CALIFORNIA INSTITUTE OF TECHNOLOGY
Computing and Mathematical Sciences

CS 142

R. Murray, M. Chandy
Fall 2019

Homework Set #4

Issued: 23 Oct 2019
Due: 30 Oct 2019

Note: Please keep track of the number of hours that you spent on this homework set (including reading) and enter this when you turn in your homework on Moodle.

1. Let $A \parallel B$ represent concurrency between two events A and B, which we define formally as

$$A \parallel B = \neg(A \to B) \wedge \neg(B \to A),$$

where $\to$ means that event A happens before B (note that we are overloading the operator $\to$ depending on whether it acts on predicates or events). Is concurrency transitive? That is, does the following property hold?

$$(A \parallel B) \wedge (B \parallel C) \implies (A \parallel C)$$

Prove or give a counterexample.

2. Consider the program *LogicalClock* in Section 5.3.4 of Sivilotti. The algorithm for each agent j is given by

Program *LogicalClock* j
var j, k : processes
$ch(j,k)$: channel from j to k
m : message
A, B : events
$clock(j)$: logical time of j
$time(A)$: logical time of A
initially $clock(j) = 0$
assign

$$\begin{array}{lll} \text{local event } A \to & clock(j) := clock(j) + 1 & (1) \\ & ; \quad time(A) := clock(j) & \\ [] \quad \text{send event } A \to & clock(j) := clock(j) + 1 & (2) \\ (\text{to } k) & ; \quad time(A), time(m) := clock(j), clock(j) & \\ [] \quad \text{recv event } A \to & clock(j) := \mathbf{max}(time(m), clock(j)) + 1 & (3) \\ (m \text{ from } k) & ; \quad time(A), ch(j,k) := clock(j), tail(ch(j,k)) & \\ & & (4) \end{array}$$

Show that the following property is an invariant of the program:

(a) $(\forall A, j : A \text{ occurs at } j : \text{time}(A) \leq \text{clock}(j))$

(b) $\wedge$ $(\forall m, j, k : m \in \text{ch}(j,k) : \exists A : \text{A occurs at } j : \text{time}(A) = \text{time}(m))$

(c) $\wedge$ $(\forall A, B :: A \to B \implies \text{time}(A) < time(B)$

3. A system has 3 agents, P_1, P_2, P_3 and channels between all pairs of agents. The system has 2 indivisible tokens. A message can contain any number of tokens. An invariant is that the total number of tokens held by processes and in channels is 2.

   Supposed that at some point in the computation, agents P_1 and P_2 hold one token each, P_3 has no tokens, and all the channels are empty. At this point the vector clocks for P_1, P_2, P_3 are $[1, 0, 0]$, $[0, 2, 0]$, and $[0, 0, 0]$, respectively.

   At some later point, P_3 holds both tokens. Various scenarios can take the system from the earlier point to the later point. For example, P_2 may send a token to P_3 which sends a token to P_1 which sends two tokens to P_3.

   (a) Considering all possible scenarios, what are the minimal vector clock values for the point at which P_3 holds both tokens?

   (b) Would the answer change if P_3 originally had a token?

   (c) Would the answer change if you were told each process had sent at least one message?

   Explain your yes/no answers with a few sentences or an example space-time diagram.

4. You are given a distributed system which is represented by an undirected graph in which the vertices represent agents (also called processes) and the undirected edges represent channels. Thus if i and j are neighbors in the graph then there is a message channel from i to j, and another message channel from j to i. The graph is strongly connected, i.e., there exists a path from every vertex to every other vertex.

   (a) Develop an algorithm initiated by some agent, say agent 0, by which the agent discovers the number of vertices in the graph. Make sure you write out the program for both the agent 0 (which is not included in Sivilotti) and other agents in the UNITY format as in Section 6.5 of Sivilotti.

   (b) Prove that your algorithm terminates execution eventually. If you use an algorithm that is proved in the book you don't need to give the proof in your homework. For example, you don't need to prove that a diffusing computation terminates.

   (c) Prove that when it terminates, agent 0 has the correct counts. For this purpose it is helpful to define an invariant. You should provide similar level of rigor to Sivilotti 6.6.1.