CALIFORNIA INSTITUTE OF TECHNOLOGY
Computing and Mathematical Sciences

CS 142

R. Murray, M. Chandy
Fall 2019

Homework Set #3

Issued: 16 Oct 2019
Due: 23 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. For each of the properties below, provide a formal (detailed) proof or give a counterexample.

   (a) Strengthening: $\textbf{transient}(P) \wedge [P' \implies P] \implies \textbf{transient}(P')$

   (b) Weakening: $\textbf{transient}(P) \wedge [P \implies P'] \implies \textbf{transient}(P')$

   (c) Determine whether the program

   $$n \leq 2 \rightarrow n := n + 1$$

   has the property

   $$\textbf{transient}(n = 0 \vee n = 1).$$

2. For each of the statements below, indicate whether the statement is true or false. If true, give a formal (detailed) proof. If false, give a counterexample.

   (a) $P \leadsto \textbf{true}$

   (b) $\textbf{false} \leadsto P$

   (c) $P \leadsto P$

   (d) Weakening: $(P \leadsto Q) \wedge [Q \implies Q'] \implies P \leadsto Q'$

   (e) Strengthening: $(P \leadsto Q) \wedge [P' \implies P] \implies P' \leadsto Q$

   (f) Stable strengthening: $\textbf{stable}(P) \wedge \textbf{transient}(P \wedge \neg Q) \implies P \leadsto (P \wedge Q)$

3. Give a full proof for the sorting algorithm in Sivilotti, Section 4.3 (similar to the proof provided for FindMax in Section 4.2).

4. [Nondeterministic Iteration – Shortest Paths]

   This problem considers an algorithm to find the shortest path between every pair of vertices in a finite directed graph. Let W be the edge-weight matrix, i.e., $W[j,k]$ is the weight of edge (j,k). Weights are real numbers. Assume that the graph is completely connected, and therefore $W[j,k]$ exists for all j,k. Also $W[j,j] = 0$ for all j. The graph has no cycles of negative weight.

   Let D be a matrix with the same dimensions as W. The proposed algorithm is as follows:

   - **initially**: $D = W$

- There is a command for every triple (i, j, k) of vertices, and the command is:

$$\textbf{IF } D[i,k] > D[i,j] + D[j,k] \textbf{ THEN} D[i,k] := D[i,j] + D[j,k]$$

The claim is that the algorithm will terminate with D being the matrix of shortest path lengths, i.e., $D[j,k]$ will be the length of the shortest path from vertex j to vertex k.

(a) Prove that the conjunction of following two predicates is an invariant for the above program.
   - $\forall j, k : D[j,k] \leq W[j,k]$
     and
   - for all j, k, $D[j,k]$ is the length of some path from vertex j to vertex k

(b) Does the program (algorithm) have a fixed point at which the invariant property from the previous part holds. If yes, propose the fixed point and prove it. Recall that a state is a fixed point if there exists no action in the program that changes the state.

(c) For each pair of vertices (i, k), consider the set of the lengths of all possible paths from i to k with length less than or equal to $W[i,k]$. (Note this is a set of path lengths, not a set of paths.)

Let $L[i,k]$ be the set ordered in decreasing order. For each state of the algorithm, you have shown that $D[i,k]$ is the length of some path from i to k. So, $D[i,k]$ is a value in the list $L[i,k]$. Let $f_{i,k}(D[i,k])$ be the index of $D[i,k]$ in $L[i,k]$.

Let

$$F(D) = \sum_{i,k} f_{i,k}(D[i,k]).$$

Explain why F is a metric for this program. A formal proof at a similar level of detail to Sivilotti's proofs in Chapter 4 (after FindMax) is sufficient, but not required. However, you should provide sufficient details that it is clear there are no missed steps.

(d) Write down the invariant and fixed point condition. No proof is necessary since you've already proved that. Prove that the program terminates at the fixed point using he invariant, the fixed point condition, and the metric above.