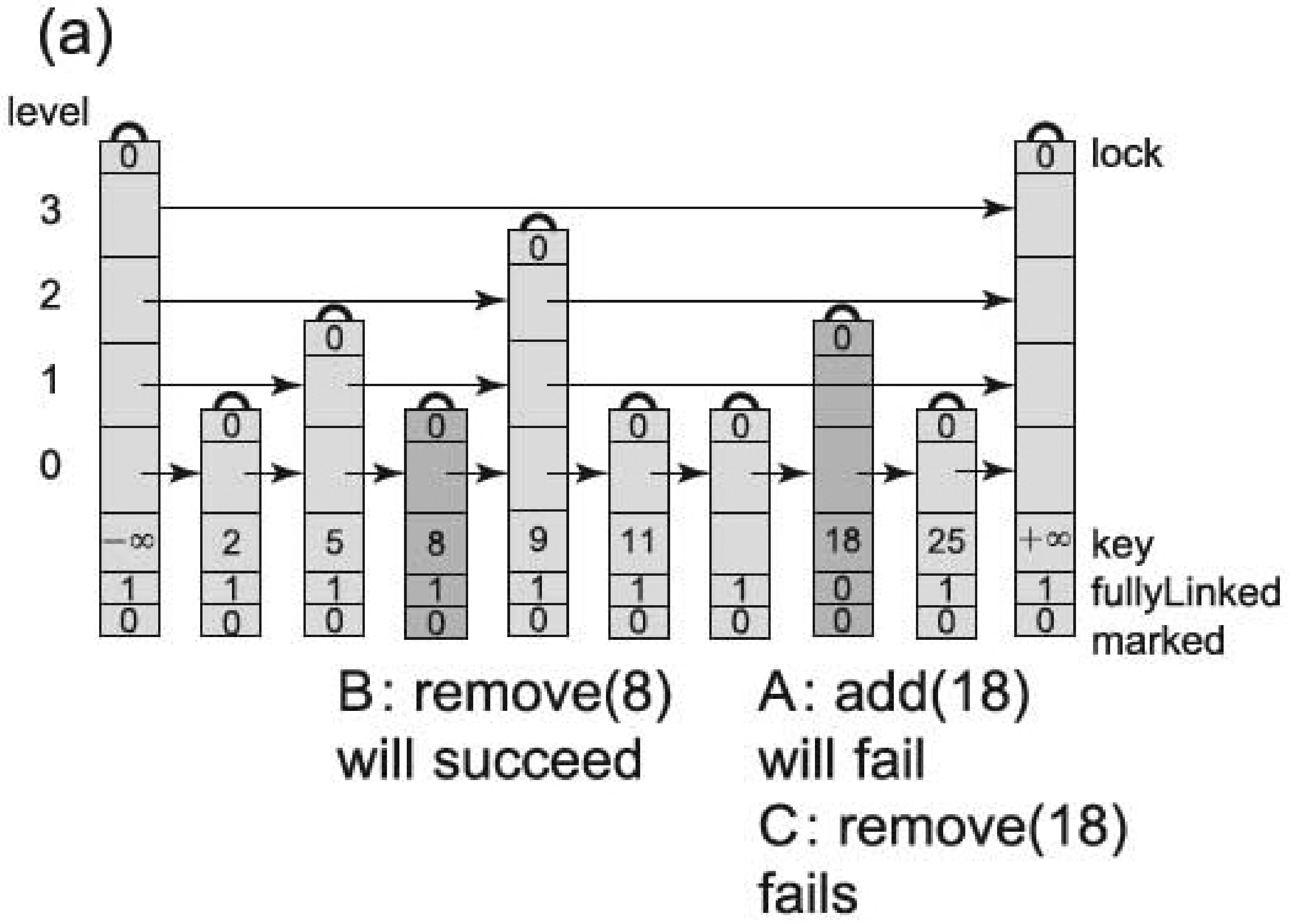

FIGURE 14.3A The LazySkipList class: failed and successful add() and remove() calls. In part (a), the add*(*18*)* call finds the node with key 18 unmarked but not yet fullyLinked. It spins waiting for the node to become fullyLinked in part (b), at which point it returns *false*. In part (a), the remove*(*8*)* call finds the node with key 8 unmarked and fully linked, which means that it can acquire the node's lock in part (b). It then sets the mark bit, and proceeds to lock the node's predecessors, in this case, the node with key 5. Once the predecessor is locked, it physically removes the node from the list by redirecting the bottom-level reference of the node with key 5, completing the successful remove(). In part (a), a remove*(*18*)* fails, because it found the node not fully linked. The same remove*(*18*)* call succeeds in part (b) because it found that the node is fully linked.