Appendix A

Proof of correctness for the Optimized one-instruction-update algorithm of Chapter 2

Define $D(m)$, where m is a memory entry, to be the depth of the longest (i.e., deepest) prefix p that covers m. Also define $L(p)$ to be the length of a prefix p.

***Claim C1*:** For all prefixes p, $PS(p) \leq MS(p)$.
Proof: By definition.

***Claim C2*:** For all prefixes p, $ME(p) \leq PE(p)$.
Proof: By definition.

***Claim C3*:** For all prefixes p and q, either of the following hold:
(1) $PS(p) \leq PS(q) \leq PE(q) \leq PE(p)$ i.e. q is completely contained in p.
(2) $PS(q) \leq PS(p) \leq PE(p) \leq PE(q)$ i.e. p is completely contained in q.
(3) $PS(p) \leq PE(p) \leq PS(q) \leq PE(q)$ i.e. p and q are completely non-overlapping.

Proof: Follows from the definitions of *PS* and *PE* and the basic *parenthesis* property of prefixes.

***Claim C4*:** For all memory entries m such that $PS(p) \leq m < MS(p)$, $D(m) > L(p)$.
Proof: As $MS(p)$ is the first memory entry covered by prefix p, all memory entries between its prefix start and memory start must be covered by deeper (i.e., longer) prefixes.

Claim C5: If a prefix p is deeper than q and $PS(q) \leq PS(p) < MS(q)$, then $MS(q) > PE(p)$; i.e. p has to end (prefix end) before $MS(q)$.
Proof: Follows from the fact that q cannot actually start in memory before any deeper prefix has completely ended.

Now, let the update instruction passed on to the hardware by the processor be *Update(m,Y,Z)*. Before any updates, as m is the first memory entry chosen to be updated by the processor,

$D(m) \leq Y$. If while executing the algorithm, hardware encounters a start-marker marking a new prefix, say q, on a memory entry $m2$, $m2$ equals $MS(q)$ but it may or may not equal $PS(q)$. We will show that $L(q) > Y$ in both cases.

Case (S1): $m2 = PS(q) = MS(q)$
Proof: Since the hardware is not done scanning, it has not yet encountered $PE(p)$. As it has encountered $m2 (= PS(q))$, (C3) tells us that q is wholly contained in p, and so $L(q) > L(p) = Y$.

Case (S2): $PS(q) < m2 = MS(q)$
There are two subcases possible within this case depending upon where $PS(q)$ lies with respect to m:
Case(S2.1) $m < PS(q) < m2 = MS(q)$
Clearly in this case, prefix q is wholly contained in prefix p, and so $L(q) > Y$.
Case (S2.2) $PS(q) \leq m < m2 = MS(q)$. Clearly in this case, p is deeper than q (as m needs to be updated, and m lies in between $PS(q)$ and $MS(q)$). By (C3) and (C5), $PE(p) < MS(q)$, and therefore the hardware should have stopped scanning before reaching $m2$. This subcase is thus not possible at all.
The correctness of the update algorithm now follows immediately. If the hardware, while scanning memory entries encounters a start-marker, it indicates the start of a prefix which is necessarily deeper than Y, and hence is not to be updated. This is exactly what the algorithm does. By updating only when DC equals 1, it ensures that a memory entry is updated only if it had a prefix shallower than Y covering it before the update. ❑