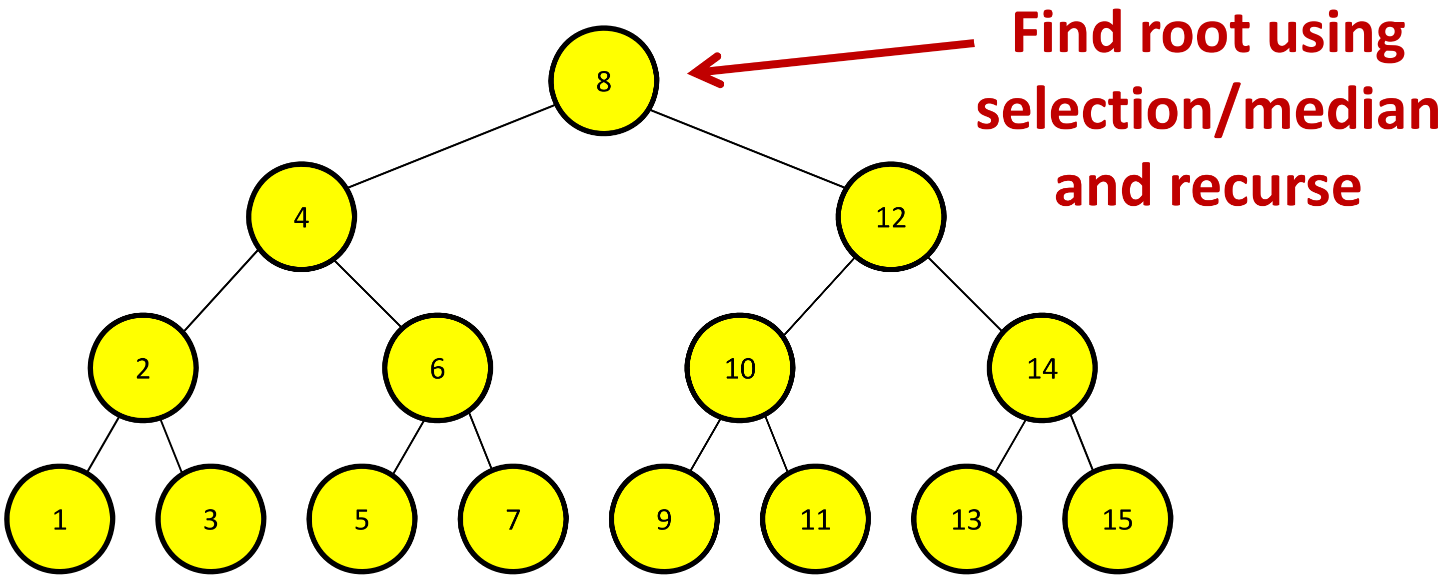


Construction T= sorting = $O(n \cdot \log n)$ Searching $O(\log n)$

Construction + k searches **$O((n+k) \cdot \log n)$**

Application:
Lazy Construction of Search Trees

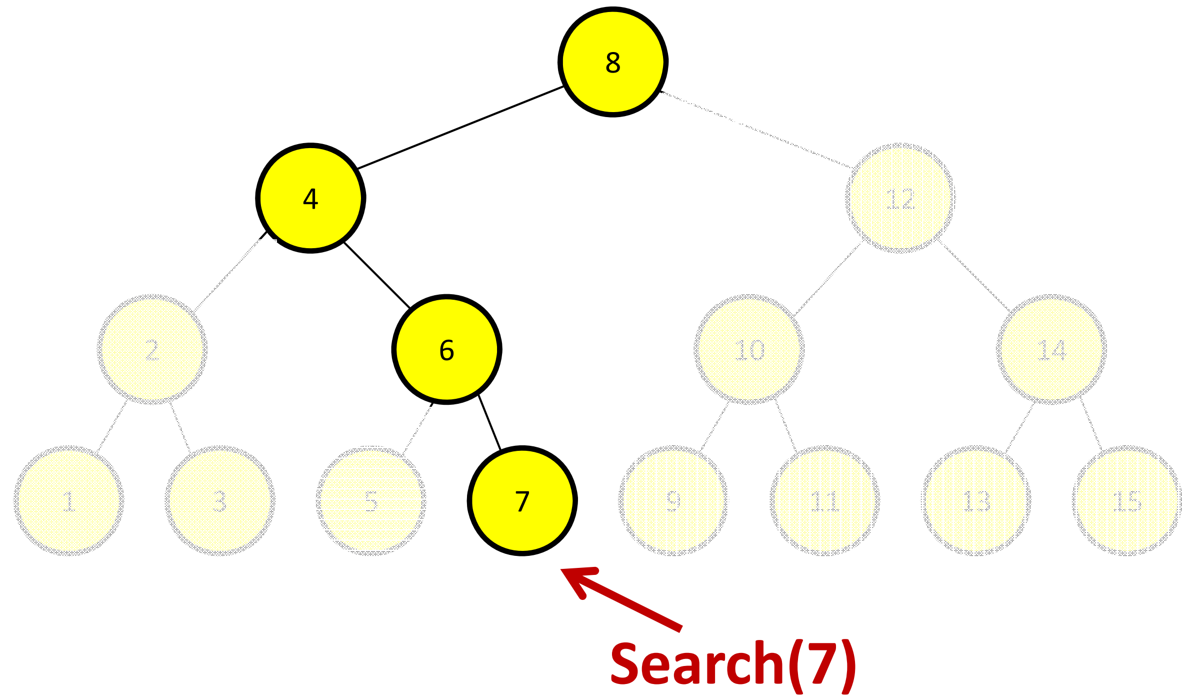
1	**10**	**3**	**6**	**2**	**15**	**12**	**7**	**14**	**13**	**8**	**4**	**5**	**11**	**9**



