

FORMAL METHODS IN NETWORKING
COMPUTER SCIENCE 598D, SPRING 2010
PRINCETON UNIVERSITY

LIGHTWEIGHT MODELING
IN PROMELA/SPIN AND ALLOY

Pamela Zave

AT&T Laboratories—Research

Florham Park, New Jersey, USA

THE PRESS RELEASE

"Three features that distinguish Chord from many peer-to-peer lookup protocols are its simplicity, provable correctness, and provable performance."

THE (NEWLY DISCOVERED) REALITY

- the only "proof" covers the join-and-stabilize case only, with no failures
- this "proof" is an informal construction of ill-defined terms, unstated assumptions, and unjustified or incomprehensible steps

however, the subset can be proven correct, formally

- the full protocol is incorrect, even after bugs with straightforward fixes are eliminated
- *not one* of the six properties claimed invariant for the full protocol is invariantly true
- some of the many papers analyzing Chord performance are based on false assumptions about how the protocol works

USE

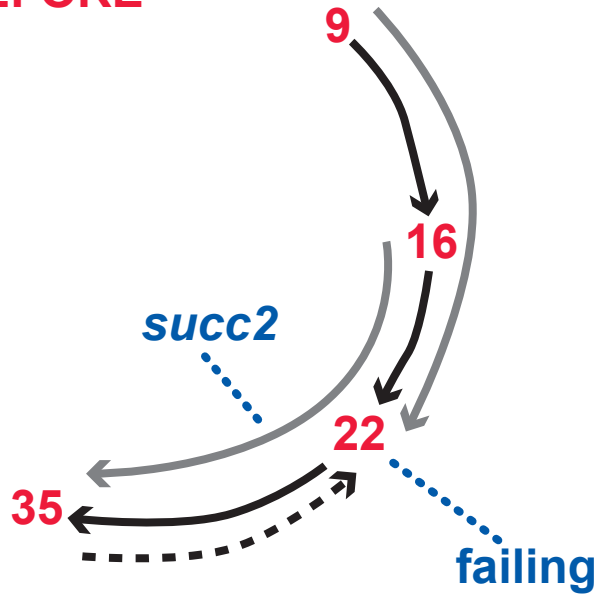
LIGHTWEIGHT MODELING

and avoid embarrassment!

