Time, Clocks, and the Ordering of Events in a Distributed System by Leslie Lamport

CSC 724 Paper review - Vaibhav Singh, vsingh7

January 14, 2019

1 Summary

The paper defines a "happens-before" system and talks about a distributed algorithm for synchronizing a system of logical clocks which can be used to order events following invariant partial ordering in the system. The system is then extended to a system following arbitrary total ordering of events (but which could result in anomalous behavior). In order to remove anomalies caused by arbitrary choosing a total ordering of events, the system is extended to a system with logical clocks, and a bound is found to how close the clocks should be synchronized.

2 Description

The paper first defines a "happens-before" ordering of events as event e1 "happens-before" e2, if (a) e1 occurs before e2 in the same process, or (b) e1 is the send event of a message and e2 is the receive event of the same message, or (c) e1 happens before e' and e' happens before e2. A "logical clock" is defined as a function assigning a number C(i) to any event i in a process. The correctness of a system following logical clocks must ensure the Clock Condition i.e if an event a "happens-before" b, then C(a) is less than C(b). To guarantee this, one can describe a system in which any process P increments the clock for any two successive events. Also, when message passing occurs between processes, process p1 adds its current timestamp to the message, which is then incremented and used by process p2 to set its own local clock. The paper then describes a total ordering of events as events sorted by their timestamps, with events having the same timestamp arbitrarily given an ordering. This system is used to solve the problem of mutual exclusion of a resource among processes. However, examples of "anomalous behavior" is also provided for a system following total ordering, in which message ordering information is kept outside the system. To solve this, the author proposes a system using physical clocks and describes bounds on how close the clocks in the system need to be synchronized.

3 Strong Points

The paper proposes a very novel idea from special relativity and is able to successfully model time ordering in distributed systems using it.

4 Weak Points

The paper assumes that processes do not fail (as there is no way to differentiate a process which is hanging from a process which is failed).

5 Improvement

Systems in which process failure could occur should have been addressed as well, although differentiating processes which are slow with processes which have failed is difficult without having an acknowledgment scheme as well.