

DHCP-201.cas.caltech.edu

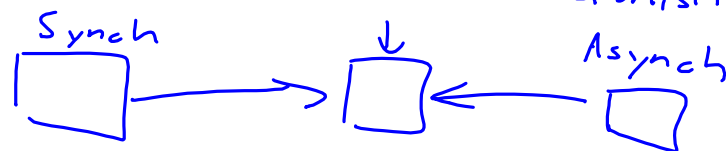
Lynch, Ch. 12

Goal: "Impossibility" Result

Conclusion: Some timing may be required

or

Settle for Probabilistic Success



Problem: Agreement

Fairness: one task/process

- processes take ∞ steps unless stopped

- Each user sends exactly one init.

Execution α is successful if:

① well-formedness

$trace(\alpha) \setminus U_i \in \{ \emptyset, init(v)_i, init(v)_i, decide(v)_i \}$

② agreement

all decisions are the same

③ Validity

if all inits are v , then all decides are v

④ Termination

(Diag. 12.1)

④ f-failure termination

if init on all ports
and $\leq f$ ports stopped
then unstopped
ports decide

④a Wait-free ($f=n$)

④b Failure-free ($f=0$)

④c Single-failure ($f=1$)

Restrict to Read/Write Memory

Enabling depends only on process state

Read_{ij}: $P_i := f(x_j)$

Write_{ij}: $x_j := f(P_i)$

(not) Modify: $(P_i, x_j) := f(P_i, x_j)$

Proof outline :

- Ⓐ Simplify assumptions, defns
- Ⓑ Impossibility for Wait-free alg
- Ⓒ " " Single-failure alg

Assumptions (WLOG)

- $V \in V = \{0, 1\}$
- users generate exactly one init
- determinism
- every non-failed process has exactly one ^{locally} enabled step
(add dummy read steps)

defn: initialization $init_1(v_1), init_2(v_2), \dots, init_n(v_n)$
 Assume all executions "input-first" begin with initialization

Valence

Finite execution α is:

0-valent: for α , all extensions
 all decisions are 0
 and some decision occurs

1-valent: same

univalent

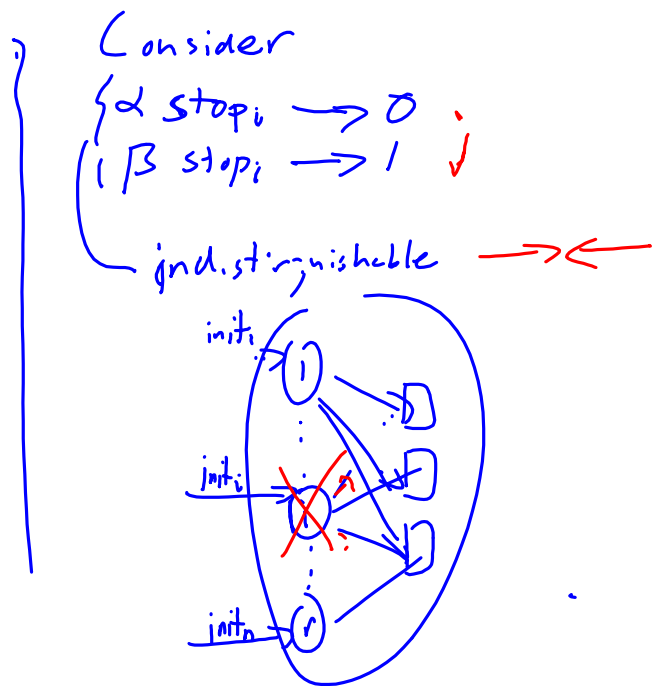
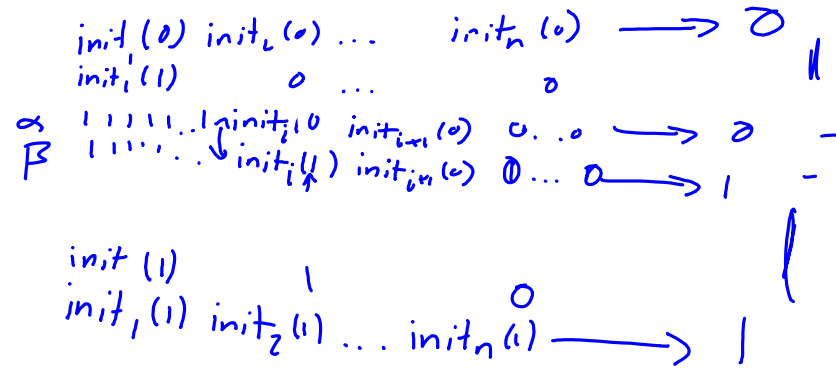
bivalent no midment
 has all {extensions}
 contain decisions and
 0 decisions

Lemma: For a working algorithm, exes are uni-or bi

Suppose A is working single-failure alg

Lemma: A has bivalent initialization

Suppose all initializations univalent



Defn: Finite, failure free exec α is a decider if

- α bivalent
- $\forall i$ α_i univalent
↖ ↗
└─ unique locally enabled step of P_i

Lemma: A is a working, wait-free alg, A has a decider exec α

Pf: Suppose no such α .
 Start with bivalent initialization α_0 .
 extend to infinite bivalent exec α .

$\exists i$ st. P_i takes ∞ steps

Consider α' : α with all P_i taking finite # of steps
 Stopped after last step

α : α 1 2 1 1 1 1 1 1 2 1 1 1 1 ...
 α' : 3 stops 2 1 1 1 1 2 stops 1 ...

α' is fair = α decides

α, α' are indistinguishable
 $\Rightarrow \alpha$ decides
 \Rightarrow at some point α must become univalent in any extension $\rightarrow \leftarrow$

Thm 1: No wait-free waiting alg.

Pf: Suppose A working wait-free alg.
 $\Rightarrow \exists$ decider exec α .

α bivalent, α_i 0-valent
 α_j 1-valent

Case 1: suppose i is a read step

indisting. $\begin{cases} \alpha_i \text{ stop } i \rightarrow 0 \\ \alpha_j \text{ stop } i \rightarrow 1 \end{cases} \quad X$

Case 2: j read, same.

Case 3a: i, j writes to different vars.

indis. $\begin{cases} \alpha_{ij} \rightarrow 0 \\ \alpha_{ji} \rightarrow 1 \end{cases} \quad X$

Case 3b: i, j write to some var

indis $\begin{cases} \alpha_{ij} \text{ stop } i \rightarrow 0 \\ \alpha_{ji} \text{ stop } i \rightarrow 1 \end{cases} \quad X$

$\therefore A$ does not exist.

With Read/Modify/Write:

Alg: each process waits for init
shared variable $x := \text{"unknown"}$

init(v) _{i} \Rightarrow 1) $P_i := v$

2) Modify:

$$(P_i, x) := \begin{cases} (P_i, P_i) & x \text{ "unknown" } \\ (x, x) & \text{otherwise} \end{cases}$$

3) decide _{i} (P_i)