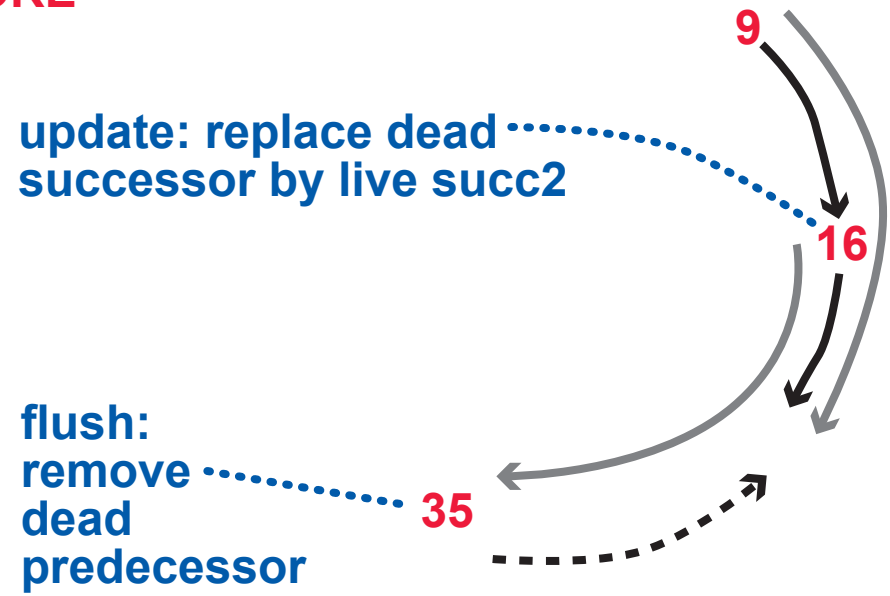
THE FAIL EVENT

THE RECONCILIATION OPERATION

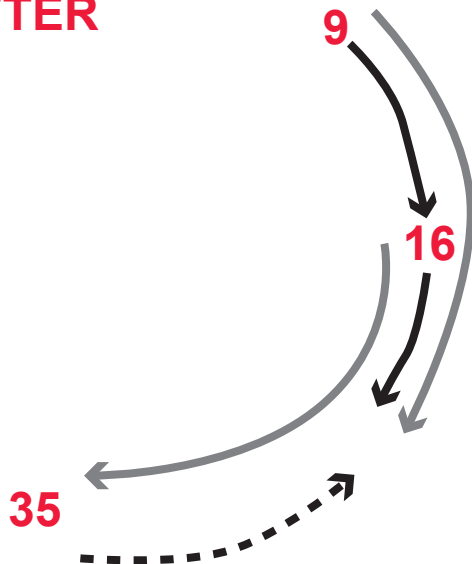
BEFORE



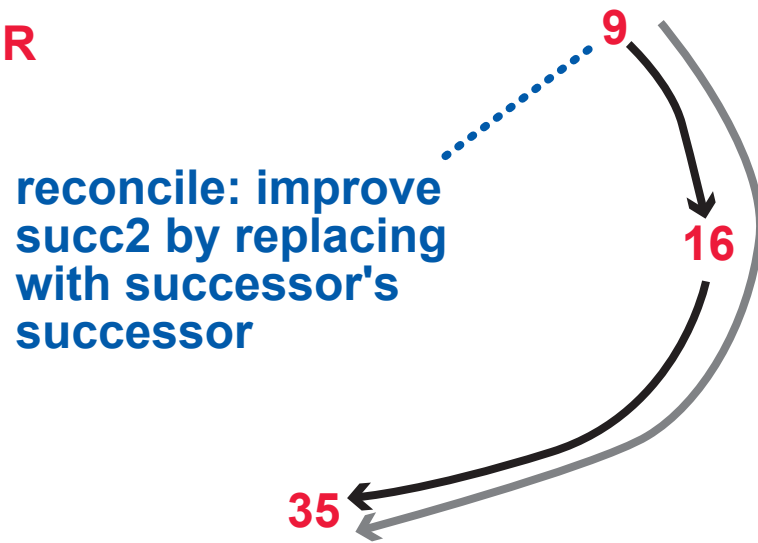
BEFORE



AFTER



AFTER



ANTECEDENT PREDECESSORS

```
pred AntecedentPredecessors [t: Time] {  
  all n: Node | let antes = (succ.t).n |  
    n.prdc.t in antes  
}
```

at time t, the set of all nodes whose successor is n

WHERE DID IT COME FROM?

