Life in 1-consensus

Yehuda Afek,
Eli Daian, Eli Gafni, Giuliano Losa

Classifying Deterministic Shared Memory Objects

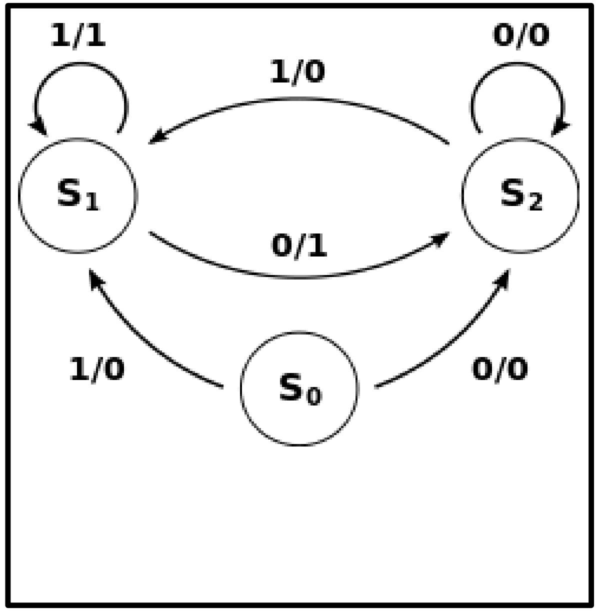
- Computability = the **power to solve tasks wait-free**

  = non-blocking

Deterministic Objects

- **Linearizable concurrent state machine**

- Atomic R/W Reg's
- Stack, Queue, F&Add, …
- CAS,



deterministic object = DFSA

Task (deterministic and non-det.)

Input-vector → Output-vector function:

(0,2,5,14,1) → {(2,5,5,2,5),(2,2,2,2,2)…..}

…..

…..

- Consensus (agreement & validity)
- Set-Consensus (set-agreement)
- Immediate Snapshot

Classifying Deterministic Shared Memory Objects

Computability = the **power to solve tasks wait-free**

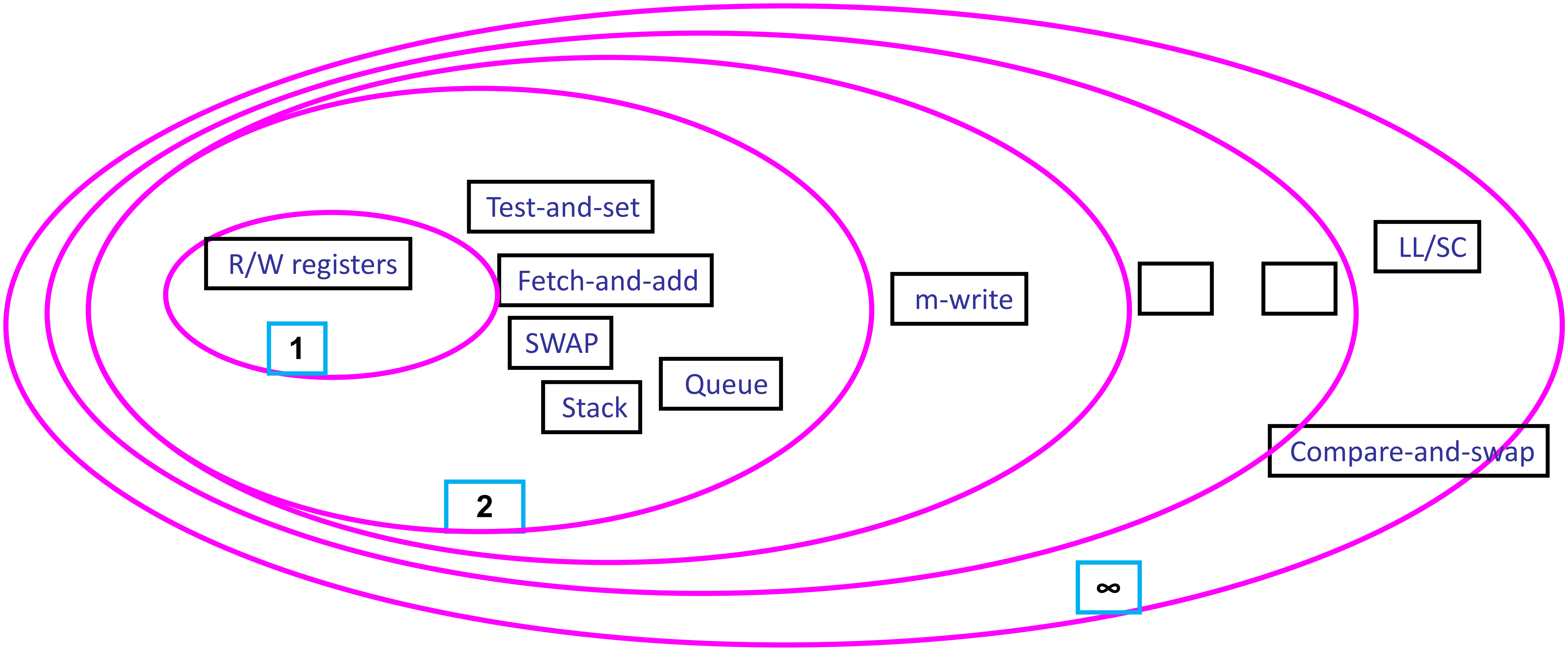
= non-blocking

- Standard asynchronous shared memory
- Shared **deterministic** objects
- Interested (only) in **wait-free** linearizable implementations of tasks == **non-blocking**!

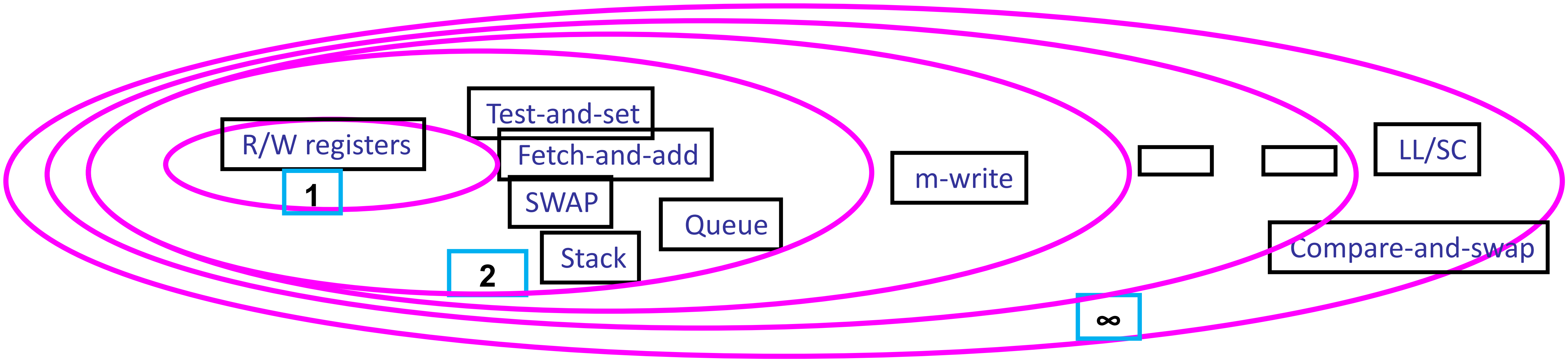
Object O Consensus Number

Cons#(O) = C=Max number of processors that can do w-f consensus with any number of O and Reg’s

Herlihy’s Consensus Hierarchy



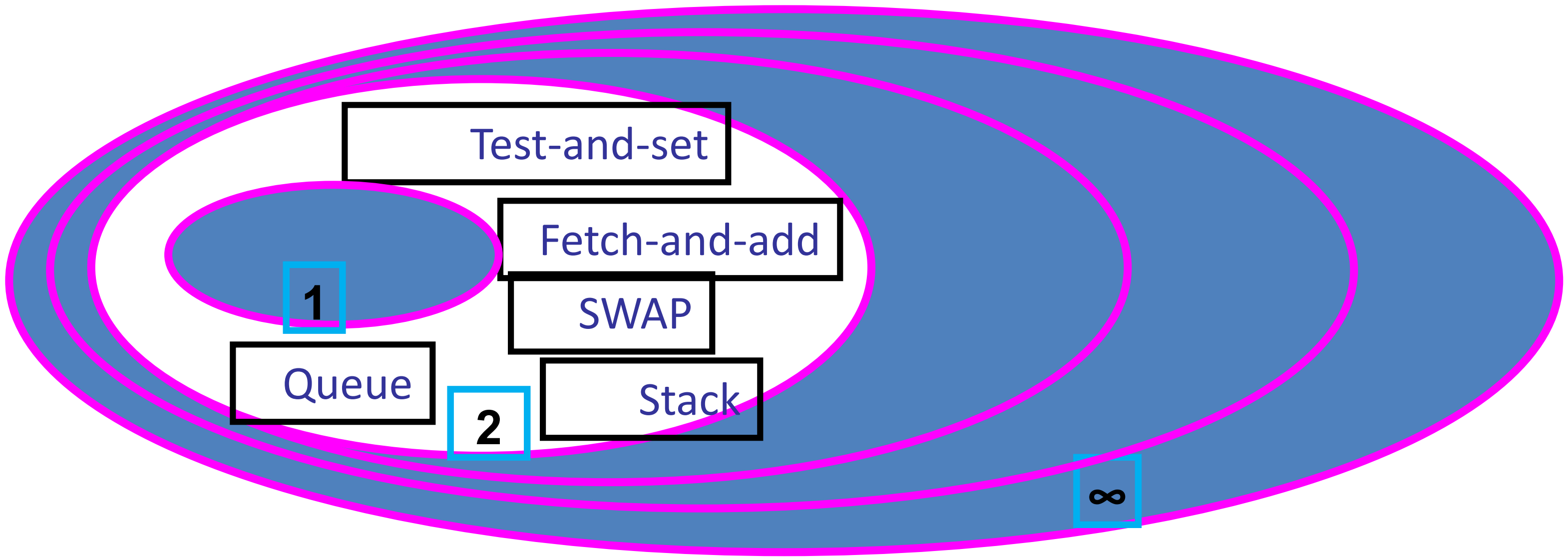
Herlihy's Consensus Hierarchy



- If cons#(O1)<cons#(O2) ➔ O1 **cannot** implement O2
- Object in L_k is universal for ***k* processes** (implements any O)
- Is level *k* complete? different objects in L_k can implement each other in a system with **n**>k processes?

