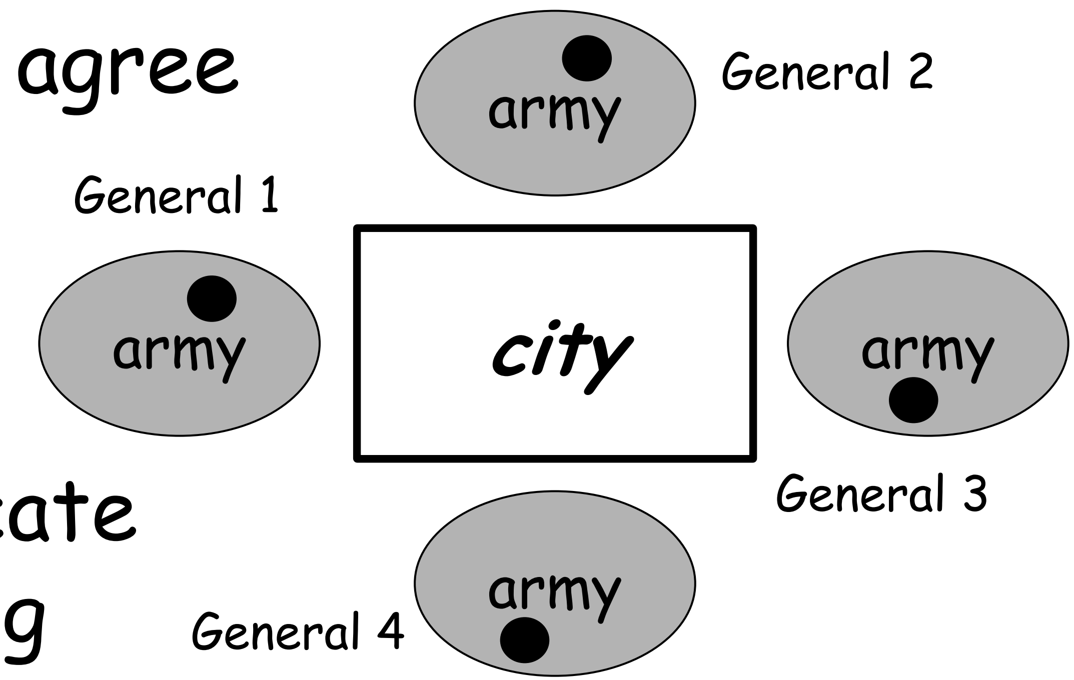
DoC 437 - 2009

Distributed Algorithms

Part 4: Byzantine Consensus

Byzantine generals problem

- Consensus in the presence of uncertainty
- All *loyal* generals must agree to attack or retreat despite the presence of any traitors



- Generals can communicate only by message passing
- Generals that are *traitors* may do anything they wish (e.g., send incorrect messages)

Problem statement

- Commanding general must send an order to lieutenant generals such that

    all loyal lieutenant generals obey the same order

    if the commanding general is loyal, then every loyal lieutenant general obeys the order sent

The "general" problem statement

- The basic model

  a network of *n* processes that communicate through bidirectional channels, where one designated process initiates messages to the other processes