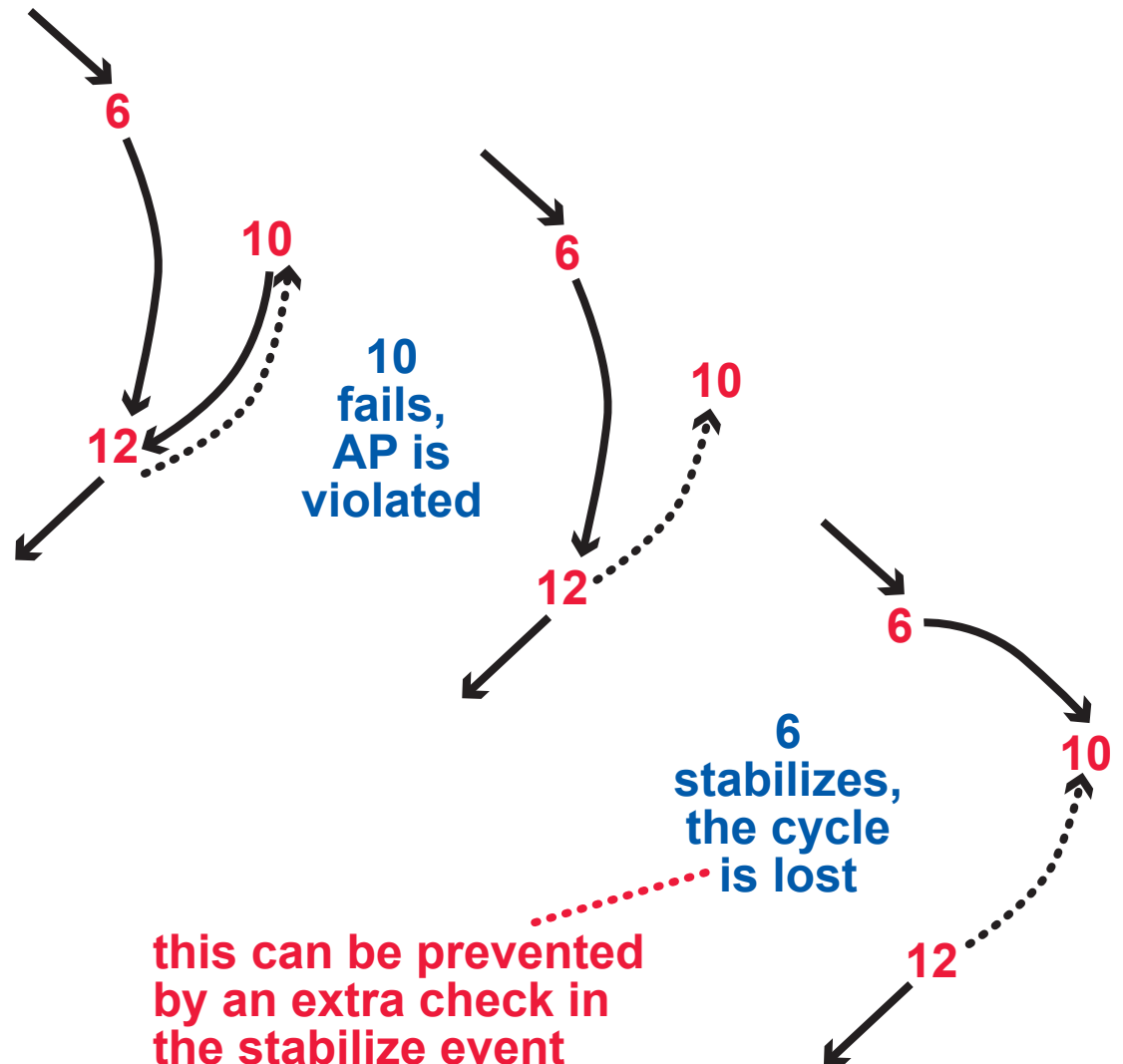
must be an invariant to prove that the pure-join model is correct

WAS IT PREVIOUSLY KNOWN?

no, supporting my allegation that the previous "proof" is useless

IS IT GOOD FOR ANYTHING ELSE?

yes, it enables us to diagnose and fix a Chord bug



PROBLEMS OF SMALL RINGS

every Chord implementation has a length r of the successor list kept by each node

