7 TERMINATION DETECTION

why: so the result of a computation can be used / next stage

process: active | idle

message: basic | control

computation is terminated iff →

$\forall_i \; P_i = idle \;\wedge\; \forall_{ij} \; C_{ij} = \phi$

all processes idle — all channels empty

DISTRIBUTED SNAPSHOTS ALGORITHM

idea: last process to become idle causes global snapshot to be recorded.

each process uses scalar time and process id pair: $P_i \rightarrow (t, i)$

R1 an active process may send a basic msg to anyone

$P_i \xrightarrow{B(t)}$

R2 on recieving a basic msg, the process becomes active (and updates time)

$\xrightarrow{B(t')} P_i$

$P_i := active$

$t := t' + 1$

R3 when a process becomes idle, it broadcasts snapshot request to all process, including itself (& time upd)

$t := t + 1$ (& idle)

$P_i \xrightarrow{R(t,i)} \forall_j P_j$

take local snapshot by $R(t,i)$

R4 when a process recieves snapshot request, it accepts only if it is later than his own idle time (& time upd)

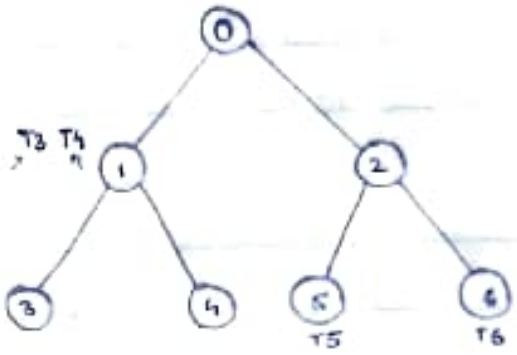
P_i = active ? ignore . $t = \max(t', t)$

$(t', i') \; \& \; (t, i)$? ignore

take local snapshot for $\xrightarrow{R(t',i')} P_i$

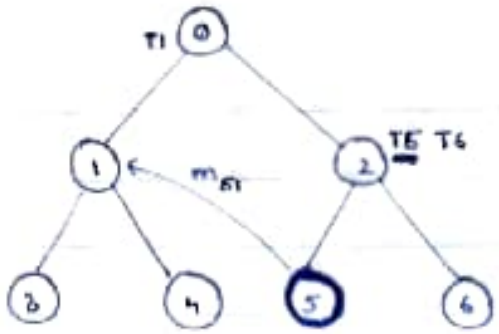
SPANNING TREE ALGORITHM

Processes are connected as fixed undirected graph. algorithm uses fixed spanning tree of graph.

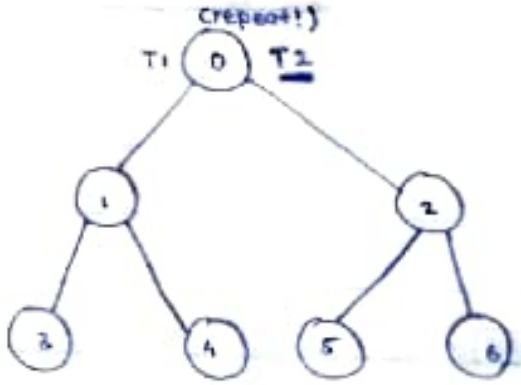


leaf - when idle, it reports to parent (token) (and includes colour rules)

node - when idle, and all children idle, it sends token to parent
- if it had sent a msg before being idle, it is colored black
- if it is black or recieved a black msg (from children) the token it sends is black
- it turns white after sending black token (to parent)



root - when idle, and recieves white token, global snapshot is taken
- if black/white token is recieved, instead, it sends a repeat token to all children and the whole process starts again



best case: $O(n)$

worst case: $O(n * M)$

n = no. of computation msgs exchanged

note: all process, tokens are initially white

- P5 turns black because it sends a basic msg to P1 (& P1 will reactivate)