Predecessor Queries in Dynamic Integer Sets

Gerth Stølting Brodal

Max-Planck-Institut für Informatik
Saarbrücken, Germany

The Problem

Maintain a set S of size n under the operations

Operation	Description
• Insert(e)	Insert e into S
• Delete(e)	Delete e from S
• Pred(e)	Return the largest element $\leq e$ in S
• FindMin/-Max	Return the minimum/maximum in S

The Problem

Maintain a set S of size n under the operations

Comparison Model

Operation	Priority Queue	Dictionary	Trade off
• Insert(e)	$O(\log n)$	$O(\log n)$	$O(t)$
• Delete(e)	$O(\log n)$	$O(\log n)$	
• Pred(e)	—	$O(\log n)$	$\Downarrow$ $\frac{n}{2^{O(t)}}$
• FindMin/-Max	$O(1)$	$O(1)$	

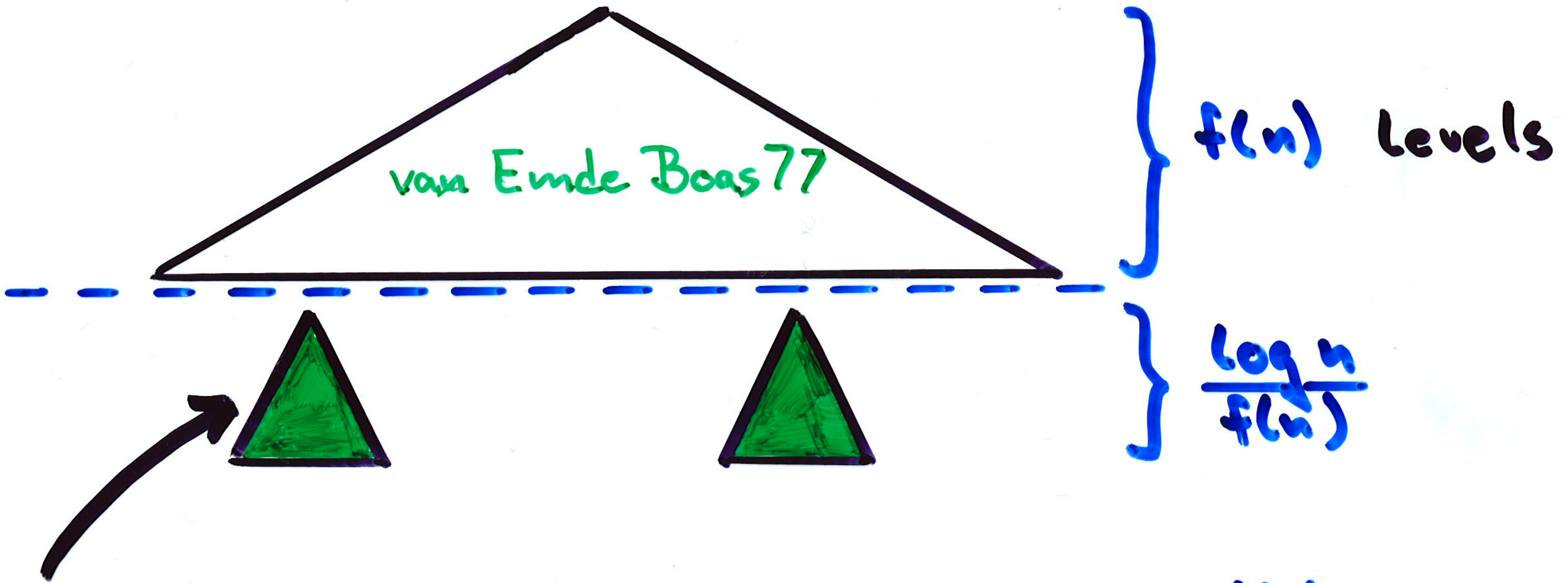
Brodal/Chaudhuri/Radhakrishnan '96

The Problem on a Practical RAM

Miltersen 96

- Unit cost RAM
- +, Shifting, bit-wise boolean operations, direct and indirect addressing, jumps, conditional statements
- Wordsize is w
- Elements are integers in the range $0..2^w-1$

The Data Structure



Packed search trees containing $\frac{w}{2^{f(n)}}$ bit integers (i.e., $2^{f(n)}$ integers can be packed into a word) supporting:

- Insert, Delete : $O(f(n))$
- Pred : $O\left(\frac{\log n}{f(n)}\right)$
- FindMin, FindMax : $O(1)$

The Problem

Maintain a set S of size n under the operations

Practical RAM

Operation	van Emde Boas 77	Thorup 96	Andersson 95
• Insert(e)	$O(\log w)$	$O(\log\log n)$	$O(\sqrt{\log n})$
• Delete(e)	$O(\log w)$	$O(\log\log n)^*$	$O(\sqrt{\log n})$
• Pred(e)	$O(\log w)$	—	$O(\sqrt{\log n})$
• FindMin/-Max	$O(1)$	$O(1)$	$O(1)$

* Delete requires **amortized** $O(\log\log n)$ time.
DeleteMin is supported in worst case $O(\log\log n)$ time.