in the Alloy model, $r = 2$

definition: a ring is small if it has r or fewer members

small rings have pathological behavior

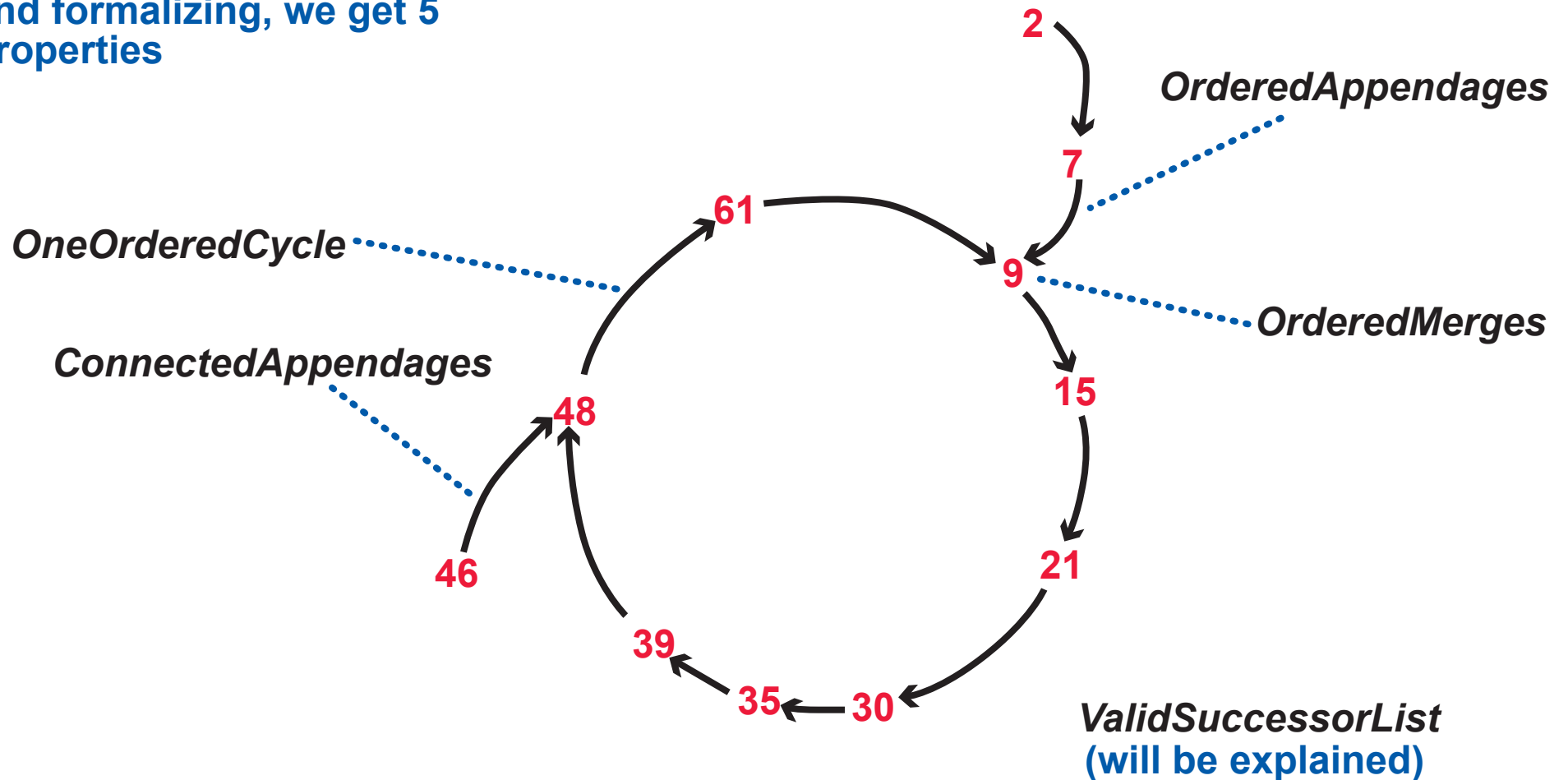
small rings are important because all rings start small

the problems can be fixed by treating small rings as a special case

none of this was known previously

PROPERTIES CLAIMED INVARIANT FOR THE FULL MODEL

after untangling bad definitions and formalizing, we get 5 properties



NOT ONE of these properties is actually an invariant!