Common 2 [AWeisbergerWeisman 1994]

- Test&Set from 2-cons
  – Yes
- Fetc&Add from 2-cons
  – Yes
- SWAP from 2-cons
  – Yes
- Stack from 2-cons
  – Yes
- Queue from 2-cons
  – ??



AEllenGafni 2016

- Common 2
- Common 3
-
-
- Common m
- Consensus Hierarchy Conjecture: Deteministic Obj of consensus number n can be implemented from n-consensus for any number of processors.

FALSE

Set-Consensus

Soma Chaudhuri 1990:

(n,k)Set-Consensus:

Agreement – output-vector is a set of at most k different values

Validity – each value is an input of some process

[HS, BG, SZ 1999] Atomic R/W cannot w-f solve (n,k)set-consensus, n>k>1

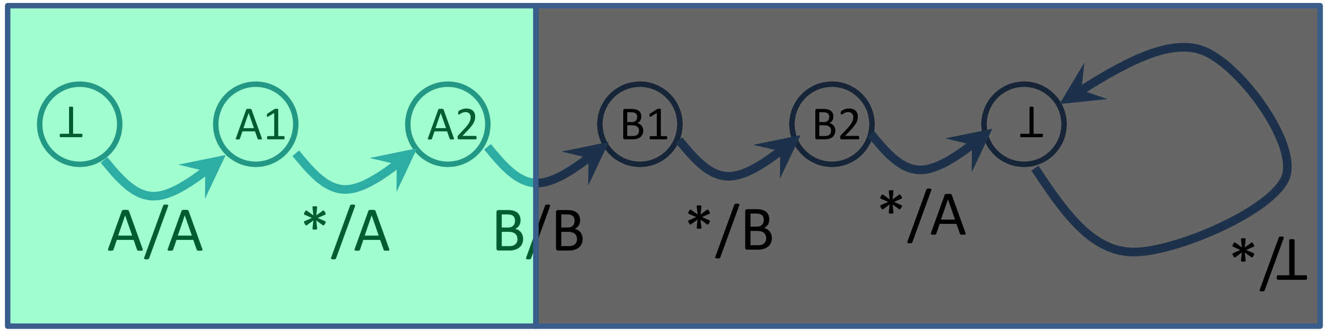
Borowsky Gafni 1993

- Roughly, $n>l$ and $k>j$: (l,j)set-consensus implements (n,k)set-consensus if $k=j*n/l$ (**use n/l copies of (l,j)set-consensus)**

- More precisely (dealing with remainders): ....if and only if
- $k \geq j$,
- $n/k \leq m/j$, and
- **either** $k \geq j \lceil n/m \rceil$ **or** $k \geq j \lfloor n/m \rfloor + n - m \lfloor n/m \rfloor$.

AEllenGafni 2016

- AABBA



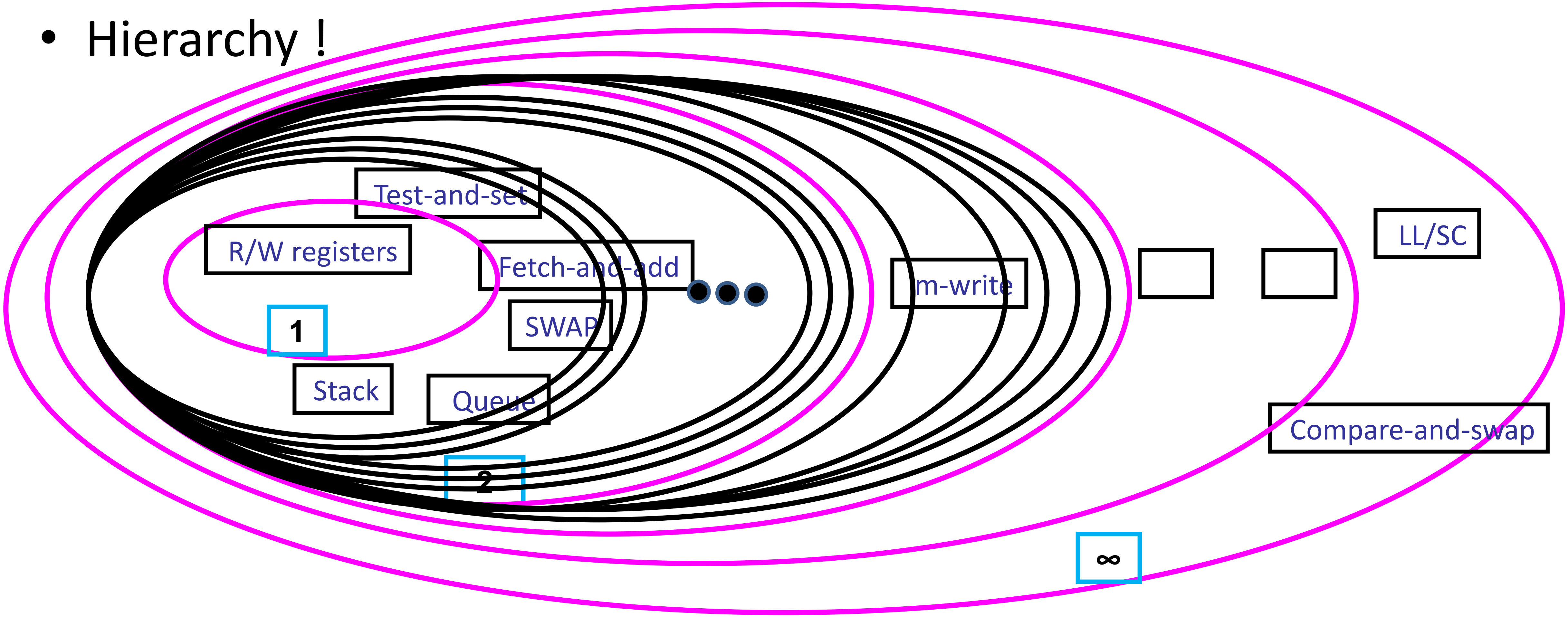
  - cons#(AABBA)=2=(2,1)sc
  - Set-consensus Power (AABBA) = (5,2)
  - BG➔(2,1)set-consensus cannot do (5,2)set-consensus
  - FLP (bivalent) proof cons#(AABBA)<3, =2 !
  - ➔counter example for common2 !!

Common2 conjecture

FALSE

AEllenGafni 2016

- AABBCCBA (= (8,3)) implements AABBA
- AABBA does not implement AABBCCBA
- Hierarchy !



Comparing the Hierarchies

Herlihy's Hierarchy

- Uses consensus
  - **Deterministic** task
- Classifies **deterministic** objects

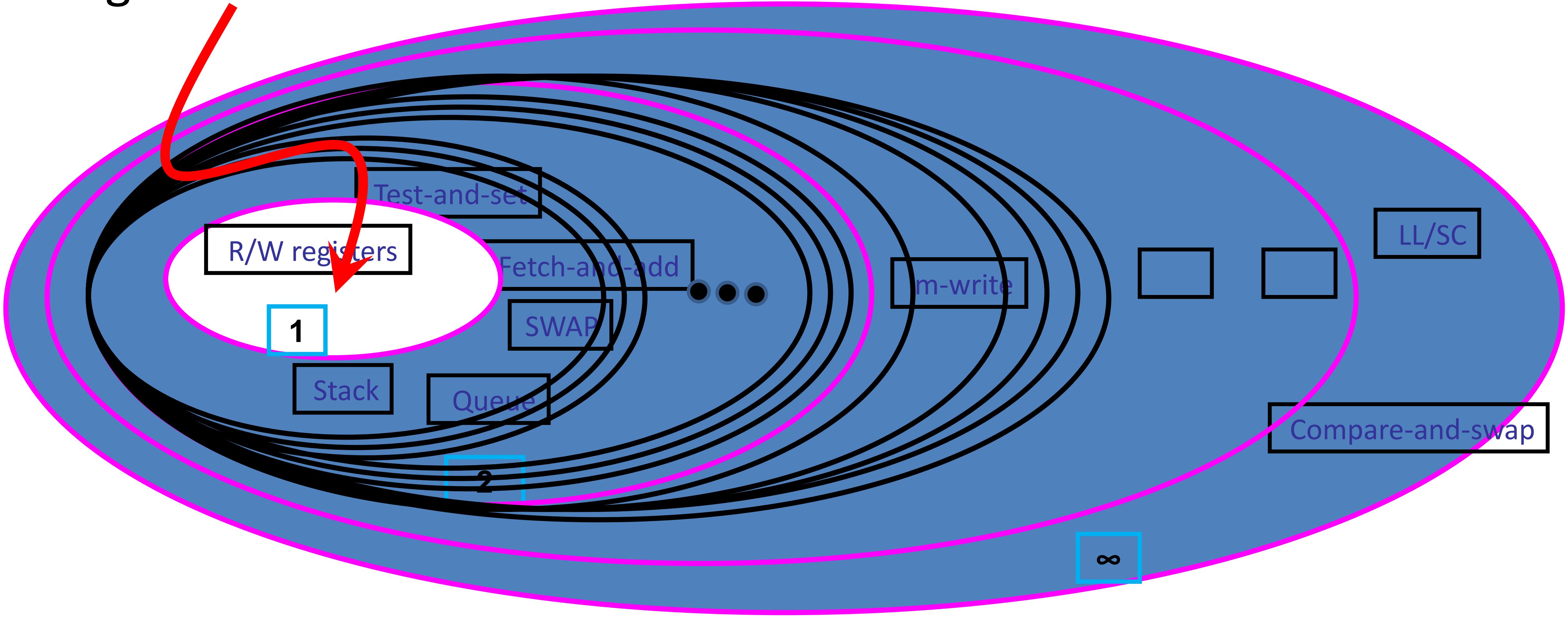
AEG

- Uses set consensus
  - **Nondeterministic** task
- Classifies **deterministic** objects

Remains Open [AEG cojecture]

Are there deterministic objects O s.t.,:

Reg’s < O < 2-consensus ??

