# Lower Bound Techniques in Distributed Computing: Home Assignment 2

May 8, 2006

Due: May 22, 2006

1. Consider the primitive $k$SC, which extends the LL/SC primitives from home assignment 1, defined as follows:

   - *LL*($x$), returns the value of the object $x$, and
   - $k$*SC*($v$,$x$,$x_1, \ldots, x_k$), which either returns true and changes the value of the object $x$ to $v$ or returns false and leaves the value of the object unchanged. It returns true if and only if no other process has performed a $k$*SC* operation on one of the objects $x_1, \ldots, x_k$ that returned true since $p$ last performed *LL*.

   Note that $k$SC only returns a Boolean indication and does not "read" the values of the objects it accesses.

   Prove an $\Omega(\log_2 n)$ lower bound on the step complexity of the COLLECT problem (studied in Lecture 4).

   **Note:** you might find it easier to work with $k$CAS ($k$-compare&swap) instead of $k$SC.

2. Prove a shared memory version of the lower bound for the $s$-session problem. Specifically, prove an $\Omega((s-1)\log_b n)$ lower bound on the step complexity of any $s$-session algorithm in synchronous executions, under the assumption that the write-contention is at most $b$.

   **Note:** An easier lower bound to prove is when the *access*-contention is at most $b$.

3. Recall the $\Omega(m)$ lower bound on the step complexity of UPDATE operations.

   (a) Show how to modify the formal definition of a flippable execution so that a variant of the lower bound proof can be applied to *binary* single-writer snapshot objects; namely, a snapshot object in which every component holds a value in $\{0, 1\}$.

   (b) What is the best lower bound you can get when the number of scanners $s$ and the number of updaters $u$ are fixed in advance? Your (nonasymptotic) answer should be expresses as a function of $s$ and $u$.