Assignment 07

Group A01
Philip Raschke, 0333883
Pietro Rancati, 0368002
Shikhar Singh, 0368042
Paul Mälzer, 0336763

Exercise 7.1: Vector Clocks
If e is a system event, then V(e) defines its vector timestamp and C(e) ∈ N is the sum of its components. If an event e occurs V(e) is computed as introduced in the lecture.

a. Show that C(e) together with the "less than"-relation (<) defines a partial order of the set of system events.

A partial order is a reflexive, transitive and antisymmetric relation R with $R \subseteq M \times M$.
In our case R is the relation "<" and M is the set C(e). Note that we also use the infix notation, i.e. x < y means (x,y) in "<" for x,y in C(e).

First we need to show that "<" is reflexive that means x < x or (x,x) in "<" for all x in C(e). Since "<" is constructed from the cross product of C(e) obviously this conditions holds true for all elements in C(e).
Next we need to show that the relation is transitive which means that if x < y and y < z then x < z. Again, this actually means if (x,y) and (y,z) in "<" then (x,z) in "<). Since x,y and z are natural numbers and the relation "<" is defined according to the mathematical definition of the "<" relation this holds true by definition. The relation "<" can not contain a single element that does not fulfill this condition.
At last, we have to show that the relation is antisymmetric which means if x < y and y < x then x = y (or if (x,y) and (y,x) in "<" then x = y). Analogously to before we can find that this condition holds true since x and y are natural numbers. The condition is necessarily fulfilled.

We showed that the relation "<" is a partial order. Now we want to show that this relation with the function C(e) defines a partial order of the set of system events. As we can see every event e can be mapped to a natural number with the function C. We have shown that with such mapping the conditions for a partial order are fulfilled.

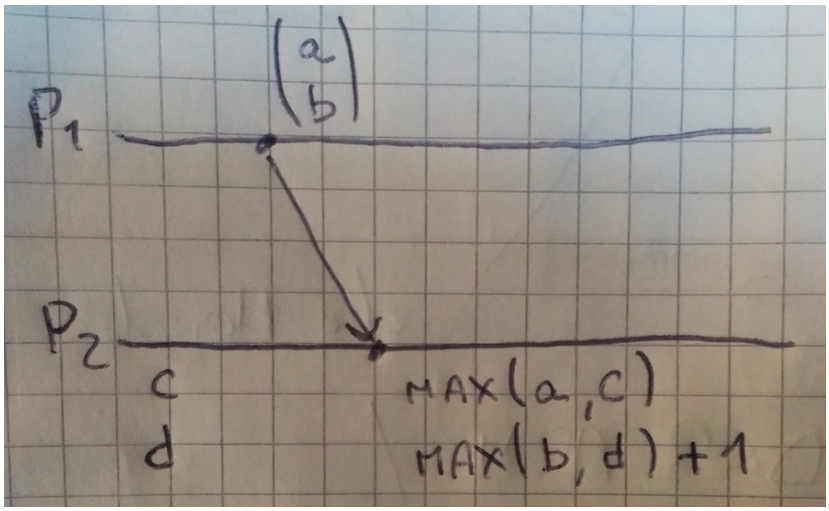
b. Show that the respective logical clock fulfills the clock condition.

Clock condition:
$a \rightarrow b \Rightarrow C(a) < C(b)$

The same time stamp can appear only with two event independent causally of each other.

If the event is causally dependent on a previous event, the sum of the component must be bigger than the sum of component of the previous. This because when a message arrived to a process, carrying also the vector clock, the receiving process compute his new vector clock calculating the maximum, for each row of the vector clock, between the value of the message and his value. It also increment is component in the vector clock by 1.

See the following example of a message from P1 to P2, in delivering the message, the second component of the vector is incremented:



c. Show that the partial order can be extended to a total order by applying process IDs together with the vector timestamp.

See the partial order defined by answer 7a. If an event is causally dependent on another event is C(e) should be > than the other event. If two event are not causally dependent may occur that their C(e) is equal. In this case, in order to have a total order of the event, the process ID of the two process must be compared and the event, that belong to the process with a greater process ID, should be considered happened after the other event in order to have a total order with the vector timestamp.

Exercise 7.2: Relation between Timestamps
Explain the relations between the different timestamps. Why it is not possible to invert given implications?

In case of real clocks, for 2 events 'a' and 'b', if a happens before b and are causally dependent i.e a--> b, then surely C(a) < C(b). However, if two events 'c' and 'd' are such that C(c) < C(d), then surely this does not mean that 'd' causally depends on 'c' i.e. c--> d, since c and d can be two totally unrelated events as well. Moreover, due certain real clock cases where the clocks are not synchronized accurately, C(c) < C(d), does not even implies

that 'c' happened before 'd'. And hence inverse of the clock condition does not hold true for real clocks.

In case of Lamport clocks, a logical clock provides a timestamp to an event. The current value of the logical clock is assigned when an event occurs in a process. This synchronisation in logical clocks fulfill the clock condition. And hence, a-->b => C(a) < C(b) holds true. These logical timestamps however define only a partial order on the set of events.And do not vouch for the inverse of clock condition. This is because if C(a) < C(b) does not mean a-->b. The event a and b might be totally unrelated. And hence the converse of the clock condition is not true for lamport clocks as well.

Vector clocks on the other hand turn the clock condition into an equivalence. Each process maintains a vector timestamp consisting of N counter, one each for all the processes. a--> b => C(a) < C(b). Also, if C(a) < C(b) implies a-->b and if C(b) < C(a) implies b--> a . Hence, !(C(b) < C(a) V C(a) < C(b)) = > a||b.

Exercise 7.3: Distributed Deadlock Detection
Implement the algorithm of Chandy, Misra and Haas for distributed deadlock detection. If a deadlock has been detected, resolve that situation in a simple way.