Algorithmic Modeling and Complexity **Fall 2003**

Lecture 17: 28 October

Lecturer: M. Veldhorst *Scribes: Paulo Magalhães and Rui Coelho*

17.1 Overview

How to reach consensus in a faulty synchronous networks, in particular, in the presence of *Byzantine processes* was the topic for this lecture. We approached this by using an analogy of the *Byzantine Generals Problem* and finished by introducing a solution using signed messages.

17.2 Synchronous networks

In a synchronous networks the operations are done according with pulses. A pulse is a periodic moment when all messages are sent, received and local computations for the given period take place.

A *Byzantine* process is a process with an erroneous behaviour, having unpredictable actions. For our study, let's consider the following assumptions:

- a message sent by s to d arrives in d without any change;
- each message contains its correct source (even a *Byzantine* process can't change it);
- the absence of an expected message can be detected.

17.2.1 Byzantine broadcast

The origin of the term Byzantine comes from the city of Byzantium, later renamed Constantinople and then Istanbul. In this city, the Byzantine empire were vitiated by a bureaucratic over-elaboration bordering on lunacy: quadruple banked agencies, dozens or even scores of superfluous levels and officials with high flown titles unrelated to their actual function, if any.
Our problem of Byzantine processes, is then similar to the Byzantine Empire army generals that are trying to conqueror a city. Hence, the commander gives an order (i.e: attack, retreat, etc) spreading it through all the generals. The main problem is that the generals, or even the commander, might be traitors (Byzantine processes).
Assume that there are N different armies, each one having only one general commanding them and being the commander one of those generals. Let t be the number of traitors among the N generals. Knowing that the commander issues an order v, the following conditions must occur:

1. all loyal generals must have the same value;
2. if the commander is loyal, all loyal generals must have have the value v.

Theorem 17.1 *If in a total of* 3 *generals, one of them is a traitor, then the problem can't be solved.*

Proof: Let's consider, for simplicity, the case in which the only possible orders v are *attack* and *retreat*. Considering the situation pictured in Fig. 17.1. in which the *commander* is loyal and sends an *attack* order but *general 2* is a traitor and reports to *general 1* that the commander ordered *retreat*. For condition (2) to be satisfied, *general 1* must comply with the *attack* order. Now looking at Fig. 17.2. in which the the *commander* is a traitor and sends different messages to his generals. The *general 1* doesn't know who the traitor is. Then, so that he complies to condition (1) he should *retreat*. However, for *general 1* both situations are indistinguishable, and so, it's impossible for him to decide which of the orders to accept.

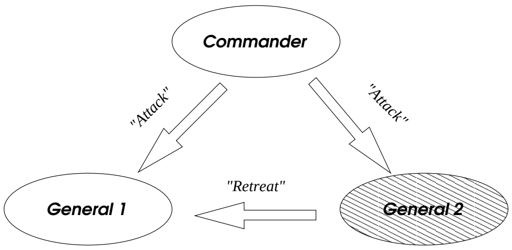

Fig. 17.1. *general 2* is a traitor

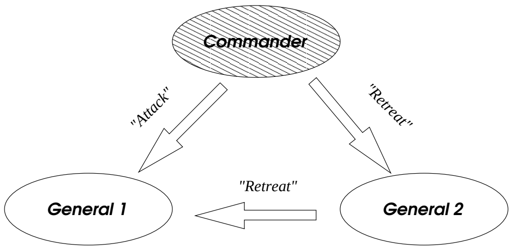

Fig. 17.2. The *commander* is a traitor.

■

Theorem 17.2 *Having* N *armies, and* $t \geq \frac{N}{3}$*, then there is no algorithm* A *that solves the problem.*

Proof: By contradiction: assume there is an algorithm A that solves the problem of N generals with $t \geq \frac{N}{3}$. Then, it is possible to partition the generals into 3 groups of equal size with one of them consisting of only traitors. Considering each of this groups as a individual general (of a larger army of lieutenant generals for example) we see that the problem reduced itself to the case of 3 generals and one traitor. Then, algorithm A must be able to solve this last problem. But, as was shown, that is impossible: contradiction. ■

Let P be a set of processes and $P' \in P$ such that $|P'| \leq t$. Then $P - P'$ has a majority of a loyal processes. Let $majority(v_1, v_2, \ldots, v_N)$ be a function with the following return values:

- v_i if value v_i occurs in $v_1, ..., v_N$ by majority.
- $v_{undefined}$ otherwise.

Let $broadcast(Q, q \in Q, v, t)$ be a function that sends to every element of Q-q (i.e.: except himself) the value v and knowing that there can be at most t traitors.