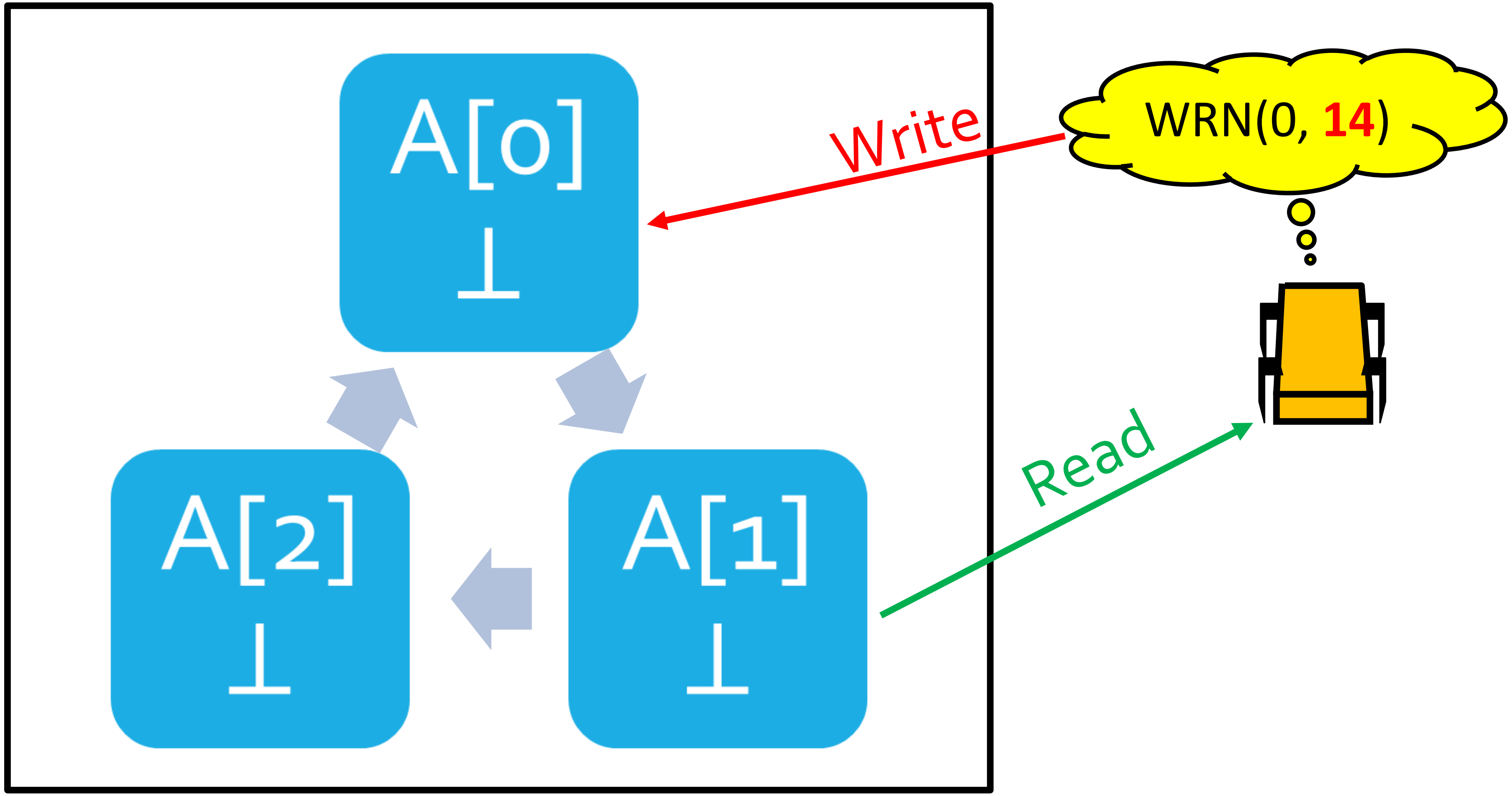


WRN_k OBJECTS

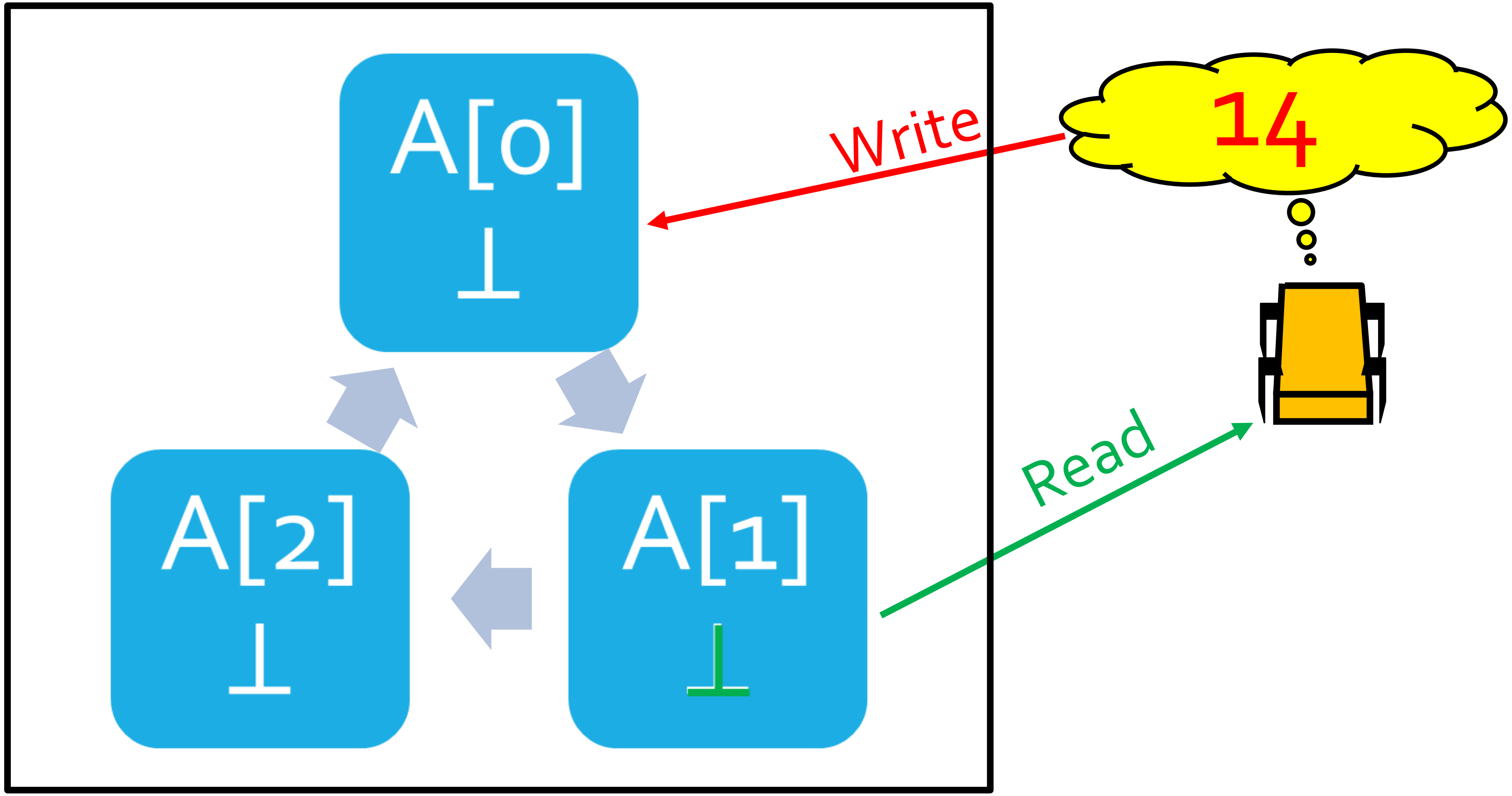
- A[0], A[1], … , A[k-1]   Registers, initially ⊥

- WRN(i,v):
  - Write v → A[i]
  - Return A[(i+1) mod k]

