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 description 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


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

$A = \begin{bmatrix} 3 & 7 & 8 & 2 & 9 & 15 & 28 & 6 & 5 & 13 \end{bmatrix}$

3’rd smallest ($k=3$)
Randomized Selection Algorithm


Algorithm QuickSelect(A,k)

\[ p = \text{random element from } A \]

\[ A_\prec = \{ e \mid e \in A \text{ and } e < p \} \]

\[ A_\succ = \{ e \mid e \in A \text{ and } e > p \} \]

if |A_\prec| = k-1 then return p

if |A_\prec| > k-1 then return QuickSelect(A_\prec,k)

return QuickSelect(A_\succ, k-|A_\prec|-1)

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

Proof \( O(\sum_{i=0}^{\infty} n(3/4)^i) = O(n) \) \( \square \)
Deterministic Selection Algorithm

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

**Algorithm Select(A, k)**

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

2. **if** $A = n/5$ groups
3. $A' = n/5$ medians, one for each group
4. $p = \text{Select}(A', |A'|/2)$
5. $A_\prec = \{ e \mid e \in A \text{ and } e < p \}$
6. $A_\succ = \{ e \mid e \in A \text{ and } e > p \}$
7. **if** $|A_\prec| = k - 1$ **then** return $p$
8. **if** $|A_\prec| > k - 1$ **then** return $\text{Select}(A_\prec, k)$
9. return $\text{Select}(A_\succ, k - |A_\prec| - 1)$
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


Construction \( T = \text{sorting} = O(n \cdot \log n) \)  \hspace{1cm}  Searching \( O(\log n) \)

Construction + \( k \) searches \( O((n+k) \cdot \log n) \)
Application:
Lazy Construction of Search Trees

Find root using selection/median and recurse

Construction $O(n \cdot \log n)$
Application:
Lazy Construction of Search Trees

Lazy construct nodes on 1 search path: 1+2+4+⋯+n/2+n=O(n)
Application: Lazy Construction of Search Trees

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

**Proof**

- Log $k$ levels completely built in $O(n)$ time per level
- $k$ leaf paths take $O(n/k)$ time per leaf
**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 (interval 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 decision tree must determine the coloring.
- Decision tree depth is $\geq \log_2 (k+1)^n = n \cdot \log_2 (k+1)$