Correctness proof for SRSW multi-valued register simulation

November 30, 2011

The proofs provided herein are taken from the book "Distributed Computing", by Hagit Attiya and Jennifer Welch, with some changes, including terminology/notation adaptation.

1 Multi-valued SRSW register from binary SRSW registers

The wait-freedom of the simulation is clear from the code. In this section, we prove that the algorithm shown in class for implementing a multi-valued SRSW register from binary SRSW registers is linearizable.

We call reads/writes from base objects low-level reads/writes. We call reads/writes from a simulated object high-level reads/writes. Let $E$ be an execution. We say that a low-level read operation $r$ of $B[v]$ reads from a low-level write operation $w$ to $B[v]$, if $w$ is the latest write to $B[v]$ that precedes $r$ in the linearization of the operations on $B[v]$. We say that a high-level read $R$ in $E$ reads from a high-level write $W$, if $R$ returns $v$ and $W$ contains the write to $B[v]$ that $R$’s last read of $B[v]$ reads from.

We linearize the operations in $E$ by (1) placing all the high-level write operations in the order they were performed, and (2) placing any high-level read operation between the high-level write operation from which it read (or before all high-level write operations, if there was none) and
the following write operation (if any); when doing that, we consider the read operations in the order that they appear in $E$. Let the resulting linearization be $\pi$. From construction, $\pi$ meets the sequential specification. We need to prove that $\pi$ respects the partial order of high-level operations induced by $E$. The following cases need be considered.

1. The order between writes is clearly maintained.

2. If a read operation $R$ precedes in $E$ a write operation $W$ then, clearly, $R$ is linearized before $W$, as it cannot read in $E$ a value written by $W$ or by a later write operation.

3. We still need to prove that the order of read operations is maintained.

4. We need to prove that if a write operation $W$ precedes in $E$ a read operation $R$, then $W$ also precedes it in $\pi$.

**Lemma 1** Consider two values $u$ and $v$, with $u < v$. If, in execution $E$, read operation $R$ returns $v$ written by write operation $W$ and $R$ reads 0 from $B[u]$ during its upward scan, written by some write operation $W_1 \neq W$, then $W$ follows $W_1$.

**Proof:** Suppose in contradiction that $W$ precedes $W_1$ and let $v_1$ be the value written by $W_1$. Since $W_1$ writes 1 to $B[v_1]$ and then does a downward scan, $v_1 > u$ holds. Also, $v_1 < v$, since otherwise $W_1$ would overwrite $W$’s write to $v$ (before it is read by $R$), contradicting the fact that $R$ reads from $W$.

Thus, in $R$’s upward scan, it reads $B[v_1]$ after $B[u]$ and before it reads $B[v]$. $R$ must read a 0 from $B[v_1]$, since otherwise it would not return $v$. Consequently there must be another write $W_2$, that follows $W_1$, that writes 0 to $B[v_1]$ before $R$’s read of $B[v_1]$. Let $v_2$ be the value written by $W_2$. Note that $v_1 < v_2 < v$ must hold by the same reasoning as before. Similarly, $W_2$ must be followed by a write $W_3$ that writes $v_3$, with $v_1 < v_2 < v_3 < v$, and so on. Thus there exists an infinite increasing sequence of integers $v_1, v_2, v_3, \cdots$, all of which are smaller than $v$. This is a contradiction. \[\square\]
Lemma 2  Proof of case (4): suppose that Write operation $W$ precedes a read $R$ in execution $E$. Then $W$ is put before $R$ in $\pi$.

Proof: To obtain a contradiction, suppose that $W$ is put after $R$ in $\pi$. Then $R$ reads from some Write operation $W'$ that precedes $W$. Let $W$ write $v$ and $W'$ write $v'$. Thus $R$ returns $v'$.

Assume first that $v' \leq v$. Then $W$ overwrites the write to $B[v']$ made by $W'$ before $R$ begins and $R$ cannot read from $W'$.

Now consider the case $v' > v$. $W$ writes 1 in $B[v]$. Since $R$ does not see this 1 and stop at $B[v]$ during its upward scan, there must be a Write $W''$ after $W$ containing a write of 0 to $B[v]$ that $R$'s read of $B[v]$ reads from. From Lemma 1, the value returned by $R$ was written by a Write operation that follows $W''$, hence the value returned by $R$ is by a Write operation that follows $W'$. Thus $R$ cannot read from $W'$. This is a contradiction.

Lemma 3  Proof of case (3): suppose that Read operation $R_1$ precedes Read operation $R_2$ in execution $E$. Then $R_1$ is put before $R_2$ in $\pi$.

Proof: To obtain a contradiction, suppose $R_1$ is put after $R_2$ in $\pi$, though it precedes $R_2$ in $E$. This implies that $R_1$ reads from a Write $W_1(v_1)$ that follows the Write $W_2(v_2)$ from which $R_2$ reads. We consider 3 cases.

- If $v_1 = v_2$, then when $W_1$ writes 1 to $B[v_1]$ it overwrites the 1 that $W_2$ wrote to $B[v_2]$ earlier. Thus $R_2$ cannot read from $W_2$, a contradiction.

- Assume $v_1 > v_2$. Since $R_2$ reads from $W_2$, no write to $B[v_2]$ is linearized between $W_2$’s write of 1 to $B[v_2]$ and $R_2$’s last read of $B[v_2]$. Since $R_1$ reads from $W_1$, $W_1$’s write of 1 to $B[v_1]$ precedes $R_1$’s last read of $B[v_1]$. So $B[v_2]$ has value 1 starting before $R_1$ does its last read of $B[v_1]$ and ending after $R_2$ does its last read of $B[v_2]$. But then $R_1$’s read of $B[v_2]$ during its downward scan would return 1, not 0, hence $R_1$ should return $v_2$ or a smaller value, a contradiction.
• Assume \( v_1 < v_2 \). Since \( R_1 \) reads from \( W_1 \), \( W_1 \)'s write of 1 to \( B[v_1] \) precedes \( R_1 \)'s last read of \( B[v_1] \). Since \( R_2 \) returns \( v_2 > v_1 \), \( R_2 \)'s first read of \( B[v_1] \) must return 0. So there must be another Write after \( W_1 \) containing a write of 0 to \( B[v_1] \) that \( R_2 \)'s read of \( B[v_1] \) reads from. Then, from Lemma 1, \( R_2 \) must return a value written by a Write operation that follows \( W_1 \). Thus it cannot return the value written by \( W_2 \).