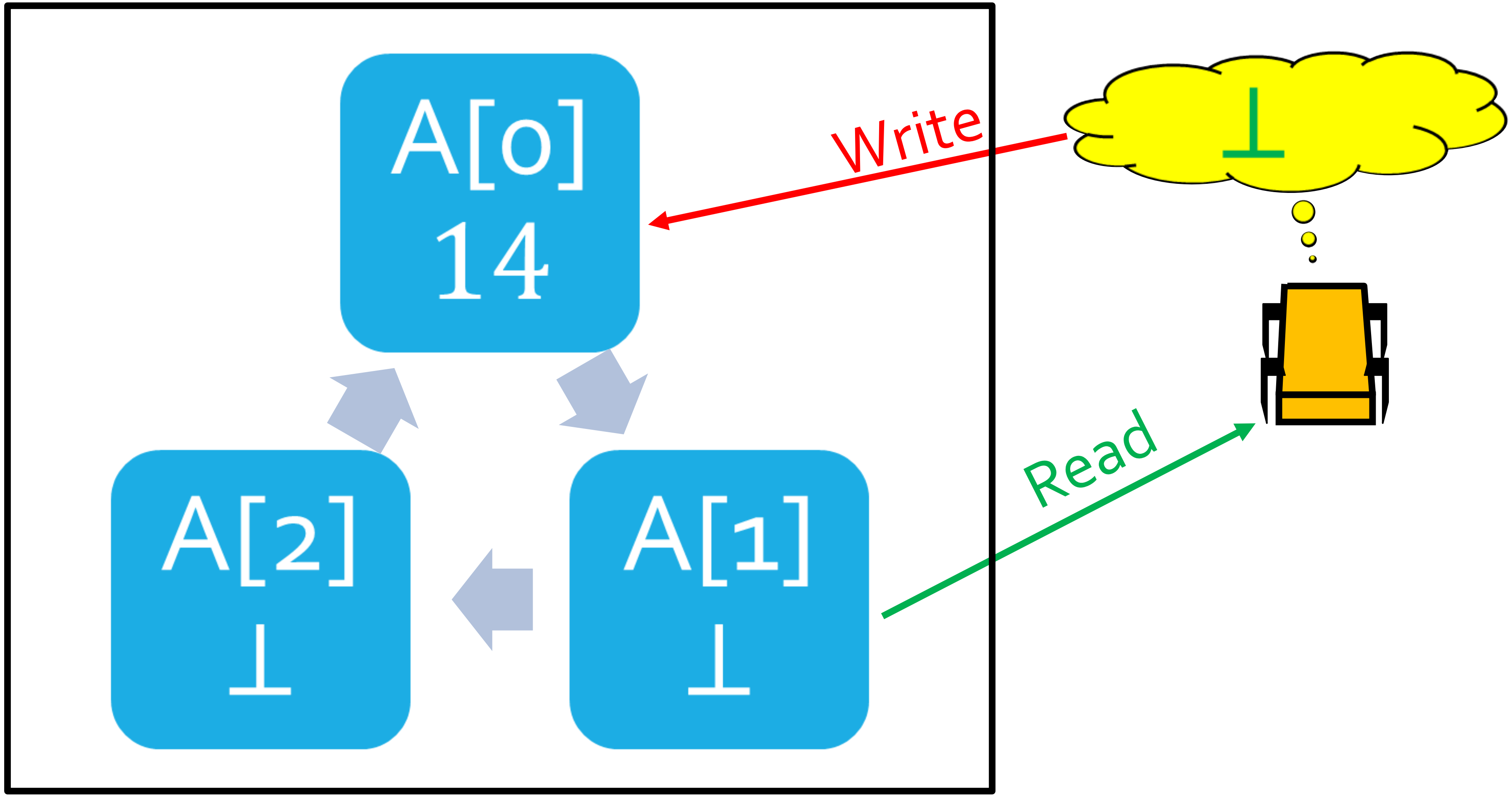
Example of WRN_3 Objects



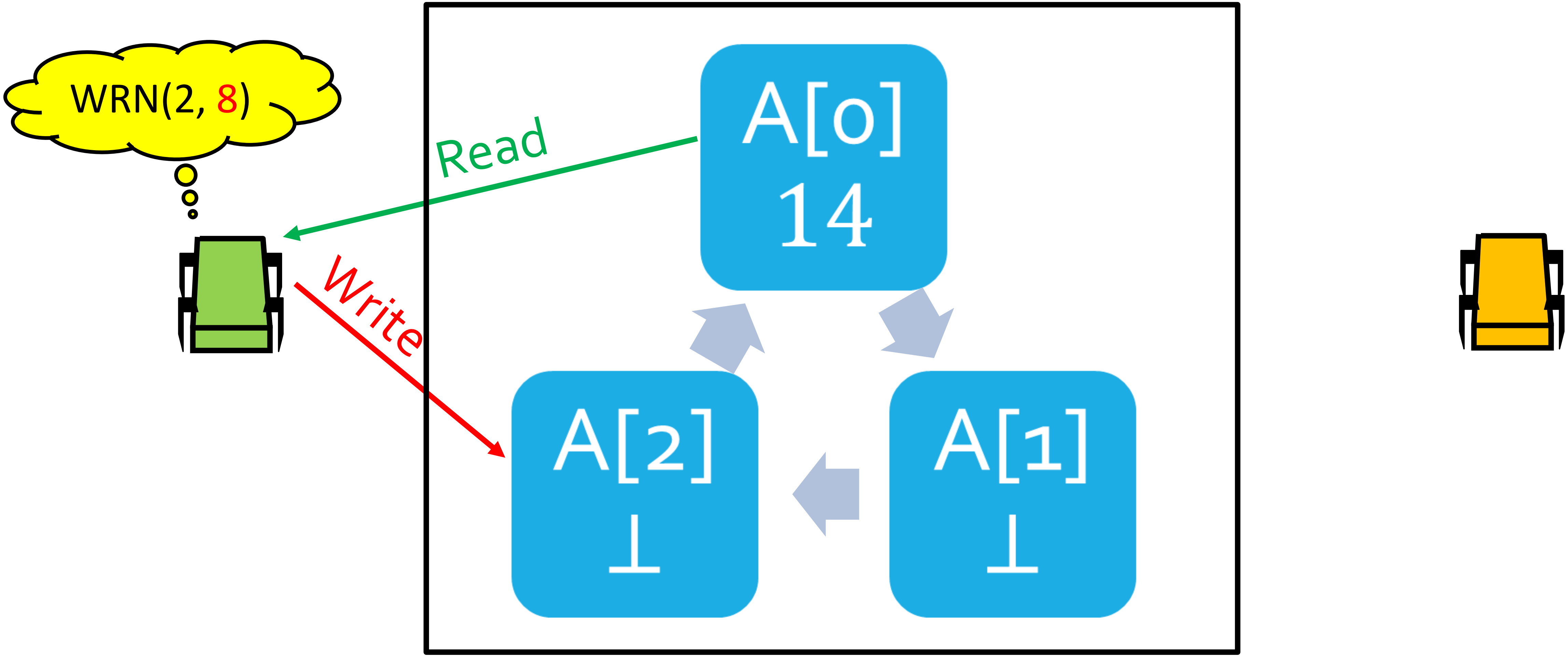
Example of WRN_3 Objects



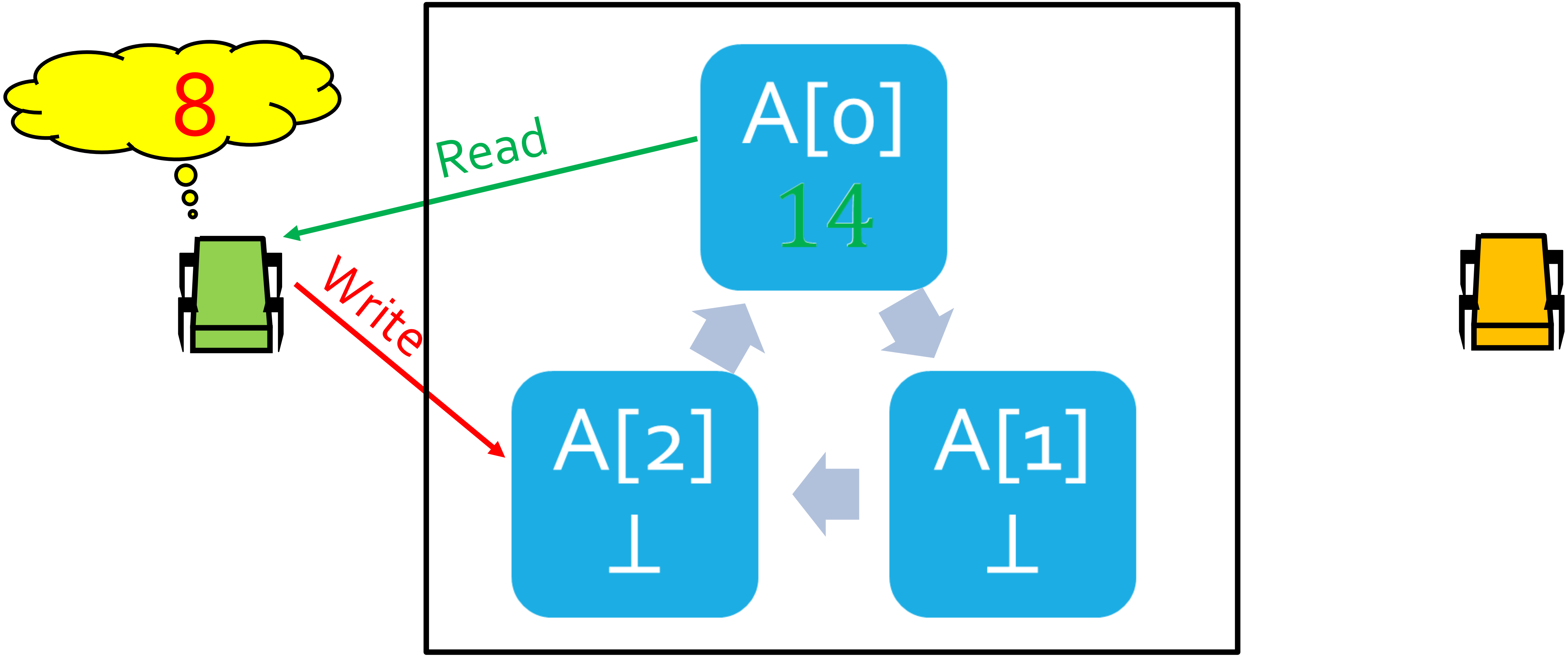
Example of WRN_3 Objects



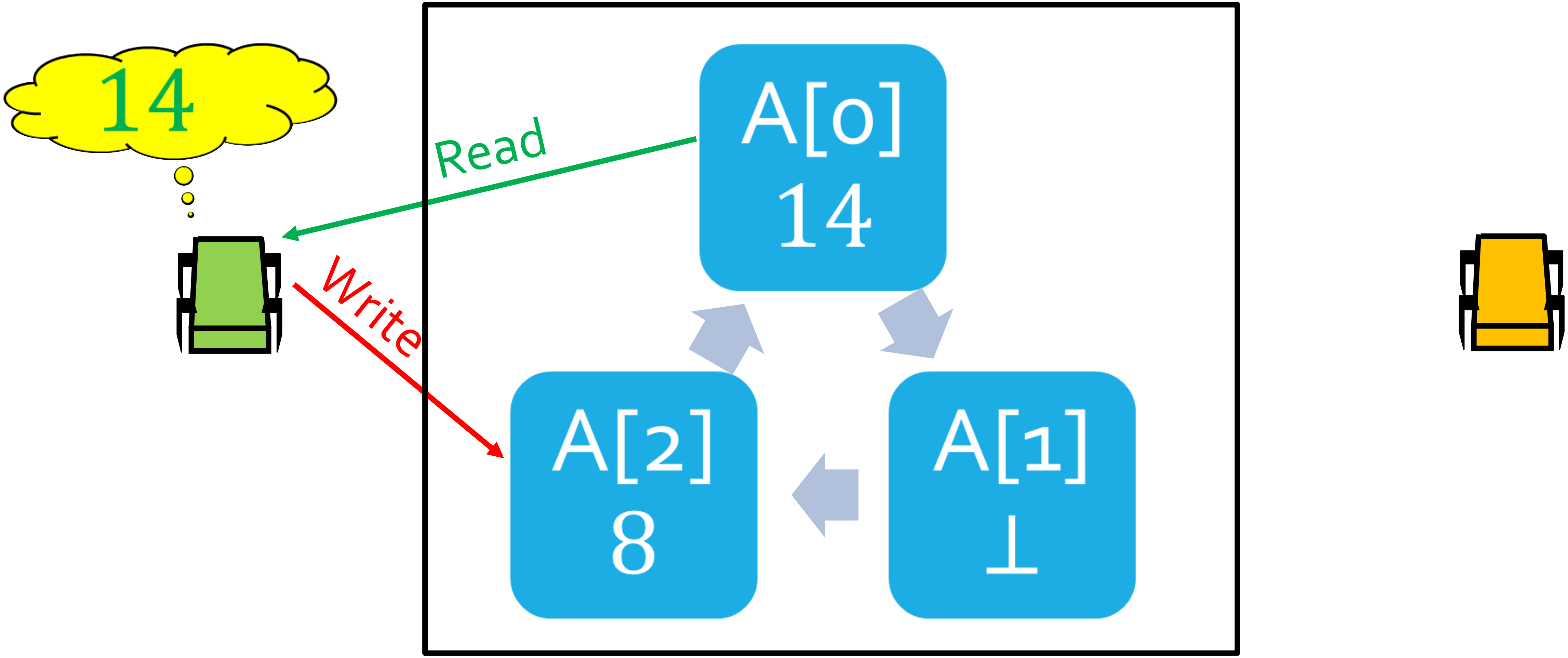
Example of WRN_3 Objects



Example of WRN_3 Objects



Example of WRN_3 Objects



Want to Prove that WRN_k is:

