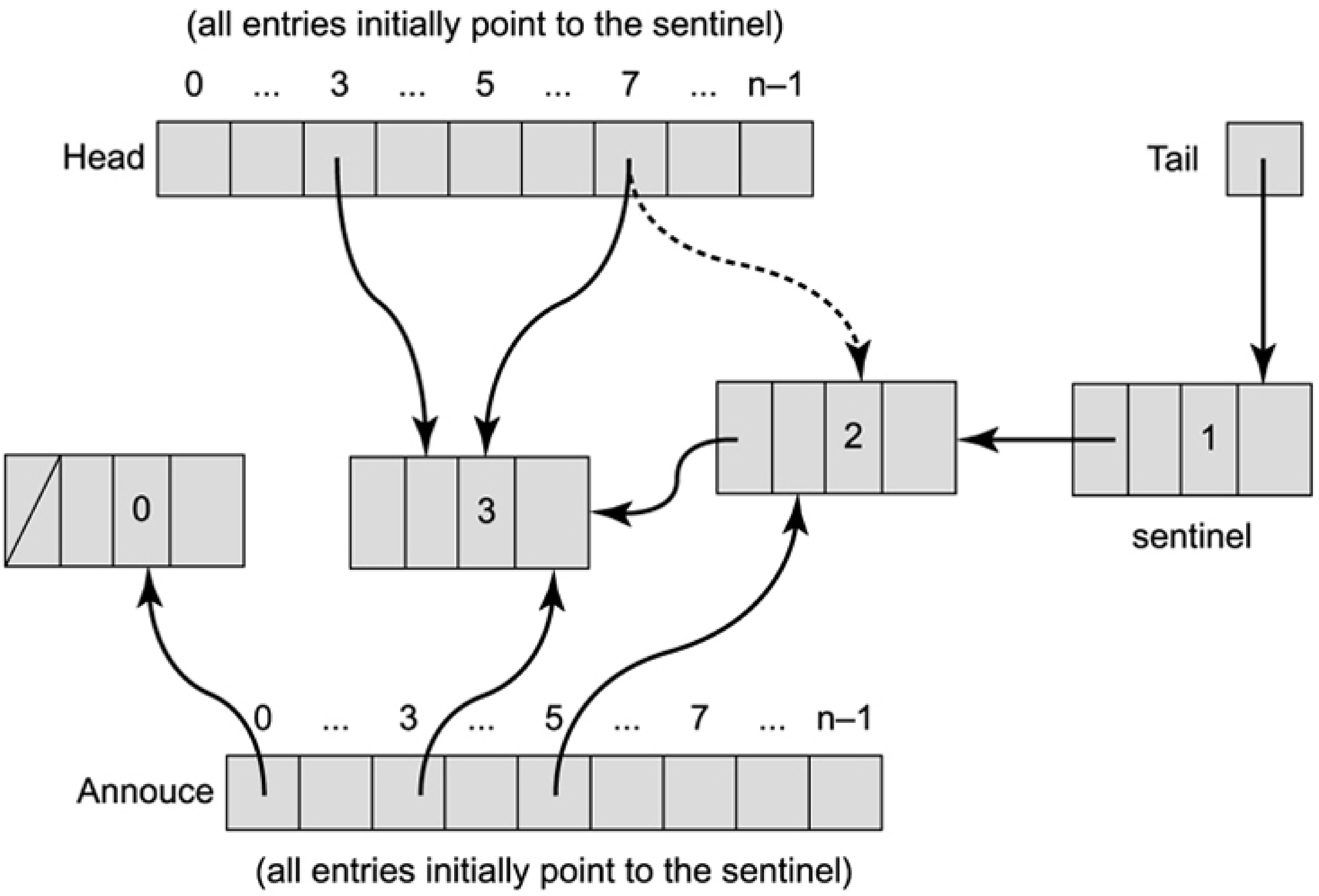

(all entries initially point to the sentinel)
0 ... 3 ... 5 ... 7 ... n–1
Head
Tail
2
1
0
3
sentinel
0 ... 3 ... 5 ... 7 ... n–1
Annouce
(all entries initially point to the sentinel)

FIGURE 6.7 Execution of the wait-free universal construction. Thread 5 announces its new node and appends it to the log, but halts before adding it to the head[] array. Thread 7 does not see thread 5's node in the head[] array. Since thread 2 (whose ID is (before.seq + 1) mod n) is not trying to add a node, thread 7 tries to add its own node. However, it loses the consensus on the sentinel node's decideNext object since thread 5 already won. Thread 7 therefore completes updating the fields of thread 5's node, setting the node's sequence number to 2, and adding the node to the head[] array. Note that thread 5's own entry in the head[] array is not yet set to its announced node. Next, thread 3 announces its node and then pauses before entering the main loop. Thread 7 now successfully helps thread 3, appending its node and setting its sequence number to 3. Now thread 3 wakes up. It does not enter the main loop because its node's sequence number is nonzero, but will update the head[] array and compute its output value using a copy of the sequential object.