The Problem

Maintain a set S of size n under the operations

Results (for Practical RAM)

Operation	Main*	Corollary
• Insert(e)	$O(f(n))$	$O(\log\log n)$
• Delete(e)	$O(f(n))$	$O(\log\log n)$
• Pred(e)	$O\left(\frac{\log n}{f(n)}\right)$	$O\left(\frac{\log n}{\log\log n}\right)$
• FindMin/-Max	$O(1)$	$O(1)$

* $f(n)$ is a smooth function satisfying $\log\log n \le f(n) \le \sqrt{\log n}$

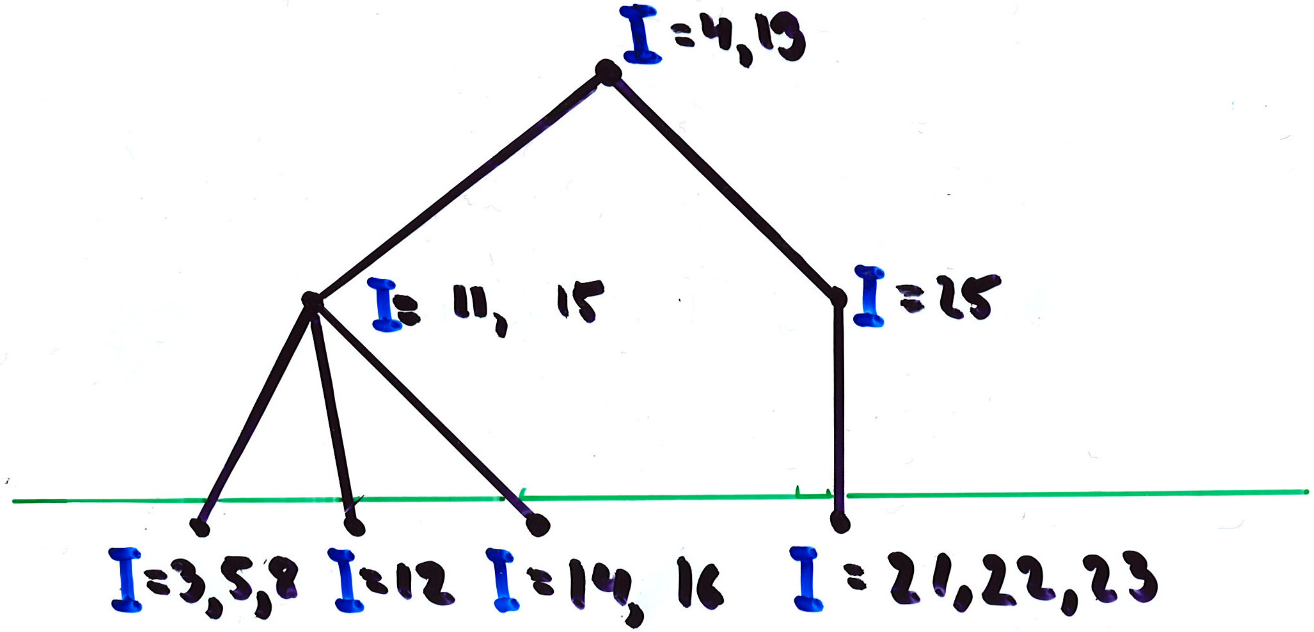
Packed Search Trees – Preliminaries

- Assumption: Lists containing at most k integers can be stored in $O(1)$ words. $(k = 2^{f(n)})$
- **Lemma** (Albers/Hagerup 92)

