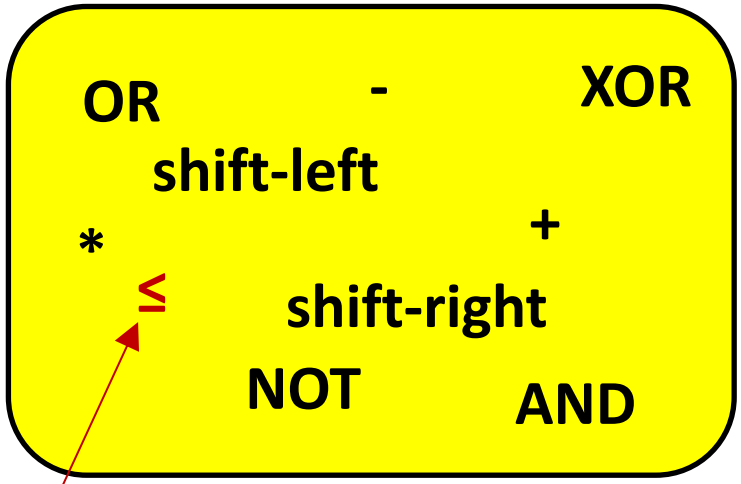
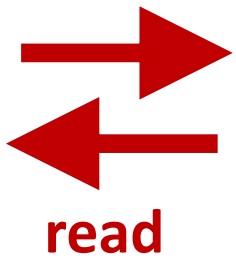
Implicit model

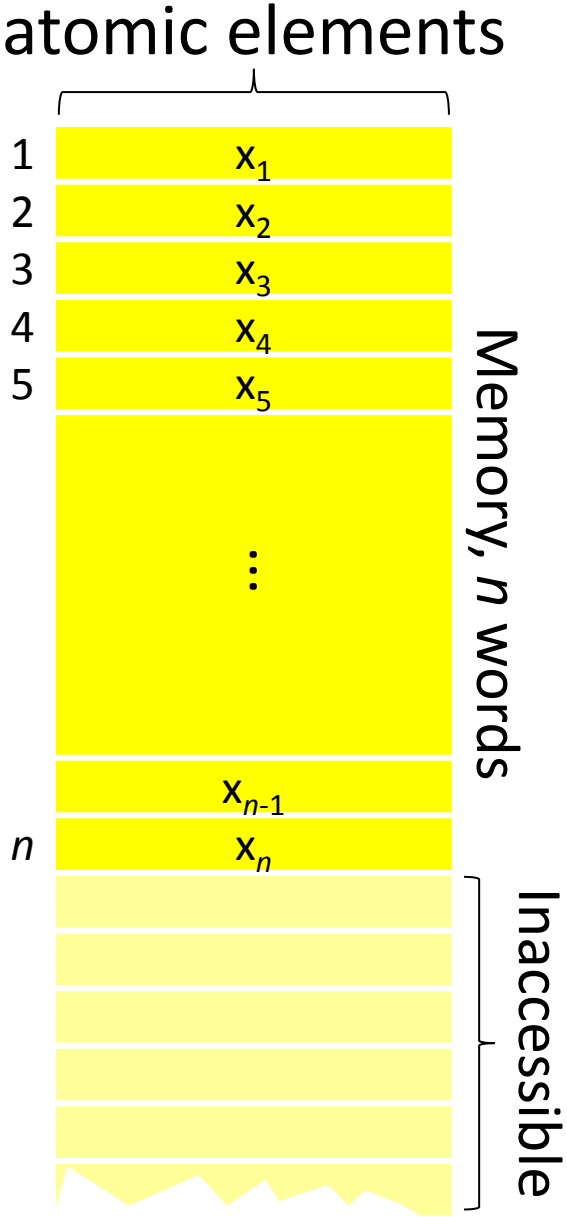
CPU, O(1) registers

O(log *n*) bits or atomic elements



only allowed operation on elements





Complexity =
- **# reads**
- **+ # writes**
- **+ # instructions performed**

Sorting

Implicit priority queue

	Comparisons	Data moves/writes	Implicit
HeapSort	$O(n \cdot \log n)$	$O(n \cdot \log n)$	Yes
SelectionSort	$O(n^2)$	$O(n)$	Yes
SearchTree	$O(n \cdot \log n)$	$O(n)$	No
[FG05]	$O(n \cdot \log n)$	$O(n)$	Yes

[G. Franceschini, V. Geffert, *An in-place sorting with* O(*n* log *n*) *comparisons and* O(*n*) *moves*, J.ACM, 52(4), 515-537, 2005]

Search trees

	Search & updates	Range searching	Implicit	Cache-oblivious
Red-black, ...	$O(\log n)$	$O(\log n + t)$	No	No
Sorted array (no updates)	O(log n)	$O(\log n + t)$	Yes	No
[FG02]	$O(\log n)$	$O(\log n + t)$	Yes	No
[FG03]	$O(\log n)$	-	Yes	Yes
[BFJ02], ...	$O(\log n)$	$O(\log n + t)$ am.	No	Yes

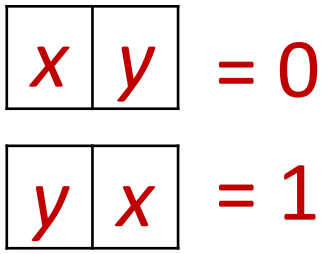
[G. Franceschini, R. Grossi, *Optimal Cache-Oblivious Implicit Dictionaries*, Proc. 30th International Colloquium on Automata, Languages, and Programming, volume 2719 of Lecture Notes in Computer Science, 316-331, Springer-Verlag, 2003.]

