

Distributed Systems

Exercises

Werner Nutt

Bully Algorithm

1. Sketch the Bully Algorithm. Remember there are 3 types of messages:
 - election, vote, coordinator.
2. What happens if two processes notice at the same time that the leader has crashed?
3. In the Bully algorithm, a recovering process starts an election and will become the new coordinator if it has a higher identifier than the current leader.
 - Is this a feature that is necessary for the algorithm to work correctly?
 - Under which circumstances can it be dropped?

Synchronous vs. Asynchronous Distributed Systems

1. Define: When is a distributed system
 - synchronous
 - asynchronous?
2. Which of the following distributed algorithms functions only in a synchronous system?
 - Ring-based leader election
 - Bully algorithm
3. Does one of them only function in an asynchronous system?

Lamport's Logical Clocks

Consider three distributed processes P1 , P2 , and P3 . The processes are involved in the events a, b, . . . , k listed below, which happen at specific points in time, specified as “wall-clock time” (WCT) and measured in ms:

At 0ms WCT:

- a: P2 sends message m1 to P1
- b: P3 reads a startup file

At 100ms WCT:

- c: P1 opens a file containing a user profile
- d: P3 sends message m2 to P2

At 200ms WCT:

- e: P1 receives message m1
- f : P2 receives message m2

Lamport's Logical Clocks (cntd)

At 300ms WCT:

g: P1 sends message m3 to P2 and P3

At 400ms WCT:

h: P2 receives message m3

At 500ms WCT:

i: P2 sends message m4 to P1

At 600ms WCT:

j: P1 receives message m4

At 600ms WCT:

k: P3 receives message m3

Lamport's Logical Clocks (cntd)

1. Which of these events are related by Lamport's "happened before" relation (in the lecture denoted as " $E1 \rightarrow E2$ " if event $E1$ happened before event $E2$)? Draw a directed graph where the events are vertices and where there is an edge from $E1$ to $E2$ if $E1$ happened before $E2$.
2. Associate to each event a logical timestamp according to the logical clock algorithm. Use process IDs to break possible ties. What is the linear order of events induced by these timestamps?

Concurrent Events wrt “Happens-Before”

Consider three processes P1, P2, P3, where sending and receiving messages are the only events.

- Can you construct an example of two events that are concurrent?

Ring-based Mutual Exclusion and Ordering Property

Consider the ring-based algorithm for mutual exclusion.

Assume that the following are possible events in a process:

- Sending/receiving a message
 - Entering a state where a shared resource needs to be accessed (“request” event).
-
- Can it happen that process P1 needs to access the shared resource before process P2 (in “happens-before” order), but gets access only after P2?

Ordering and Single-Threaded Processes

Consider a scenario where the central-server algorithm for mutual exclusion is applied.

Suppose that all processes requesting tokens are single-threaded.

- Is it possible that the order in which requests are served contradicts the “happens-before” ordering?

Threads and Efficiency Gains

A file server uses caching, and achieves a hit rate of 80%. File operations in the server cost 5 ms of CPU time when the server finds the requested block in the cache, and take an additional 15 ms of disk I/O time otherwise.

Explaining any assumptions you make, estimate the server's throughput capacity (average requests/sec) if it is:

- single-threaded;
- two-threaded, running on a single processor;
- two-threaded, running on a two-processor computer.