ORDERED MERGES

```

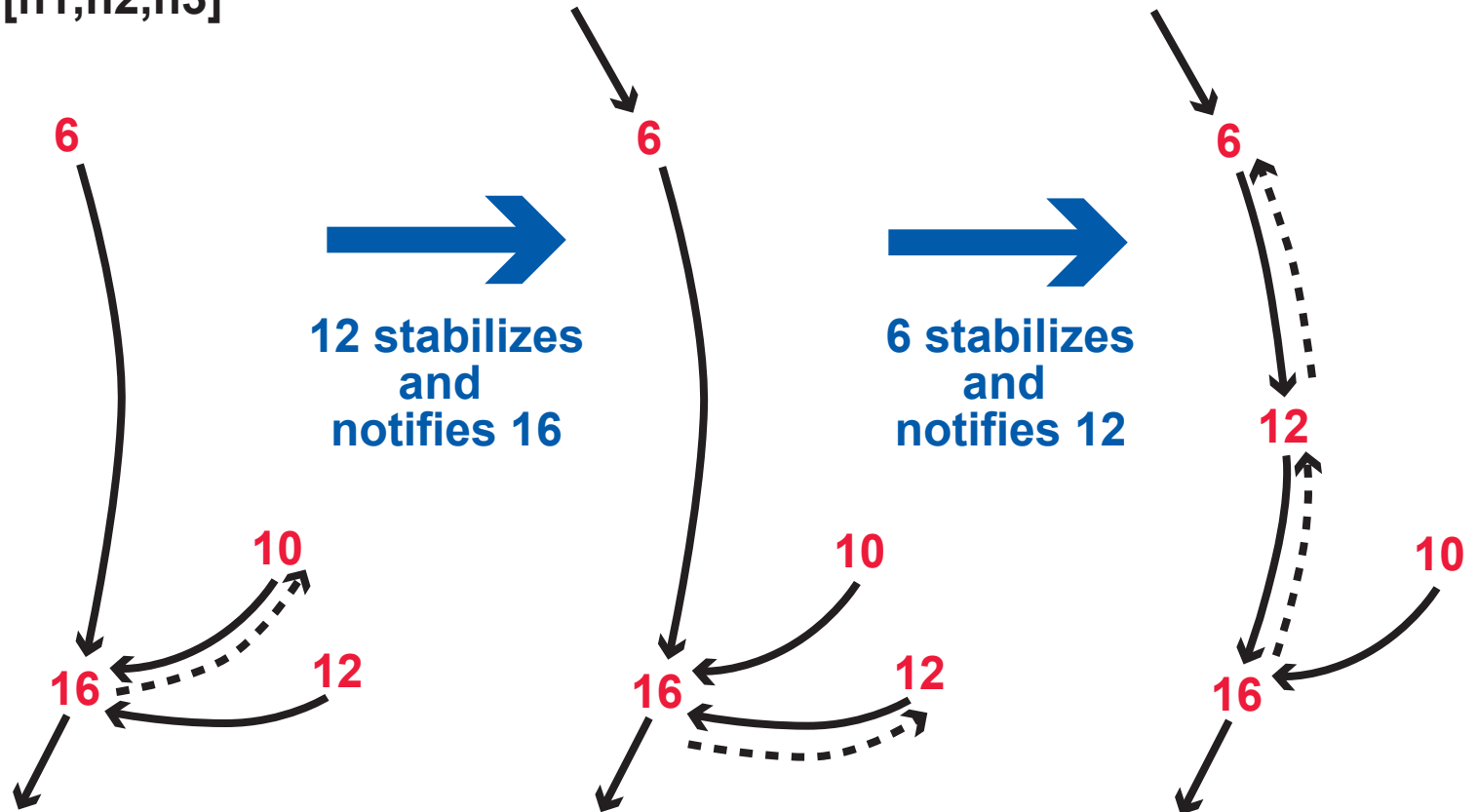
pred OrderedMerges [t: Time] {
  let cycleMembers =
    { n: Node | n in n.(^(bestSucc.t)) } |
    all disj n1, n2, n3: Node |
      ( n3 in n1.bestSucc.t
        && n3 in n2.bestSucc.t
        && n1 in cycleMembers
        && n2 !in cycleMembers
        && n3 in cycleMembers
      ) => Between[n1,n2,n3]
}
  
```

best live successor

The good news:
Violations are repaired by stabilization.