Algorithm $A(0)$

1. The commander sends his value to every general.
2. Each general uses the value he receives from the commander.

Algorithm $A(t)$, with $t > 0$

1. The commander sends his value to every general.
2. For each i, let v_i be the value general i receives from the commander. General i acts as the commander in Algorithm $A(t-1)$ to send the value v_i to each of the $N-2$ other generals using the *broadcast* function.
3. For each i, and each $j \neq i$, let v_j be the value general i received from general j in **step (2)**, using $A(t-1)$. General i uses the value of $majority(v_1, ..., v_{N-1})$.

Theorem 17.3 *For any t, algorithm $A(t)$ satisfies conditions (1) and (2) if there are more than $3t$ generals and at most t traitors.*

Proof: The proof is by induction on t. If there are no traitors, then it is easy to see that $A(0)$ satisfies (1) and (2). Induction hypothesis: assume that the theorem is true for $A(t-1)$, with $t > 0$. Now, for $A(t)$:

Case 1: commander is loyal. Then, if there are more than $3t$ generals and at most t traitors, the majority of values v will always be equal to the value given by the commander, thus satisfying (1) and (2) and so proving the claim.

Case 2: commander is a traitor. Then, it sends values $v_1, v_2, ..., v_{N-1}$ to $p_1, p_2, ..., p_{N-1}$. Each p_i then calls $A(P - p_0, p_i, v_i, t-1)$. If there are t traitors and the commander is one of them then there are $t-1$ generals that are traitors. By the theorem, there must be more than $3t$ generals and thus more than $3t-1$ generals if the commander is not included. If so, knowing that $3t-1 > 3(t-1)$, one can apply the induction hypothesis to conclude that $A(t-1)$ satisfies conditions (1) and (2). So, for each j, any 2 loyal generals get the same value v_j in step (3). Thus, any 2 loyal generals get the same set of values $v_1, v_2, ..., v_{N-1}$ and therefore decide on the same value of $majority(v_!, v_2, ..., v_{N-1})$ in step (3), proving condition (1).

■

17.2.2 Signed Messages

As we saw it is the traitors ability to lie that makes the *Byzantine Generals Problem* so difficult. The problem becomes easier to solve if we can restrict that ability. One way to do this is to allow the generals to send unforgeable signed messages. We can then assume that:

- A loyal general signature in a message cannot be forged, nor the contents changed.
- Every general can check the authenticity of another general's message.

Signed messages are of the form x:i where x denote the value signed by General i. If so, v:j:i denotes the value v signed by j, and then that value v:j signed by i. In this algorithm, each general keeps track of a growing set V_i, containing the set of properly signed orders he has received so far. Consider a message v:0 to be a message from the commander.

In the following algorithm, the commander sends a signed order to each of his generals. Each general then adds his signature to that order and sends it to the other generals, who add their signatures and send it to others, and so on. This means that a general must effectively receive one signed message, make several copies of it, sign and send those copies. It does not matter how these copies are obtained; a single message might be photocopied, or else each message might consist of a stack of identical messages which are signed and distributed as required.

Algorithm $SM(t)$

Initially $V_i = \emptyset$.

(1) The commander signs and sends his value to every general.

(2) For each general i:

(A) If general i receives a message of the form v:0 from the commander and he has not yet received any order, then

(i) he lets V_i equal {v};

(ii) he sends the message v:0:i to every other general.

(B) If general i receives a message of the form v:0:j_1:...:j_k and v is not in the set V_i, then

(i) he adds v to V_i;

(ii) if $k < t$, then he sends the message v:0:j_1:...:j_k: i to every general other than j_1, ... j_k.

(3) For each i: When general i receives no more messages , he obeys the order choice (V_i).

Note that if the commander is loyal, then the set V_i should never contain more than a single element. If that is not the case then the commander must be a traitor as shown in Fig. 17.3.

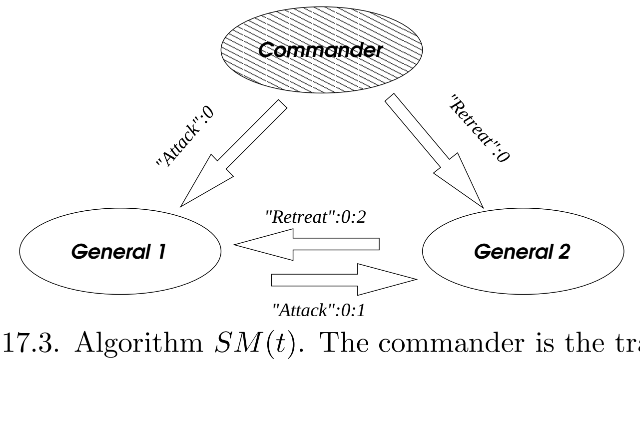

Fig. 17.3. Algorithm $SM(t)$. The commander is the traitor.

References

[1] L. Lamport, M. Pease, and R. Shostak. *The Byzantine generals problem*, ACM Transactions on Programming Languages and Systems (TOPLAS), v.4 n.3, p.382-401, July 1982.