- ❑ Stronger than registers
- ❑ Weaker than 2-consensus

Registers < WRN_k < 2-consensus

WRN_k is Stronger than Registers

- Theorem:
  - Set consensus cannot be solved using registers [HS, SZ, BG 1993]

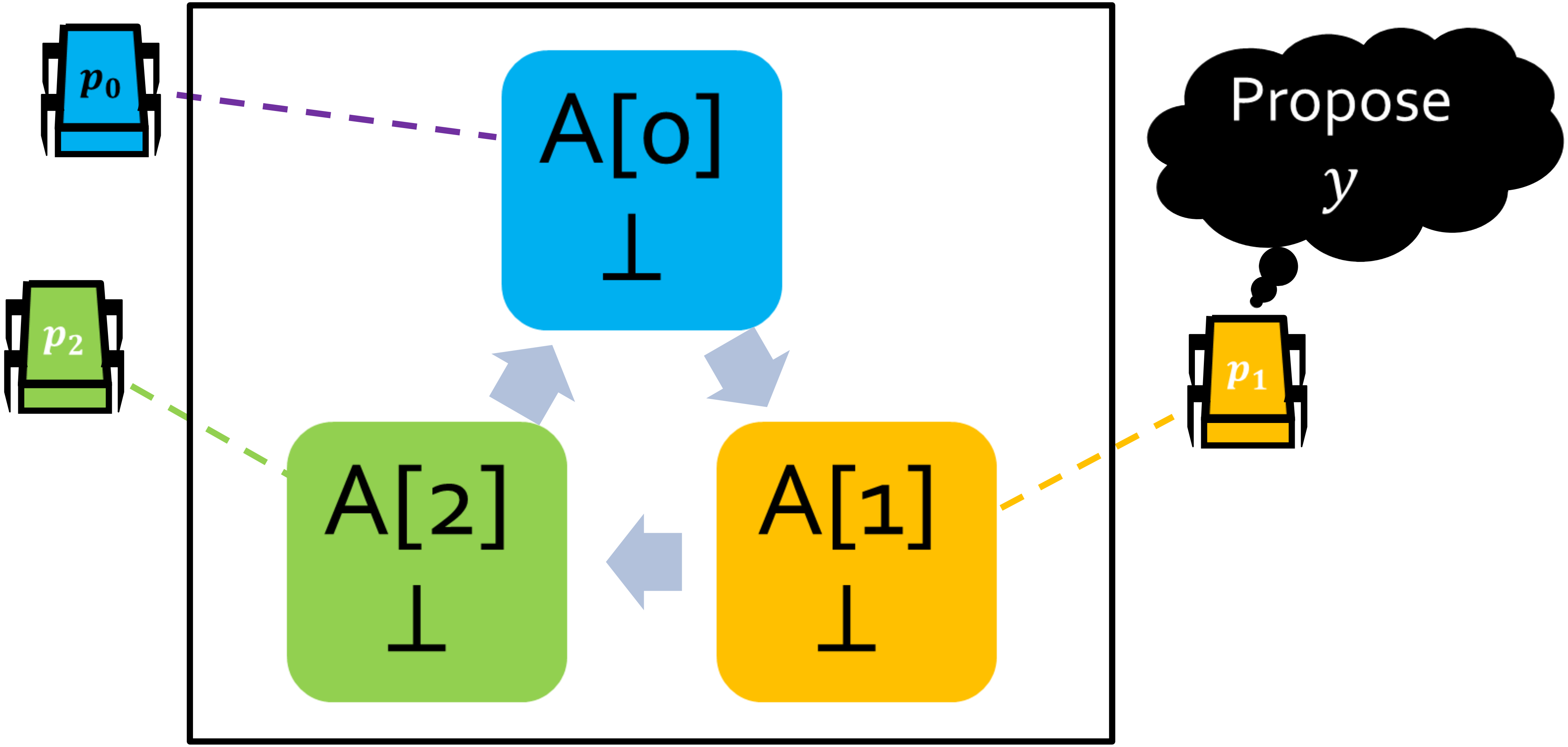
- Idea:
  - Solve set consensus using WRN_k