[G. Franceschini, R. Grossi, J.I. Munro, L. Pagli. *Implicit B-trees: New results for the dictionary problem*. IEEE Symposium on Foundations of Computer Science, 145-154, 2002]

[G.S. Brodal, R. Fagerberg, R. Jacob, *Cache-Oblivious Search Trees via Binary Trees of Small Height*, 13th Annual ACM-SIAM Symposium on Discrete Algorithms, 39-48, 2002]

The foundamental implicit trick

- The relative two elements $x < y$, can encode a bit

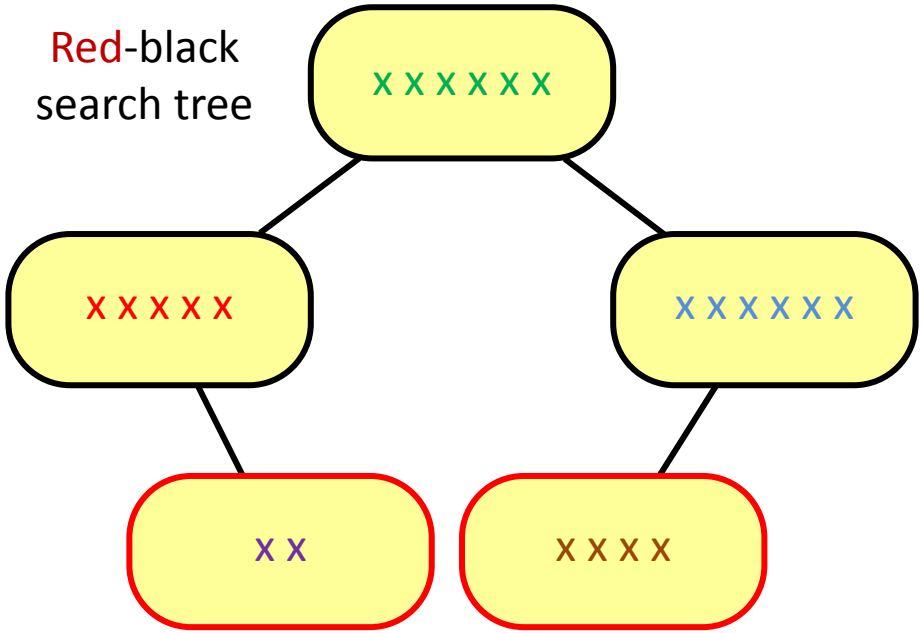


- $2\log n$ elements can encode integer $\{0,\ldots,n-1\}$

Search trees

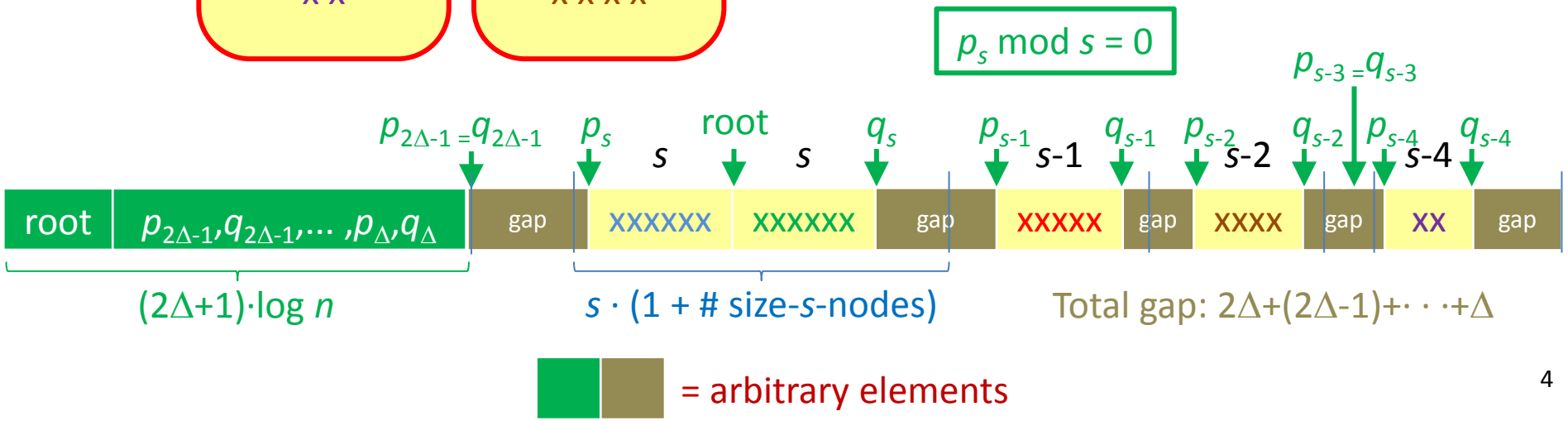
Search & updates	Range searching	Implicit
$O(\log^2 n)$	$O(\log^2 n + t)$	Yes

[J. Ian Munro, *An Implicit Data Structure Supporting Insertion, Deletion, and Search in* $O(\log^2 n)$ *Time,* Journal of Computer and System Sciences, 33(1), 66-74, 1986]



Each nodes stores $\Delta..2\Delta-1$ elements encoding the below fields ($\Delta=8\cdot\log n+2$)

field	values	encoded by #elements
left (address)	0,1,..., n-1	$2\cdot\log n$
right (address)	0,1,..., n-1	$2\cdot\log n$
parent(address)	0,1,..., n-1	$2\cdot\log n$
color (red/black)	0,1	2
node size s	0,1,...,n-1	$2\cdot\log n$



= arbitrary elements

Implicit merging O(*n* + *m*)

- can be used in an implicit O(*n*·log *n*) MergeSort