The bad news:
Compromises some lookups.
Invalidates some assumptions used in performance analysis.

easily violated, even in the pure-join model



ORDERED APPENDAGES

WHY A POWERFUL ASSERTION LANGUAGE IS NEEDED

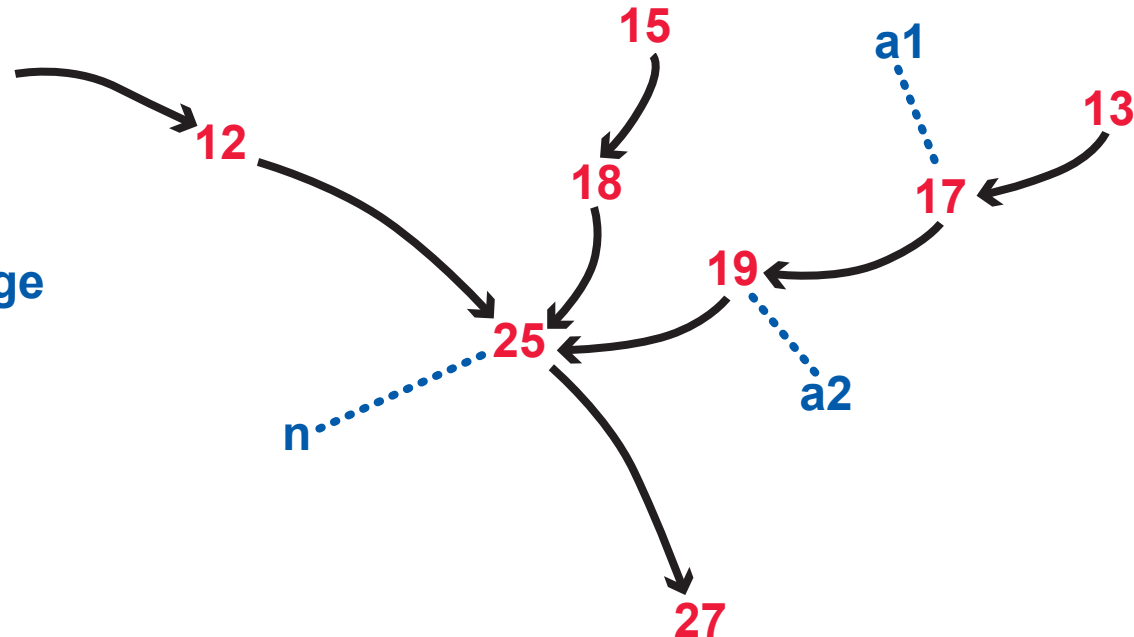
```
pred OrderedAppendages [t: Time] {  
  let members = { n: Node | Member[n,t] } |  
  let cycleMembers = { n: members | n in n.^(bestSucc.t) } |  
  let appendSucc = bestSucc.t - (cycleMembers -> Node) |  
  all n: cycleMembers |  
    all disj a1, a2, a3: (members - cycleMembers) + n |  
      ( n in a1.^appendSucc  
        && a2 = a1.appendSucc  
        && (a1 in a3.^appendSucc || a3 in a2.^appendSucc) )  
      ) => ! Between[a1,a3,a2]  
}
```

