WS 2019 / 2020 21.01.2020

Exercises to the lecture
Concurrency Theory
Sheet 8

Dr. Prakash Saivasan
Peter Chini Delivery until 28.01.2020 at 12:00

Exercise 8.1 (Sequential Consistency I)
In the *strong* memory model *Sequential Consistency* (SC) we assume that accesses to the memory are atomic. Formally, the transition relation $\rightarrow_{\mathrm{SC}}$ is defined similarly to $\rightarrow_{\mathrm{TSO}}$ but the rule (STORE) is replaced by the rule (SCSTORE).

$$\text{(SCSTORE)}\ \frac{<inst>= \mathrm{mem}[r] \leftarrow r', a = val(r), v = val(r')}{(pc, val, buf) \rightarrow_{\mathrm{SC}} (pc', val[a := v], buf)}.$$

Note that this means that the buffer is never used in a computation.

a) Argue that the following statement is true: There is a correspondence between all executions of a multi-threaded program running under SC and the single execution of all single-threaded programs obtained by shuffling the source code of the threads.

b) Let P be a program and $fen(P)$ the program obtained from P by inserting an mfence after each store operation. Argue that P executed under SC has the same behavior as $fen(P)$ executed under TSO.

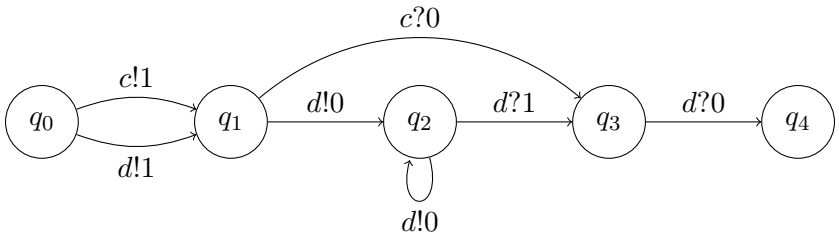
Exercise 8.2 (Sequential Consistency II)
Consider the *control state reachability problem* for SC:

SC reachability	
Input:	Program P over DOM and a program counter pc.
Question:	Is there a computation $cf_0 \rightarrow_{\mathrm{SC}}^{*} (pc, val, buf)$ for some val and buf?

Show that SC reachability is in PSPACE.

Hint: Give an algorithm solving the problem. The algorithm is allowed to use at most polynomial space. Hence, one can afford storing a pointer into each thread.

Exercise 8.3 (Backward Search)
Consider the following lossy channel system.



Use the backward search to decide whether the configuration $(q_4, \begin{pmatrix} 0 \\ \varepsilon \end{pmatrix})$ is reachable.

Delivery until 28.01.2020 at 12:00