Register implementations out of faulty base registers

Prof R. Guerraoui

Distributed Programming Laboratory
Failure modes

- Responsive: once \( \perp \), forever \( \perp \)
- Non-responsive: no reply

\( t \) denotes the number of base objects that can fail

NB. In the asynchronous model, it is impossible to distinguish a non-responsive from a slow object
Algorithms

(1) Implements a SWMR register out of $t+1$ SWMR base responsive failure-prone registers

(2) Implements a SWSR register out of $2t+1$ SWSR base non-responsive failure-prone registers
Responsive model

Write(v)

For $j = 1$ to $(t+1)$ do

Reg[$j$].write(v);

return(ok)

Read()

For $j = t+1$ to 1 do

v := Reg[$j$].read();

if $v \neq \bot$ then return(v)
Non-responsive model

Init: seq := 1

Write(v)

w_seq := w_seq + 1;
For j = 1 to (2t+1) do:
Reg[j].write(w_seq, v);
« wait until a majority of oks are returned »
return(ok)
Non-responsive model

Init: \((sn, val) := (-1, \perp)\);

Read()

For \(j = 1\) to \((2t+1)\) do:

\((s, v) := \text{Reg}[j].\text{read}()\);

\((sn, val) := (s, v)\) with the highest \(s\) from majority, including \((sn, val)\)

return \((val)\)