$(k, k-1)$-Set Consensus using WRN_k

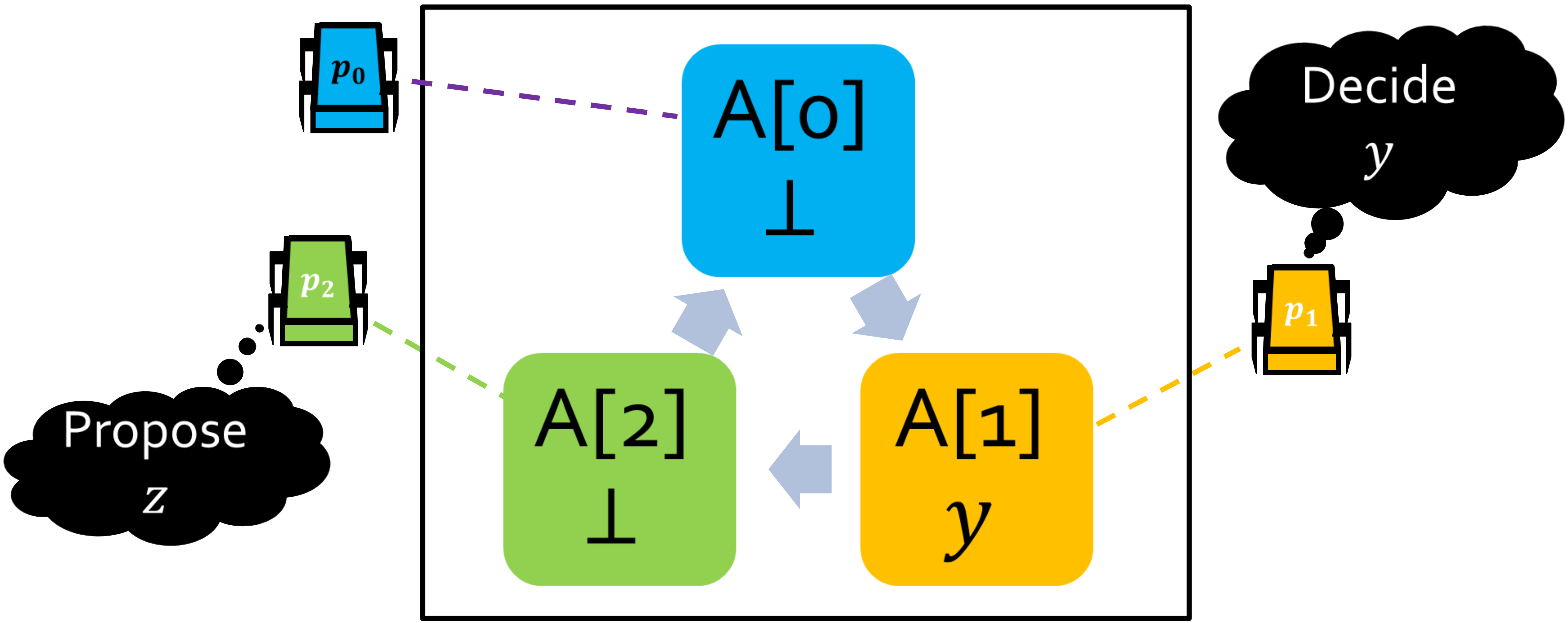
- Each process assigned with index

- Uses WRN_k with that index

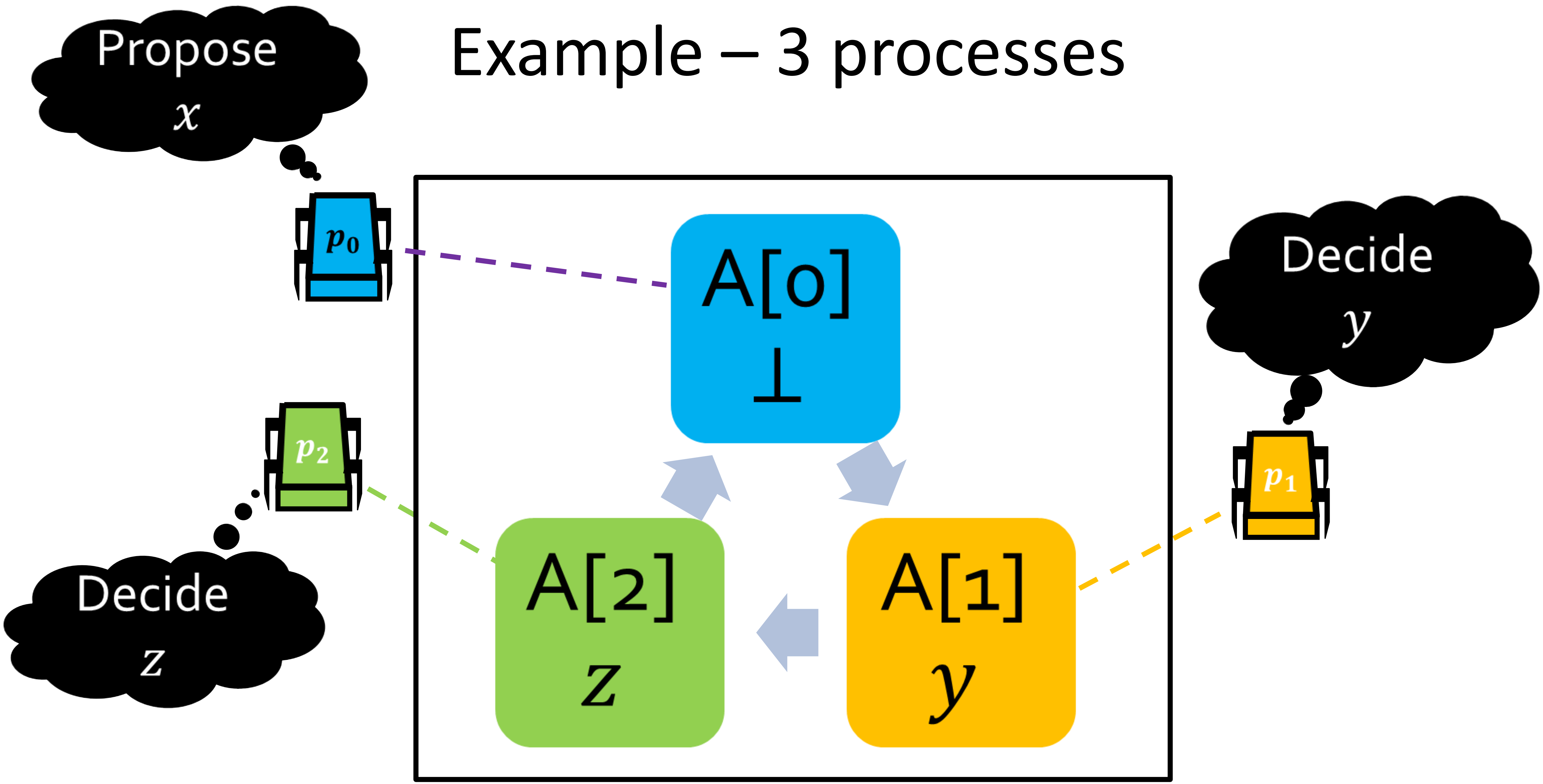
Example – 3 processes



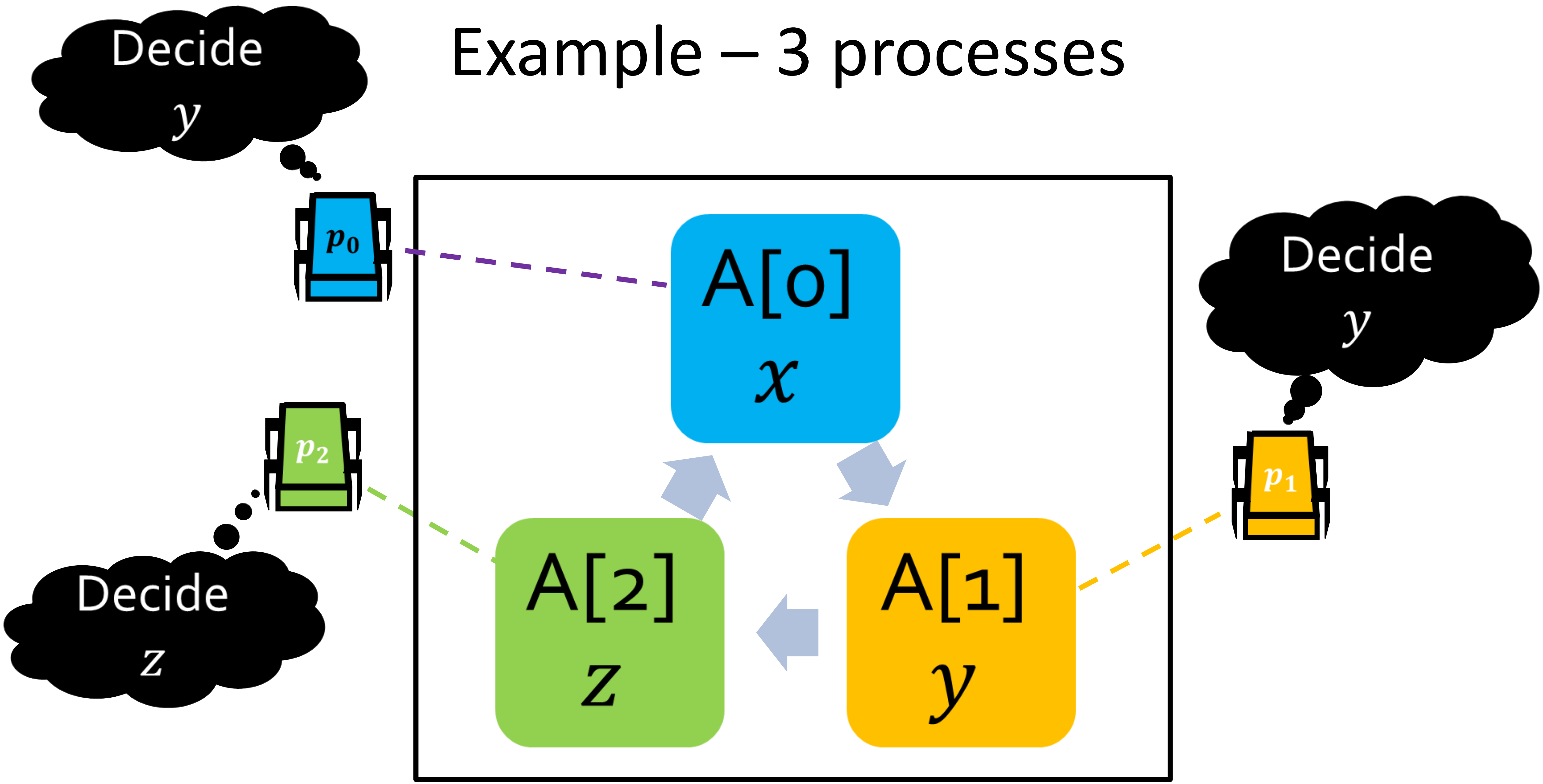
Example – 3 processes



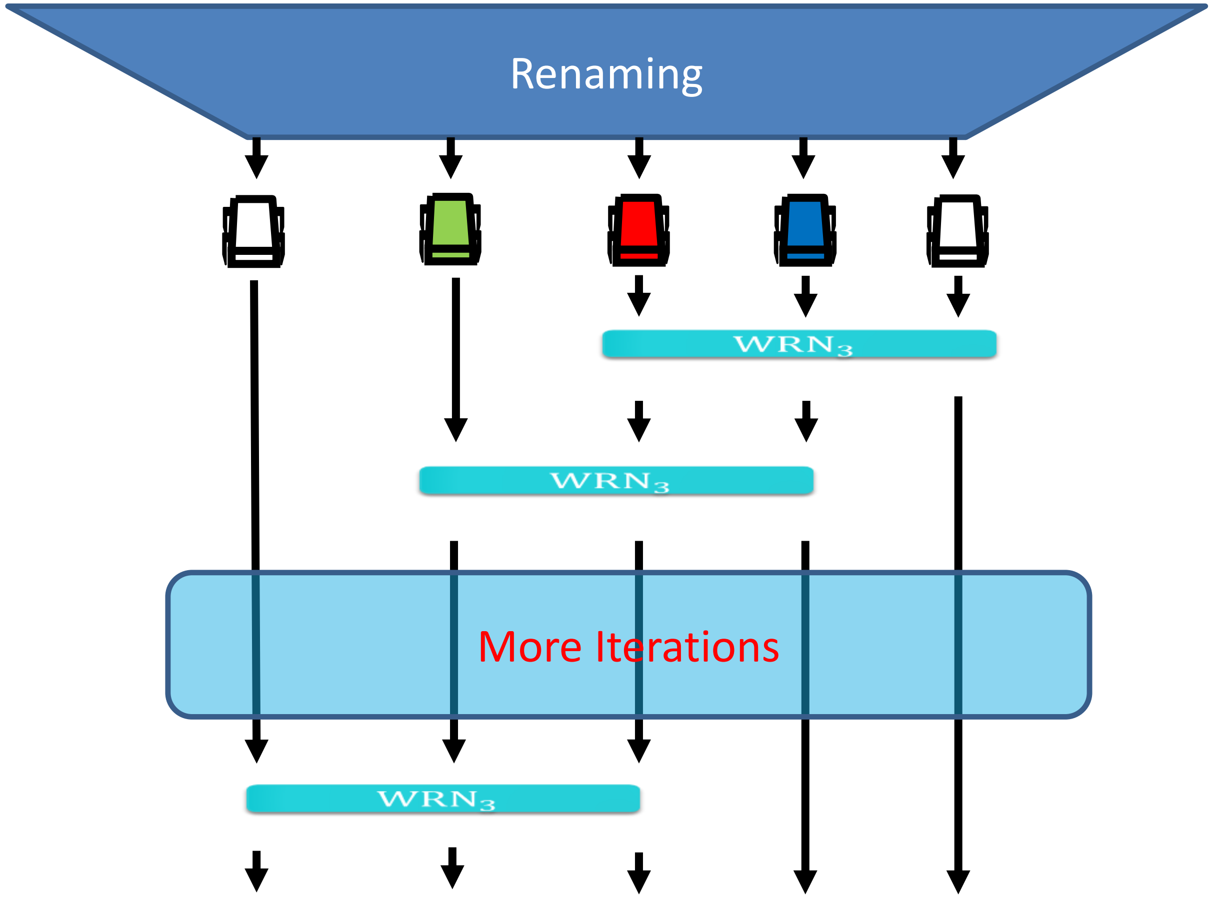
Example – 3 processes



Example – 3 processes



Solution for 3 Processes out of Many



Solution for k Processes out of Many

- Proof:
  - The processes get exactly k names in $\{0,1,\dots,2k-2\}$
  - It's a k-sized subset of $\{0,1,\dots,2k-2\}$
  - Let its index be $\ell^\star$
  - One process must lose until iteration $\ell^\star+1$