  processes are either *correct* or *byzantine*

- The desired outcome

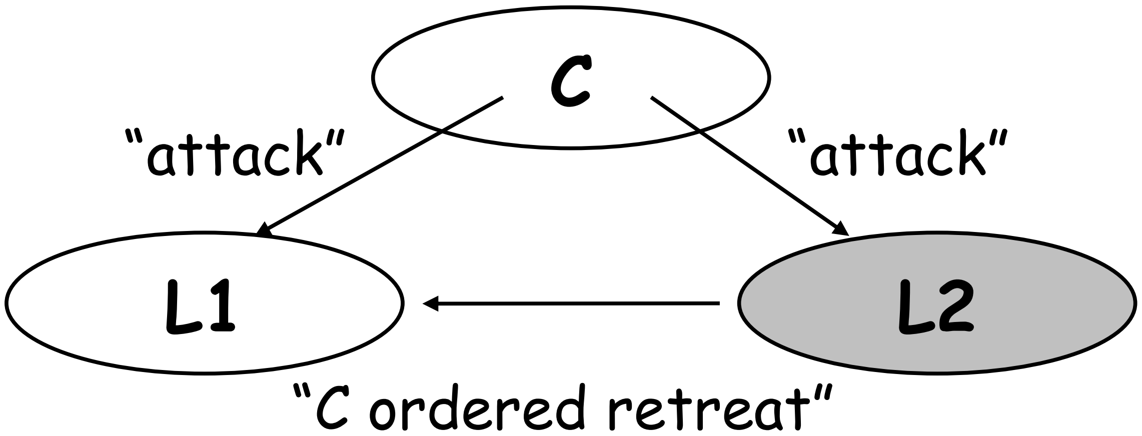
  IC1: correct processes receive the same message

  IC2: if the sending process is correct, then the message received is identical to the message sent

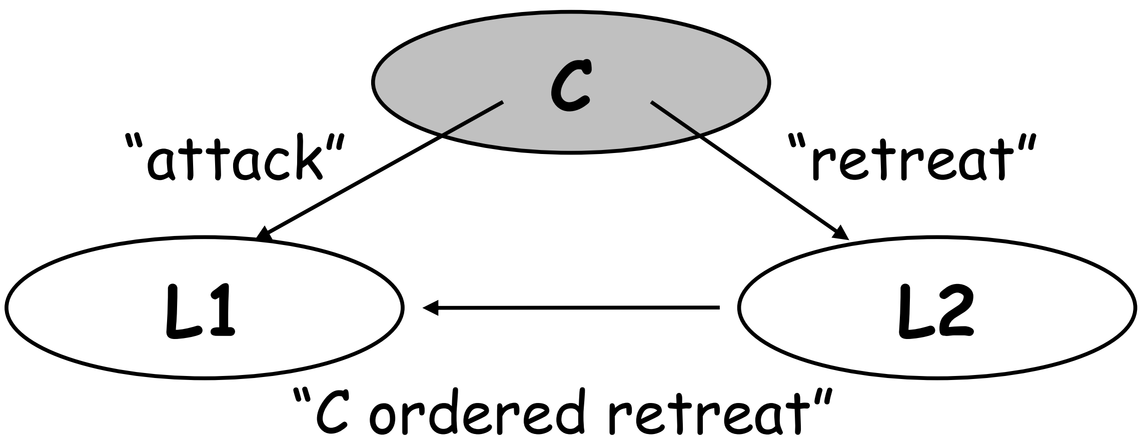
Interactive consistency

- IC1 and IC2 are known as *interactive consistency* conditions
- Note that if the commanding general is loyal, then IC1 follows from IC2
- A solution to the Byzantine generals problem allows reliable communication in the presence of *commission* errors as well as omission errors
  - 2PC handled only omission errors

Impossibility result



case 1: lieutenant is a traitor



case 2: commander is a traitor

- In case 1, L1 should attack to satisfy IC2
- In case 2, if L1 attacks, IC1 is violated
- L1 cannot distinguish case 1 from case 2
- No solution for three generals with traitor

The "general" impossibility result

- There is no solution having fewer than $3m+1$ generals, given m generals that are traitors

- If $m=1$, then we have the previous cases plus an extra loyal general

- What can we do with that extra loyal general to solve the problem?

The Lamport et al. solution

- Communication assumptions

  A1: every message sent is delivered correctly

  A2: the receiver of a message knows who sent it

  A3: the absence of a message can be detected

- Consequences

  A1 and A2 prevent traitor from interfering with messages between two other generals

  A3 prevents traitor from blocking a decision

The Lamport et al. solution

- UM(n,m): solution for n generals and m traitors

  $n > 3m$

- Assume messages are binary

  "attack" or "retreat"

- v_{def} is a global default used if traitorous commander sends no message

  e.g., "retreat"

- majority($v_1,\ldots,v_{n-1}$) yields majority value of values or v_{def} if a tie