Construction $O(n \cdot \log n)$

Application:
Lazy Construction of Search Trees

1	**10**	**3**	**6**	**2**	**15**	**12**	**7**	**14**	**13**	**8**	**4**	**5**	**11**	**9**

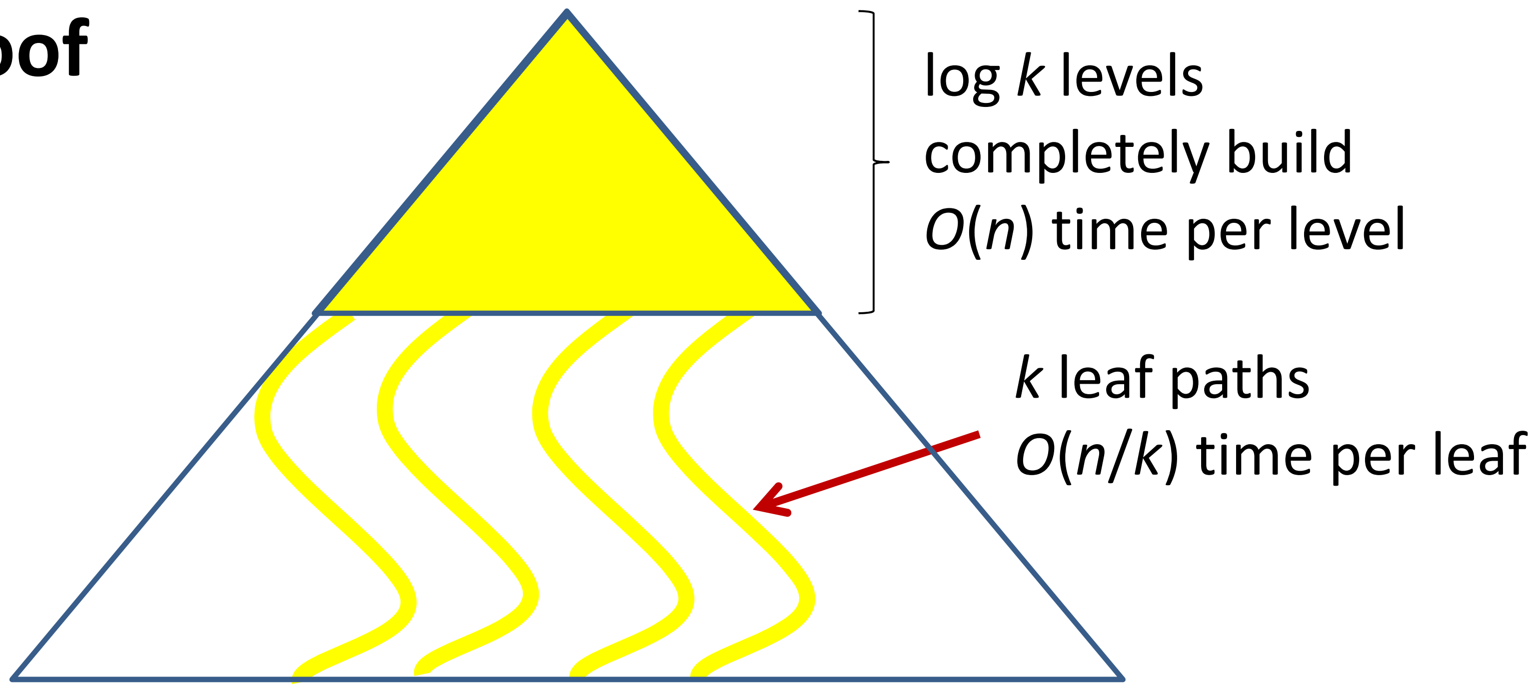


Lazy construct nodes on 1 search path: $1+2+4+\cdots+n/2+n=O(n)$

Application: Lazy Construction of Search Trees

Thm Lazy construction + *k* searches
worst-case O(*n*·log *k*) time

Proof

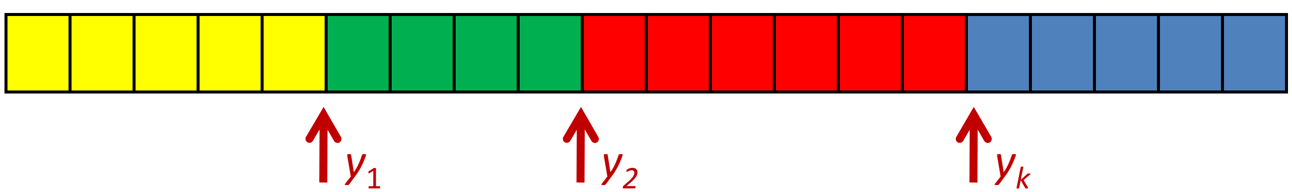


□

Thm Lazy construction + k searches requires worst-case $\Omega(n \cdot \log k)$ time

Proof

- Consider the elements of the input array in sorted order, and consider k unsuccessful search keys $y_1 < \cdots < y_k$.
- The algorithm must determine the **color (inteval between two search keys)** of each element in the input array, otherwise an element could have been equal to a search key.
- There are $(k+1)^n$ colorings.
- A dicision tree must determine the coloring.
- Decision tree depth is $\geq \log_2 (k+1)^n = n \cdot \log_2 (k+1)$



unsuccesfull searches

□