  Two sorted lists can be **merged** in $O(\log k)$ time
- **Lemma**

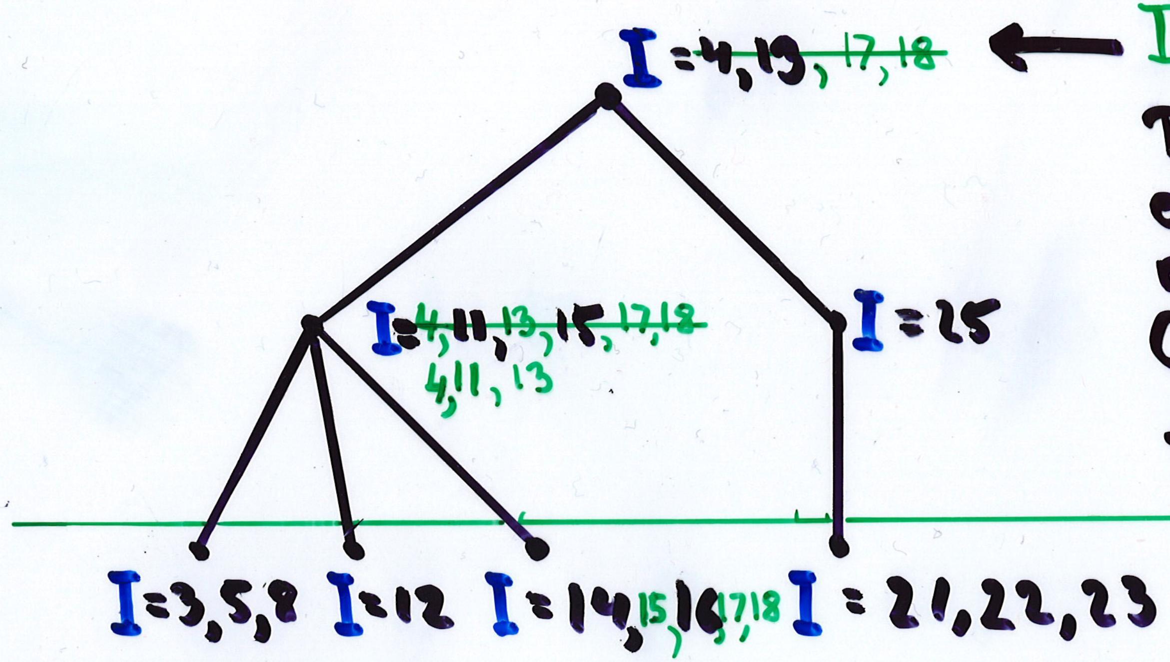
  Given two sorted lists A and B, the list $A \setminus B$ can be computed in $O(\log k)$ time

Packed Search Trees



- The tree is an $(1, \frac{k}{\log^4 n})$-tree of height $\frac{\log n}{\log k}$
- Leaf buffers have always size $\Theta(k)$
- Internal buffers have size $\le k/\log^3 n$

Packed Search Trees

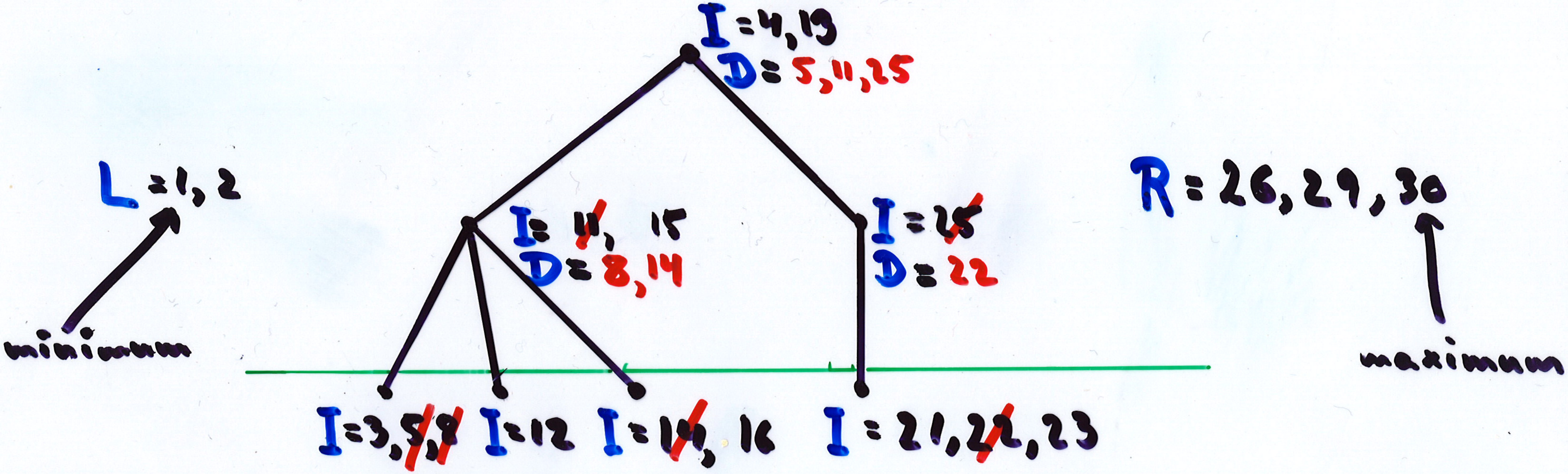


Insert(17) and Insert(18).

Removing $O(\log n)$ elements from the root buffer takes $O\left(\frac{\log n}{\log k} \cdot \log k\right) = O(\log n)$ time.

- The tree is an $\left(1, \frac{k}{\log^4 n}\right)$-tree of height $\frac{\log n}{\log k}$
- Leaf buffers have always size $\Theta(k)$
- Internal buffers have size $\leq k/\log^3 n$

Packed Search Trees



- The tree is an $(1, \frac{k}{\log^4 n})$-tree of height $\frac{\log n}{\log k}$
- Leaf buffers have always size $\Theta(k)$
- Internal buffers have size $\le k/\log^3 n$
- L and R are nonempty and have size $O(k)$

Open Problems

- Can Insert/Delete be supported in $O(\log\log n)$ time, while supporting Pred in $O(\sqrt{\log n})$ time (or better)?
- Find a tradeoff between the update time and Pred.
- —"— FindMin/FindMax.