The algorithm, in brief

UM(n,0)

step 1: general G sends v to every lieutenant general L_i

step 2: each L_i uses v or, if no value received, uses v_{def}

UM(n,m)

step 1: general G sends v to every lieutenant general L_i

step 2: for each L_i

let v_i = value received from G or v_{def} if none received

send v_i to n-2 other lieutenants using UM(n-1,m-1)

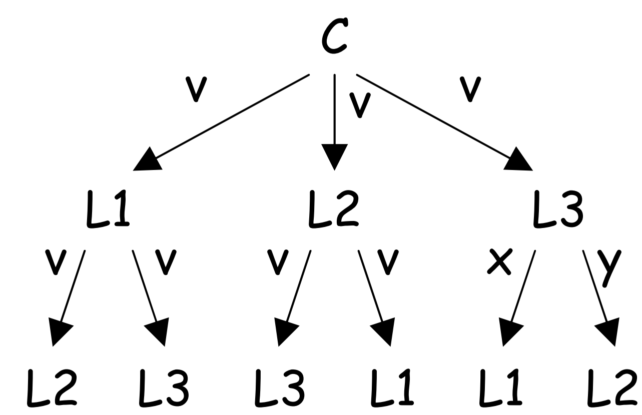
step 3: for each i and each $j \neq i$

let v_j = value L_i received from L_j or v_{def} if none received

step 4: each (loyal) lieutenant general L_i uses

majority($v_1,\ldots,v_{n-1}$) to decide action

Example 1: traitorous lieutenant



At the end of round 1:

L1: $v_1 = v$

L2: $v_2 = v$

L3: $v_3 = v$

At the end of round 2:

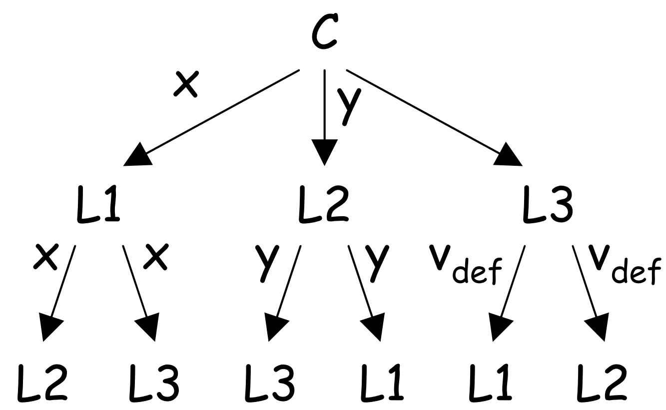
L1: $v_1 = v$, $v_2 = v$, $v_3 = x$

L2: $v_1 = v$, $v_2 = v$, $v_3 = y$

L3: $v_1 = v$, $v_2 = v$, $v_3 = v$

At the end of round 2 each of the lieutenants has received a set of values and arrives at the same decision (IC1); and the value sent by C is the majority value (IC2)

Example 2: traitorous commander



At the end of round 1:

- L1: $v_1 = x$
- L2: $v_2 = y$
- L3: $v_3 = v_{def}$

At the end of round 2:

- L1: $v_1 = x$, $v_2 = y$, $v_3 = v_{def}$
- L2: $v_1 = x$ $v_2 = y$, $v_3 = v_{def}$
- L3: $v_1 = x$, $v_2 = y$, $v_3 = v_{def}$

At the end of round 2 the three loyal lieutenants have received the same value, majority(x,y,v_{def}) and do not violate IC1 or IC2

Proof of correctness of UM(n,m)

Lemma: For any m and k, UM(n,m) satisfies IC2 if there are more than $2k+m$ generals and at most k traitors

Proof: (by induction on m)

From A1 it is obvious that UM(n,0) works if the commander is loyal, that is, UM(n,0) satisfies IC2

Proof of correctness of UM(n,m)

Now assume UM(n-1,m-1) satisfies IC2 for m>0 and prove it for m...

