The Two Generals Problem
The N Generals Problem

The coordinated Attack Problem

Akkoyunlu, Ekerdem, Huber 1975

System Model

N processes

Synchronous system

message loss

no crashes
The problem:

Every process has an input in \( S_0, I \).

1 = attack
0 = abort

Every process must decide (output 0 or 1) in finite time.

Can't change decision.
Correctness Conditions

1. Agreement: no 2 processes decide on different values

2. Validity:
   If all processes have input 0, then 0 is the only possible decision value.
   If all processes have input 1 and all messages are received, then 1 is the only possible decision value.

3. Termination: all processes decide after a finite number of messages
Algorithm Abstraction
(abstraction for a synchronous system)

Initially:
Every process in an arbitrary state
All channels are empty

Processes repeatedly perform following (in lock step):

Round

1. receive and process any incoming messages and update state
2. send finite number of messages
Proof (for N=2)

- Proof by contradiction

- Assume such an algorithm exists.
  - Agreement: $p_1$ and $p_2$ decide on same value.
  - Validity: If $p_1$ and $p_2$ both have input 0, they decide 0. If they both have input 1, and no messages are lost, they decide 1.
  - Termination: Both $p_1$ and $p_2$ eventually decide.

- Suppose the algorithm takes $r$ rounds to decide.
  - w.l.o.g. $p_1$ and $p_2$ both send a message in each round.
Execution A

- By the termination requirement, both processes decide.
- By the validity requirement, both decide 1.

No messages are lost.
Execution B

- Execution B looks exactly like Execution A to $p_1$
- Therefore, $p_1$ decides 1
- By termination and agreement, $p_2$ must also decide 1

Final message from $p_1$ to $p_2$ is lost.

Round r
Execution C

- Execution C looks exactly like Execution B to $p_2$
- Therefore, $p_2$ decides 1
- By termination and agreement, $p_1$ must also decide 1
Execution X

- Following this reasoning, we reach an execution where **no messages are received**, and **both processes must decide 1**
Execution Y

- Execution Y looks exactly like execution X to $p_1$
- Therefore, $p_1$ decides 1
- By termination and agreement, $p_2$ must also decide 1

All messages are lost.
Execution Z

- Execution Z looks exactly like execution Y to $p_2$
- Therefore, $p_2$ decides 1
- By termination and agreement, $p_1$ must also decide 1
- This violates validity – both start with 0 and must decide 0
No Coordinated Attack?

• There is no algorithm that solves the coordinated attack problem in the presence of message loss
  – Holds for $N > 2$ as well

• But we do need to solve this problem in real-world systems:
  – Must strengthen the model and/or relax the requirements

• Can make assumptions on probability of message loss:
  – There will be some chance of violating validity or agreement

• Can let processes use randomization:
  – There will be some chance of violating validity or agreement