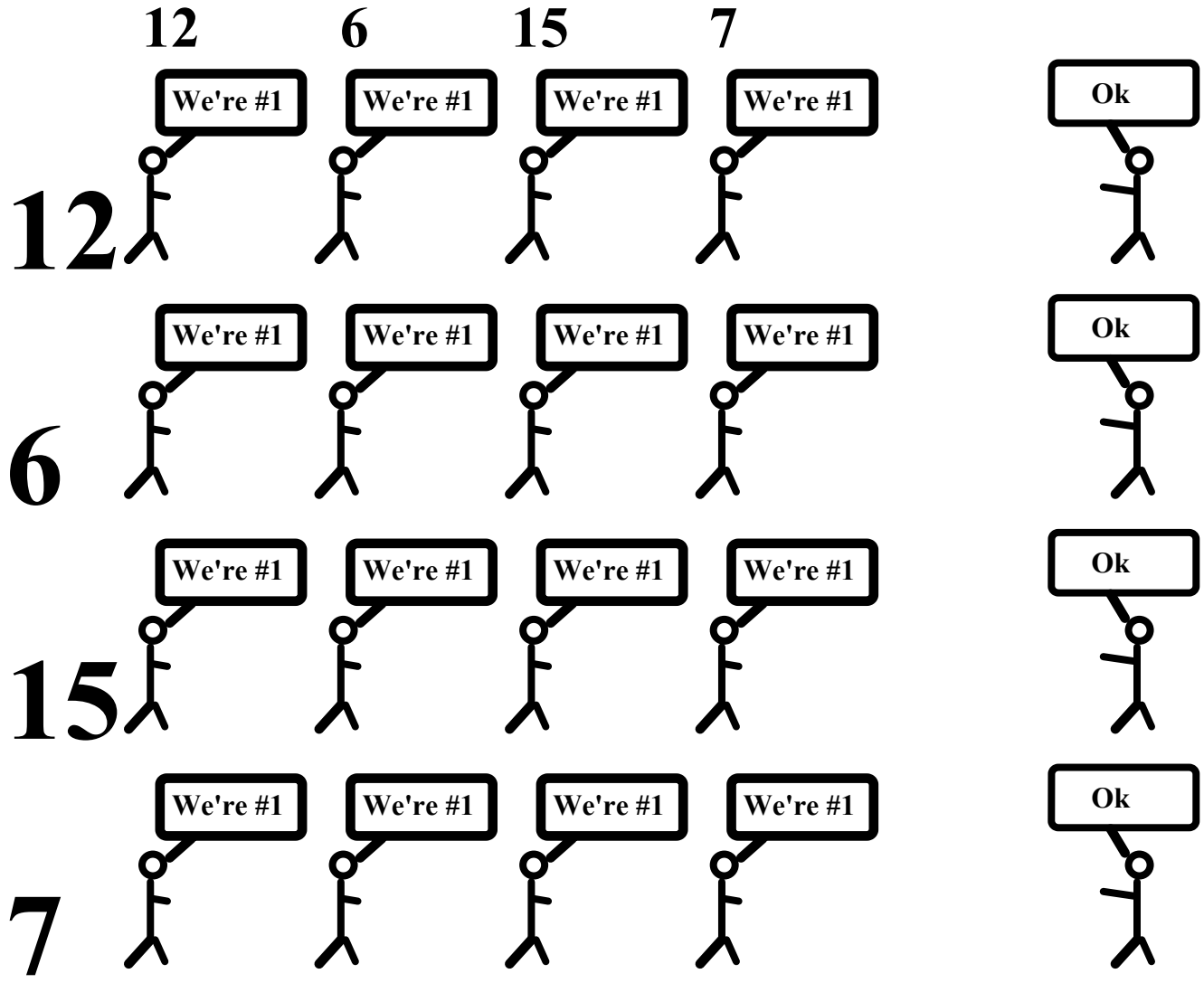
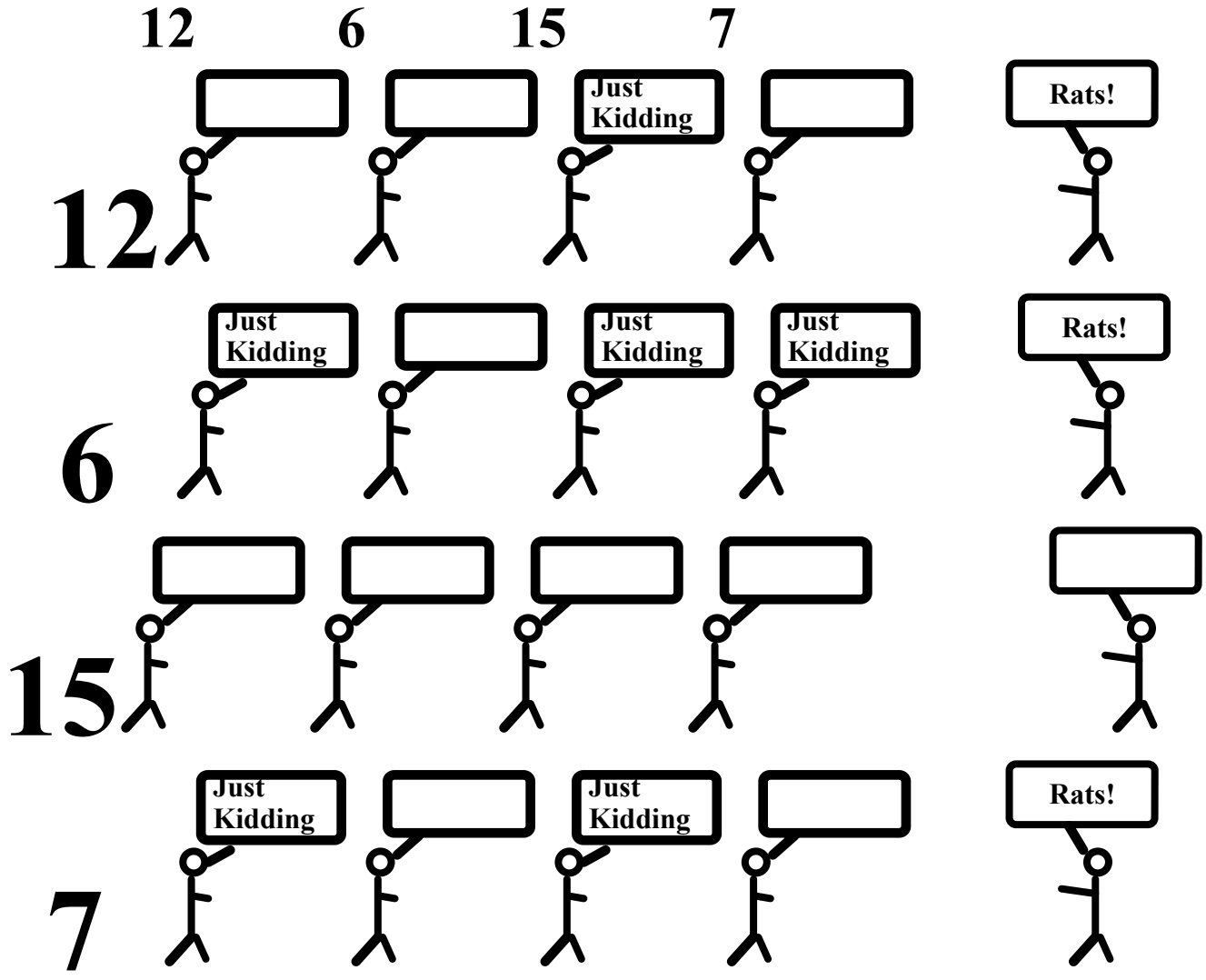
Finding the Maximum by Controlled Anarchy