In step 1, the loyal commander sends a value v to n-1 lieutenants, while in step 2 each loyal lieutenant applies UM(n-1,m-1)

By hypothesis we have n>2k+m or n-1>2k+(m-1)

By the induction hypothesis, every loyal lieutenant gets $v_j = v$ from each loyal lieutenant j

Proof of correctness of UM(n,m)

Since there are at most k traitors and

n-1 > 2k+(m-1) >= 2k... k < (n-1)/2

a majority of the n-1 lieutenants are loyal

Hence, each loyal lieutenant has $v_i = v$ for a majority of the n-1 values, so obtains majority(v_i,..v_{n-1})=v in step 3, satisfying IC2

Proof of correctness of UM(n,m)

Theorem: For any m, UM(n,m) satisfies IC1 and IC2 if there are more than 3m generals and at most m traitors

Proof: (by induction on m)

If there are no traitors it is easy to see using A1 that UM(n,0) satisfies IC1 and IC2

Proof of correctness of UM(n,m)

Now assume UM(n-1,m-1) satisfies IC1 and IC2 for m>0 and prove it for m

Case A: The commander is loyal

By taking k = m in the lemma, UM(n,m) satisfies IC2

Since IC1 follows from IC2 if the commander is loyal, we now only consider...

Case B: The commander is a traitor

Proof of correctness of UM(n,m)

Case B: The commander is a traitor

There are at most m traitors and the commander is a traitor, therefore at most m-1 of the lieutenants are traitors

Since there are more than 3m generals, there must be more than 3m-1 lieutenants and $3m-1 > 3(m-1)$

Hence, we can apply the induction hypothesis to conclude that UM(n-1,m-1) satisfies IC1 and IC2

Proof of correctness of UM(n,m)

Therefore, for each j, any two loyal lieutenants get the same value for v_j in step 3

(this follows from IC2 if one of the two lieutenants is j, and from IC1 otherwise)

Hence, any two lieutenants get the same vector of values and therefore the same majority(v_i,..v_{n-1}) in step 3, proving IC1

Message complexity of UM(n,m)

Applying UM(n,m) initially causes the issuing of n-1 messages

Each such message invokes UM(n-1,m-1) and causes n-2 messages to be issued

round 1: $(n-1)$ messages

round 2: $(n-1)(n-2)$ messages

...

round m+1: $(n-1)(n-2)\ldots(n-(m+1))$ messages

Total: $O(n^{m+1})$

Time complexity of UM(n,m)

m+1 rounds

This number of rounds is a *fundamental characteristic* of algorithms that arrive at a consensus in the presence of m possibly faulty processes

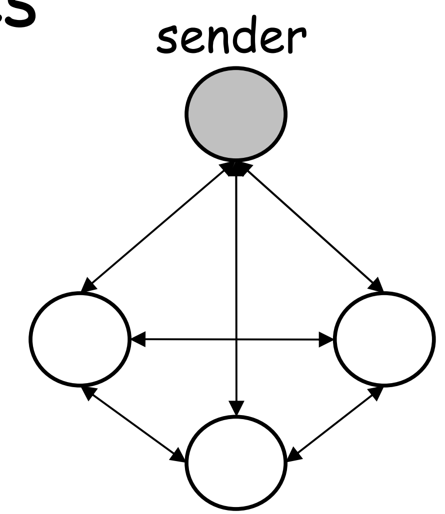
Terminating reliable broadcast

- The basic model

  completely connected, synchronous network of n processes with one sender and n-1 receivers

  initially, sender holds a value $v \in V$, $|V| \geq 2$ and the n-1 receivers do not know the value

  up to m crash failures, but no link failures



- The desired outcome

  each receiver must decide some value v or the special value SF ("sender faulty")

Terminating reliable broadcast

- The desired outcome (properties)

  agreement: no two correct processes decide different values

  validity: if the sender is correct and has initial value v, then any process that decides must decide v

  integrity: if a receiver decides $v \neq$ SF, then the sender's initial value is v

  termination: correct processes eventually decide

How is TRB different from Byzantine Agreement?