Java Model

Sarita Adve
UIUC

Definitions:

Programs: Same as for Manson/Pugh (referred to as M/P below).

Execution trace: It is a set of instruction instances (with GUIDs) from the program, including values read and written. We do not place any restrictions on these instruction instances except that (1) a memory read should return the value of some write to the same location in the execution, and (2) given the values returned by shared memory reads in the execution, a correct uniprocessor should be able to generate the set of instruction instances of any single thread. We also assume that locks and unlocks on the same monitor are "paired" (i.e., we can identify a unique last unlock before each successful lock), and an instruction instance has at most one memory action (read, write, lock, or unlock).

[Note: M/P require all of the above aspects to be formally mentioned in some form somewhere **-** either in the definition of execution trace, unithread semantics, or happens-before relation.]

Observable behavior of an execution trace: Same as M/P (i.e., the values returned by all reads).

Happens-before relation – hb: It is defined on memory instruction instances in an execution trace as follows: I hb J if:
- I hb K hb J for some K, or
- I is before J by program order, or
- I is a volatile write and J is a volatile read that returns the value written by I, or
- I is a monitor unlock and J is the "paired" lock

[Note: Minor changes are possible for the hb relation **-** I can use whatever M/P want to use.]

Correctly synchronized program: A program is correctly synchronized if the following is true for each SC execution trace of the program. If memory instruction instances I and J conflict, then either I hb J, or J hb I, or both I and J are volatile/monitor actions. (An execution trace is SC if its observable behavior is SC.)

[Note: Again, M/P will need to formalize the above as well, and I can use whatever variation they use.]

Flow dependence relation - fd: It is defined on instruction instances in an execution trace as follows: I fd J if:
- I fd K fd J for some K or
- J reads a value written by I, unless J writes the same value in all possible execution traces of the program

[Note: the analog for the above in M/P is the use of PUR's and proposed use of an "informal statement regarding causal loops."]

Memory model: An execution trace E of a program P is valid if it obeys the following properties:
(1) if P is correctly synchronized, then E's observable behavior is the same as that of an SC execution of P;
(2) E's flow dependence relation is acyclic; and
(3) the values read in E are consistent with hb; i.e., if R is a read that returns the value of a write W, then we cannot have R hb W and we cannot have a W' to the same location where W hb W' hb R in E.