Quiz 3

Dienstag, 31. Oktober 2017 17:03

Team **minker**
Josua Cantieni, Christian Knieling, Eric Mink

1. *What are the main advantages of using Vector Clocks over Lamport timestamps?*
Lamport guarantees that if B is causally dependent on A, then $L(A) < L(B)$.
However, the inverse does not hold: Just because something has a larger timestamp does not mean it depends on A.

Vector clocks allow us to find out whether messages A and B are causally related, because they provide a partial order instead of a total order. A Vector clock basically contains a Lamport timestamp for every thread. *If and only if* all those entries in A are smaller or equal and one of them is strictly smaller than the corresponding entries in B, the message A happened-before the message B.
This means we can also find out that two messages are independent.

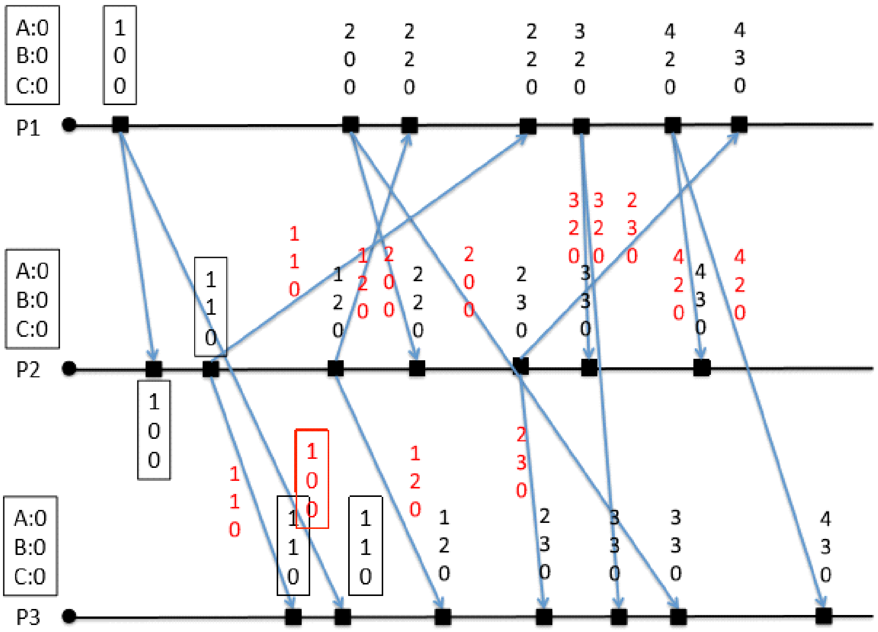
2. *Give the two conditions for two Vector Clocks to be causally dependent.*

- $\forall x.\ VC(A)_x \leq VC(B)_x$
- $\exists x.\ VC(A)_x < VC(B)_x$

Both these conditions have to hold for B to be causally dependent on A.
Notation: $VC(N)_i$ denotes the i-th component of the vector clock of message N.

3. *Does a clock tick happen before or after the sending of a message? What are the implications of changing this?*
The clock tick happens before sending a message. This means that the message itself contains its own logical timestamp. If the clock tick would happen after sending the message, the receiver would have to increment the clock of the process it received it from, before it would be able to choose a maximum of the received messages and determine 'when' this message happened.

4. *Fill in the corresponding vector clocks from Figure 1. In this example, the ticking only happens when sending a message. Red are the Vector Clocks that are attached to the messages, black the thread-local vector clocks.*



5. *Read the paper "Dynamic Vector Clocks for Consistent Ordering of Events in Dynamic Distributed Applications". Which problem of vector clocks is solved in the paper?*
The paper proposes dynamic vector clocks as a solution for a case where the number of processes is not constant, not known in advance, or both. It also proposes a way to conduct garbage collection with dynamic vector clocks (assuming the processes will initiate a termination protocol before terminating).

Assuming it is not clear from within the message from which process it was sent, then the receive-side increment is not possible.