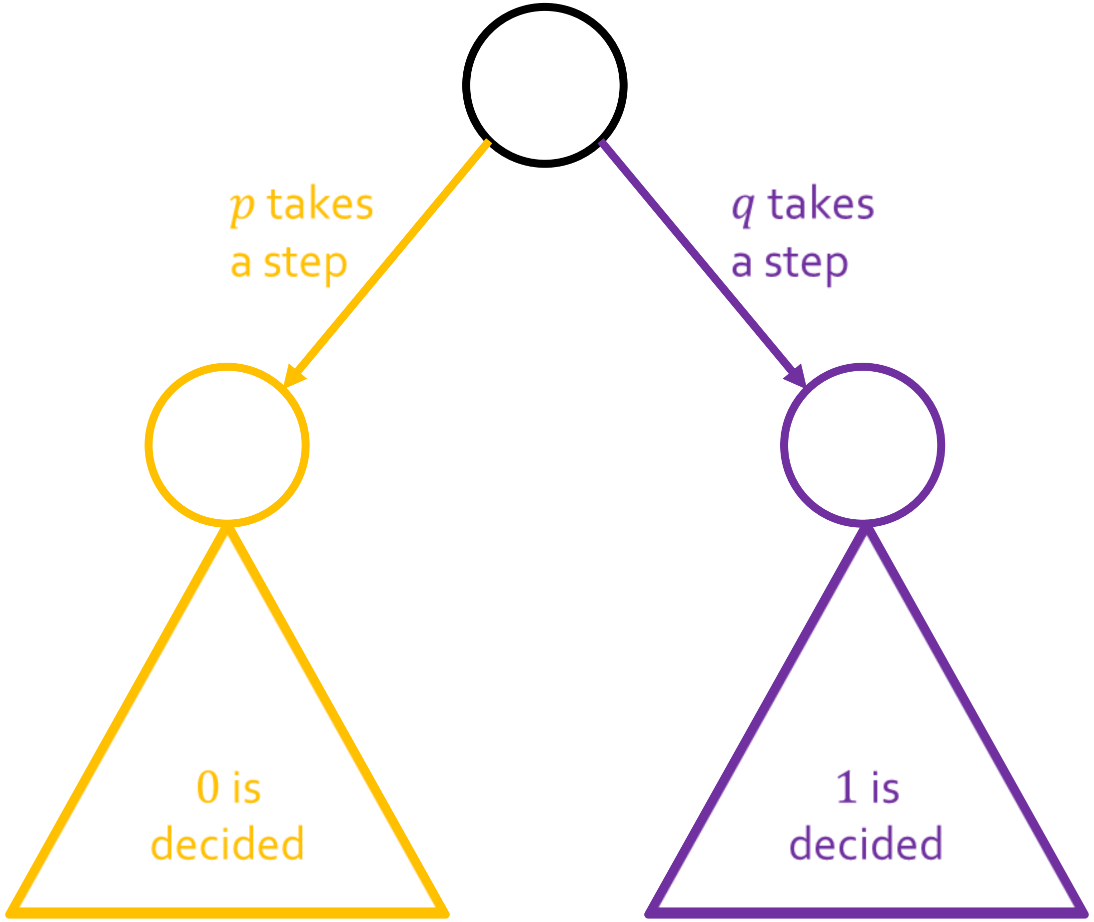
Want to Prove that WRN_k is:

✓ Stronger than registers

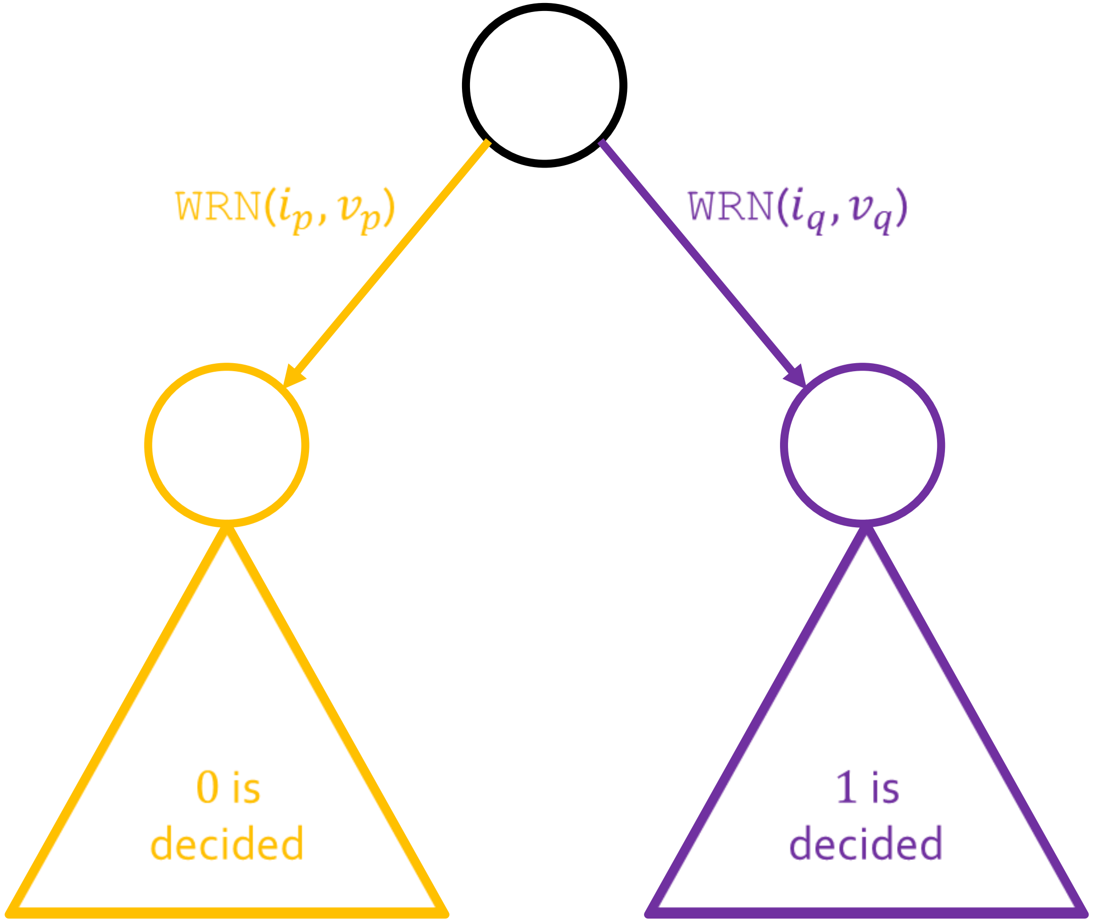
❑ Weaker than 2-consensus

Registers < WRN_k < 2-consensus

WRN_k is Weaker than 2-Consensus



The Critical State



Want to Prove that WRN_k is:

- ✓ Stronger than registers
- ✓ Weaker than 2-consensus

DONE!

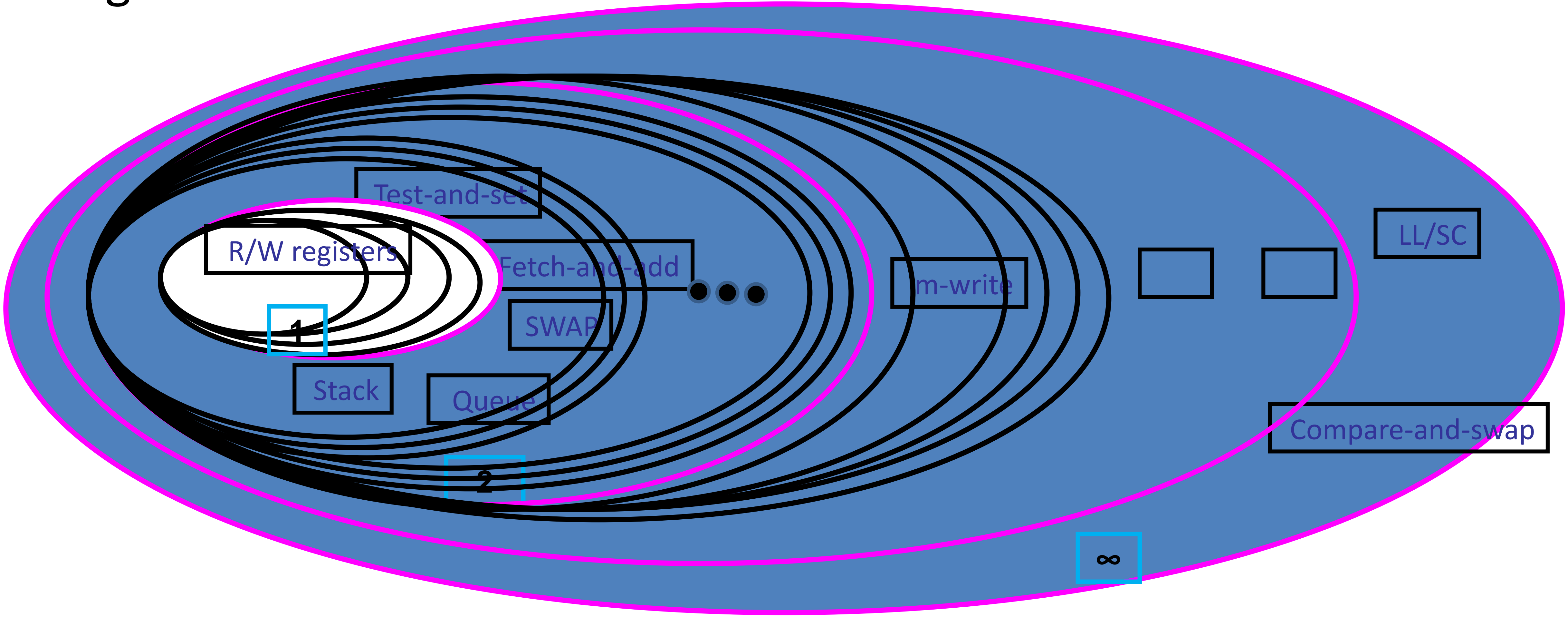
Registers $< \mathrm{WRN}_k <$ **2-consensus**

Remains Open [AEG cojecture]

FALSE

WRNk is a deterministic object WRNk s.t.,:

Reg’s < WRNk < 2-consensus ??



