6.897
L 19.3
May 5, 2003

Stack implementation

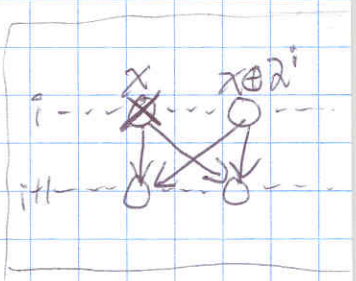
- secure storage stores in which column top layer starts
- Top is then trivial
- Pop: – remove Top element
  – follow Down pointer to update secure pointer
- Push: – add node in previous column mod s
  – add Down, Right, Diagonal pointers using secure pointers
  – update secure pointer in this column
- upon encounter of failed node, pause & invoke fault recovery

Recovery algorithm

– repeatedly pop nodes
– try to visit node via both incoming pointers from previous level – else lost ~ discard
– stop when s accessible nodes in a row
(accessible = not lost, not faulty)
– push nodes all back onto stack

Fault analysis

- suppose f faults seen
- claim $O(f \lg f)$ extra nodes get lost
- consider inaccessible node x in layer i
  – if $x \oplus 2^{i \bmod \lceil \lg s \rceil + 1}$ also inaccessible in layer i then both are inaccessible in next lower layer $i+1$
  – if $x \oplus 2^{i \bmod \lceil \lg s \rceil + 1}$ accessible in layer i then both are accessible in layer $i+1$



- can have $\le f \lg f$ pairs $\langle x, i \rangle$ such that x & $x \oplus 2^{i \bmod \lceil \lg s \rceil + 1}$ one both in set of faults (at any level)

provided $f \le s/2$ ⇒ remain in one butterfly
(at last layer, ≤ half nodes are inaccessible & none make it to next level (& none properly aligned))

⇒ $\le f \lg f$ lost nodes
⇒ $\le f(1 + \lg f)$ inaccessible nodes.