Step#1: Everyone's an Optimist



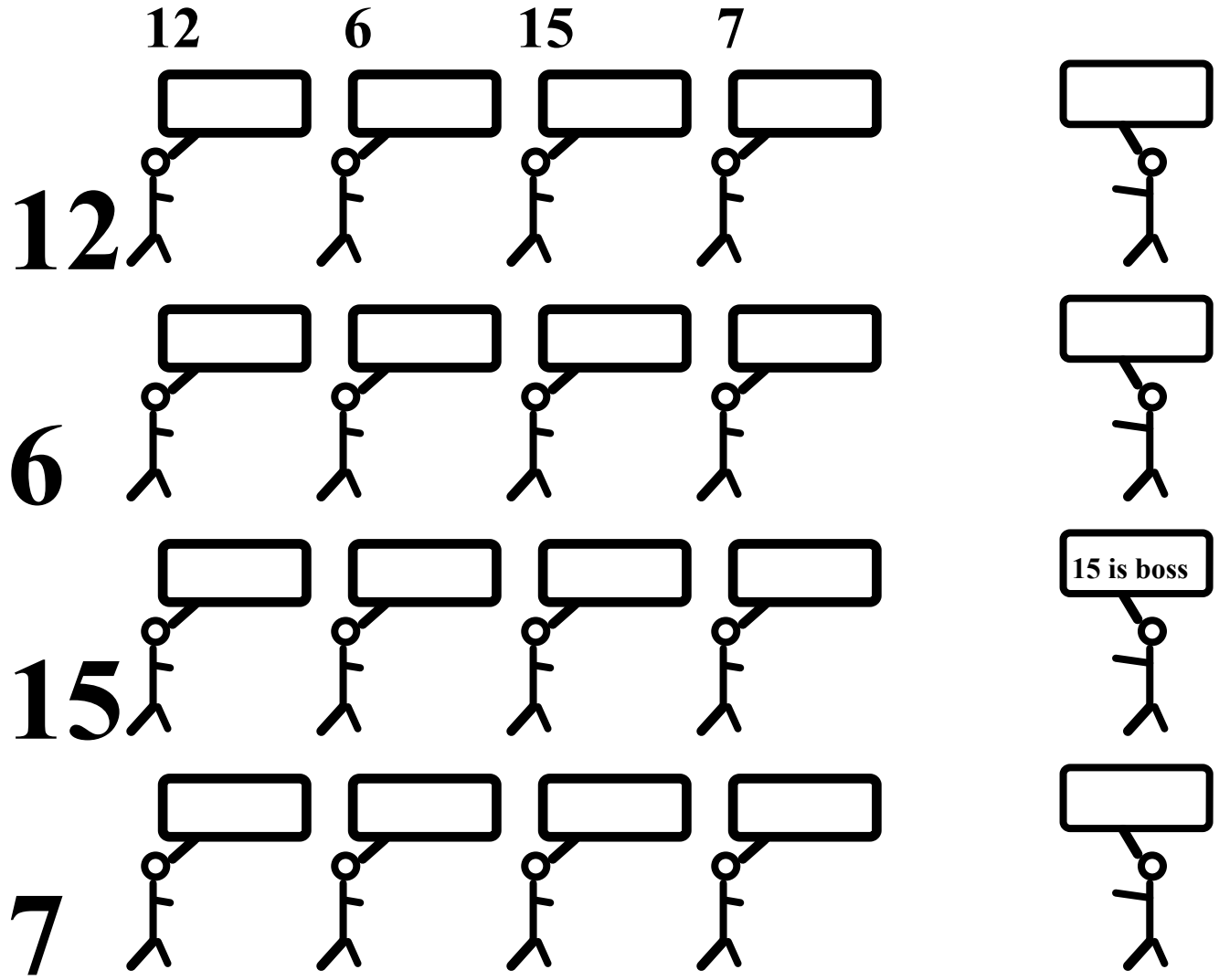
This is the Meatiest Part

Step#2: Realism Sets In



That's All Folks

Step#3: Reporting the Answer



Analysis of Very Fast Max

Optimal in Time, Not Work on CRCW (Concurrent Read Concurrent Write) PRAM (Parallel Random Access Machine)

- Assign **N** processors to initialize **M** in 1 step.
- Assign all $\mathbf{N^2}$ processors to first statement to fill **B** in 1 step.
- Assign all $\mathbf{N^2}$ processors to 2nd statement to fill **M** in 1 step.
- Assign **N** processors to 3rd statement to select **maxVal** in 1 step.

That Was Inefficient but Real Fast

- Can Solve Any Size Problem in 3 Steps
  But we need to make unreasonable assumptions about memory (CRCW)

- Use Lots of Processors
  Over a Million to Find Max of 1000

- We Want Fast but Not Too Expensive

Sense of Optimality of Max

It Depends on Model and Goals

- Can use $\mathbf{N^2}$ processors to find max of **N** elements on **O(1)** time.
- Work is $\mathbf{O(N^2)}$ on CRCW PRAM.

- Minimal work on EREW or CREW PRAM requires **O(lg N)** time.

- Can achieve **O(lg lg N)** time on CRCW doing minimal work.

Fast, Inefficient Max in Unity Notation
|| is parallel composition

Program max

declare *B* **:** *array [1..N, 1..N] of boolean*
M **:** *array [1..N] of boolean*
maxVal **:** *integer*

initially $\langle \| \; i : 1 \le i \le N :: M[i] = false \rangle$

assign

$\langle \| \; i,j : 1 \le i \le N \;\&\; 1 \le j \le N :: B[i,j] = A[i] \ge A[j] \rangle$
||
$\langle \| \; i : 1 \le i \le N :: M[i] = \langle \;\&\; j : 1 \le j \le N :: B[i,j] \rangle \rangle$
||
$\langle \| \; i : 1 \le i \le N :: maxVal = A[i]$ **if** $M[i] \rangle$

end *{ max }*