Building an Infinite Hierarchy

- 1 shot WRN_k variant: $1s\mathrm{WRN}_k$

- $1s\mathrm{WRN}_k \geq (k, k-1)$-**strong set election**

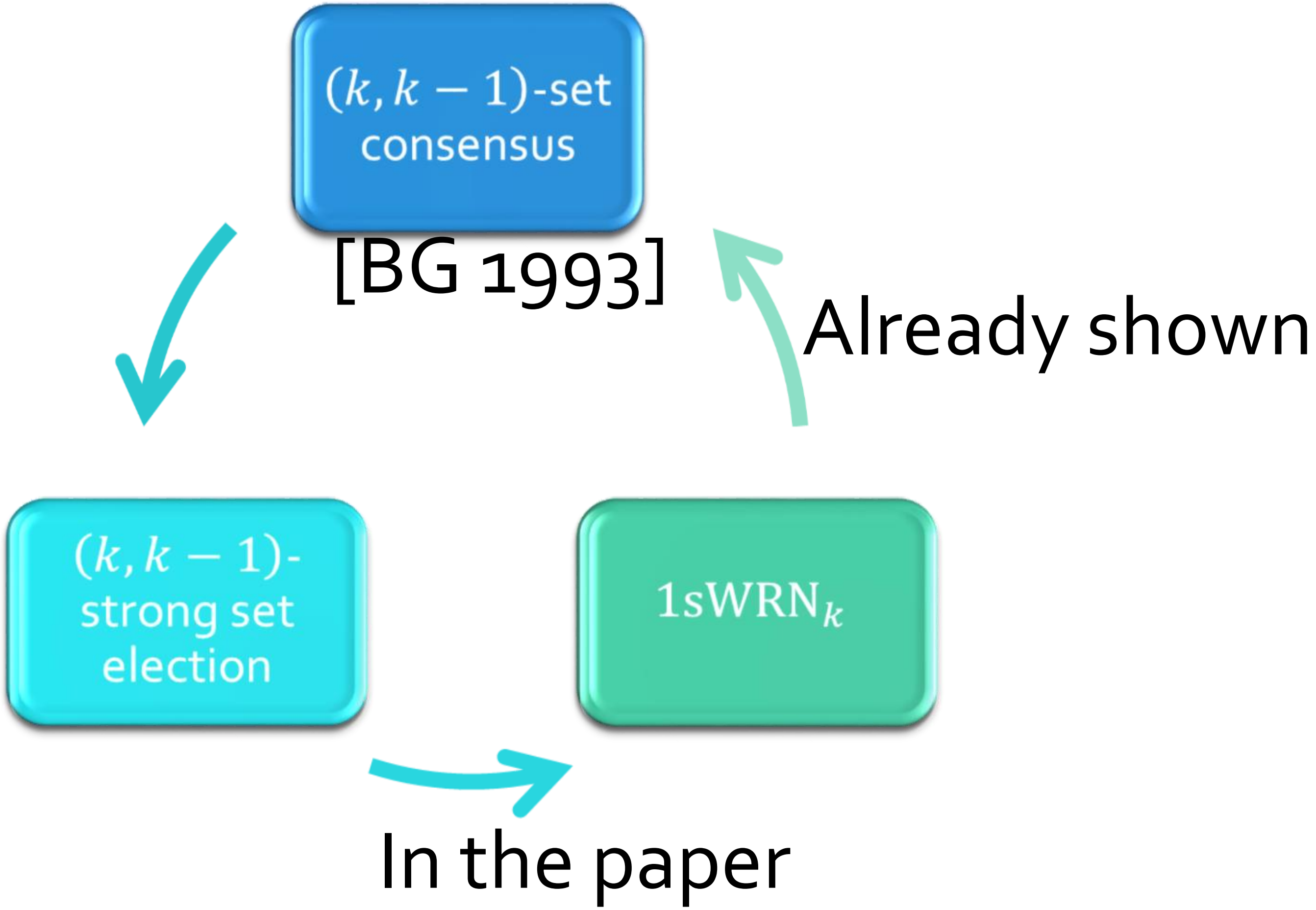
  (If p is elected, then p elects itself)

- Details in the paper

Building from Strong Set Election

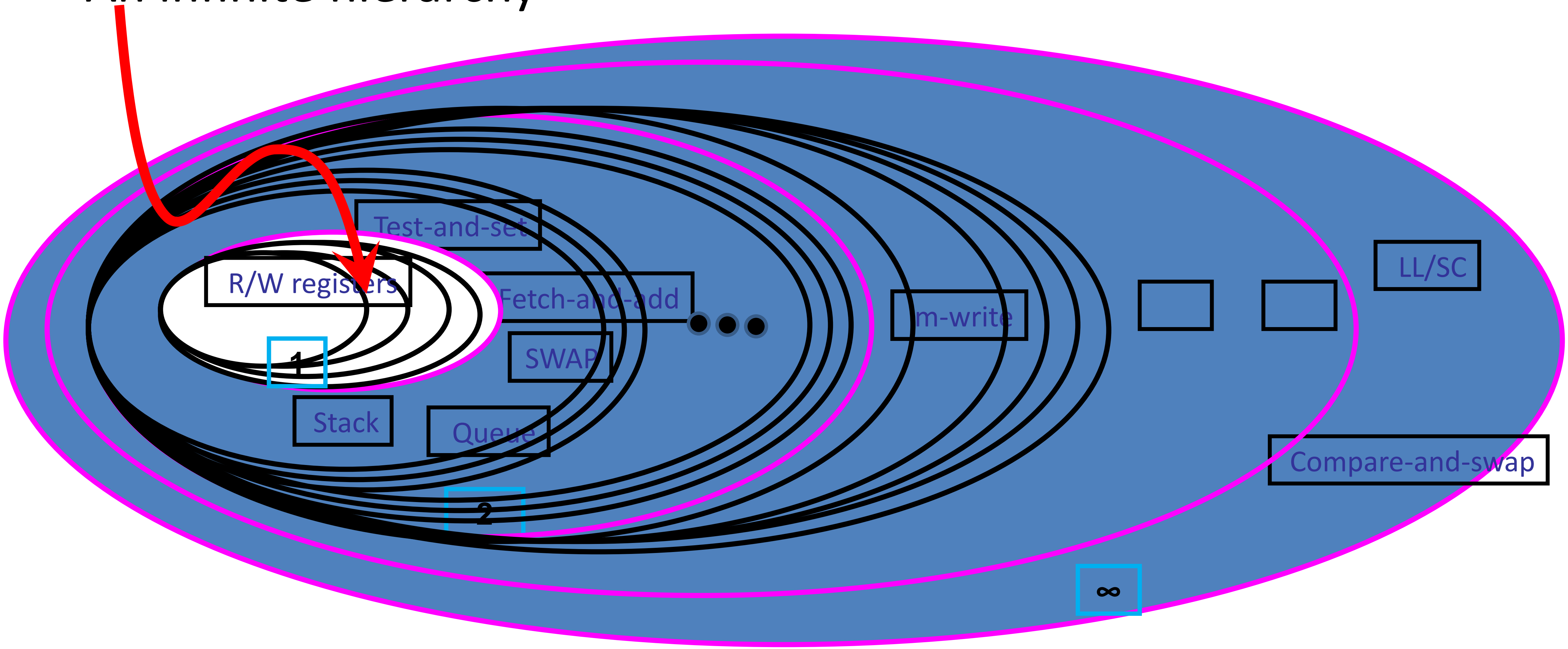
```
shared (k, k − 1)-strong set election implementation SSE
shared MWMR register Doorway, initially opened
shared SWMR register array R[i], 0 ≤ i < k; initially R[i] = ⊥ for every i
shared SWMR register array O[i], 0 ≤ i < k; initially O[i] = ⊥ for every i
function 1sWRN(i, v)          ▷ i ∈ {0, . . . , k − 1} is the index, v ∉ {⊥, ∅} is the value.
    R[i] ← v                                          ▷ v is announced at the index i.
    if Read(Doorway) = opened then
        Doorway ← closed
        if SSE.Invoke(i) = i then
            return ⊥
        end if
    end if
    SR ← Snapshot(R)                                  ▷ SR is a local array.
    O[i] ← SR
    SO ← Snapshot(O)                                  ▷ SO is a local array.
    for j = 0, 1, . . . , k − 1 do
        if SO[j][i] = v and SO[j][(i + 1) mod k] = ⊥ then
            return ⊥
        end if
    end for
    return SR[(i + 1) mod k]
end function
```

Implementation Directions



Infinite hierarchy in L1

- An infinite hierarchy



Set-Consensus below 2-consensus

processes	2-setCons	3-setConse	4-setCons	5-setCons	6-setCons	7-setCons
3						
4						
5						
6						
7						
8						

Reg’s=1-consensus

Set-Consensus below 2-consensus

processes	2-setCons	3-setConse	4-setCons	5-setCons	6-setCons	7-setCons
3	WRN3					
4						
5						
6						
7						
8						

Reg’s=1-consensus

Set-Consensus below 2-consensus

processes	2-setCons	3-setConse	4-setCons	5-setCons	6-setCons	7-setCons
3	WRN3					
4		WRN4				
5			WRN5			
6				WRN6		
7					WRN7	
8						WRN8

Reg's=1-consensus

Set-Consensus below 2-consensus

processes	2-setCons	3-setConse	4-setCons	5-setCons	6-setCons	7-setCons
3	WRN3					
4		WRN4				
5			WRN5			
6			2xWRN3	WRN6		
7					WRN7	
8						WRN8

Reg’s=1-consensus

Set-Consensus below 2-consensus

processes	2-setCons	3-setConse	4-setCons	5-setCons	6-setCons	7-setCons
3	WRN3					
4		WRN4				
5			WRN5			
6			2xWRN3	WRN6		
7					WRN7	
8					2xWRN4	WRN8

Reg’s=1-consensus

Set-Consensus below 2-consensus

processes	2-setCons	3-setConse	4-setCons	5-setCons	6-setCons	7-setCons
3	WRN3					
4		WRN4				
5			WRN5			
6			2xWRN3	WRN6		
7					WRN7	
8					2xWRN4	WRN8

Reg's=1-consensus

??? OPEN

???

Consensus vs. Set Consensus Power

- We've shown: Registers < WRN_k < 2-consensus
  - Consensus number is 1
  - WRN_k solves $(k-1)$-set consensus

- Consensus and set consensus powers are **unrelated**!

Delporte-Gallet, Fauconnier, Gafni, Kuznetsov

- Set-consensus power vector (Obj)=

  (k1, k2, k3, …..)

  where Obj and registers can w-f implement (ki,i)set-consensus and NOT (ki+1,i)set-consensus.

Set Consensus Power Vector

- For an object O:
  - $v_O = \langle a_1, a_2, \ldots \rangle$
  - O and registers solve **wait-free** i-set consensus:
    - for a_i processes
    - not for $a_i + 1$ processes
  - a_1 is the consensus number of O

Remaining Open Questions

- For deterministic objects:
  - The case $(k, k-1)$-set consensus is resolved
    - What about arbitrary (n, m)-set consensus?
      - I.e., $(4,2)$ or $(5,2)$
  - Is there a more fine grained hierarchy?
    - More than set-consensus vector power?

Thank you!