the successor
relation on
appendages only

a1, a2, a3 have to be
confined to the appendage
tree rooted at n

a3 can be 13

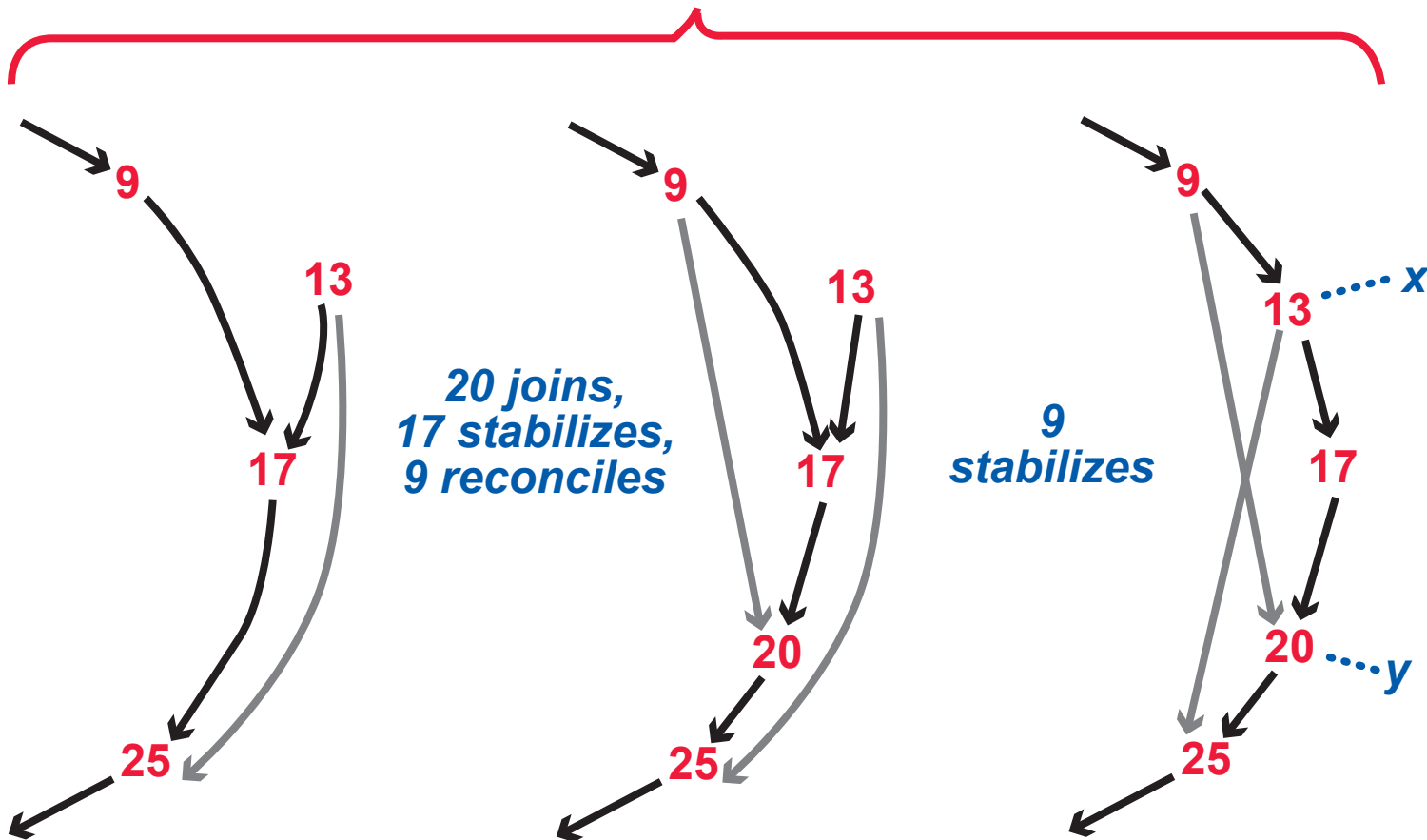
a3 cannot be 18



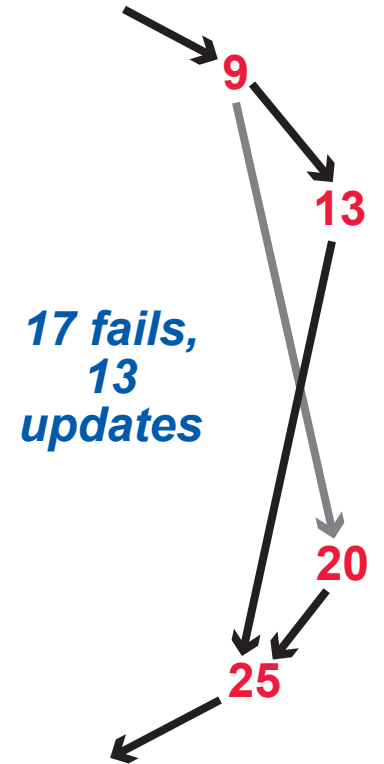
VALID SUCCESSOR LIST

"if a node x's successors skip over a live node y, then y is not in the successor list of any x antecedent"

how it can be violated



why it matters

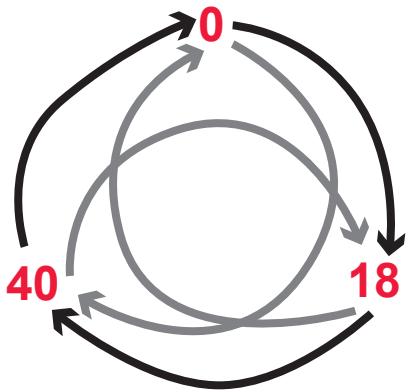


20 was part of the cycle, is now an appendage

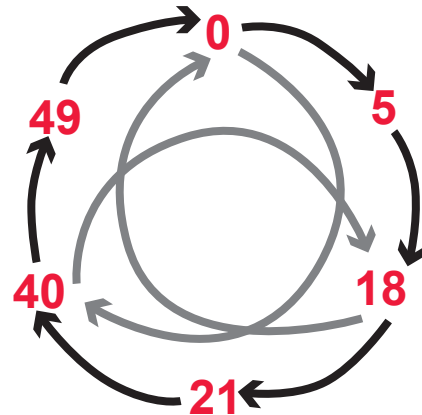
WHY THE FULL PROTOCOL IS NOT CORRECT

DESIRED THEOREM: In any reachable state, if there are no subsequent joins or failures, then eventually the network will become ideal and remain ideal.

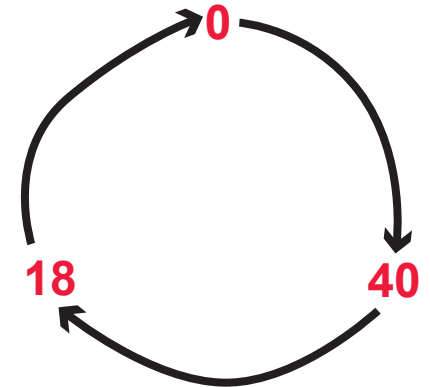
this ring is ideal



3 nodes
join and
become
integrated



new nodes
fail, old
nodes
update



this ring is
disordered, so
the protocol
cannot fix it

this is actually a class of counterexamples:

- any ring of odd size becomes disordered
- any ring of even size splits into two disconnected subnetworks (which the protocol cannot fix)

COMPARISON, REVISITED

PROMELA/SPIN

state
structure

primitive in Promela;
displayed poorly by Spin

invariants

except for the most basic
ones, an invariant must be
written as a C program

sometimes searching for
the right invariant requires
a great deal of trial and
error—this is why C
programs don't make
good invariants

ALLOY

Alloy language is rich and
expressive; many display options

Alloy language is rich,
expressive, and concise

these are not superficial
properties—they cannot
be slapped on top of Spin
like frosting on a cake