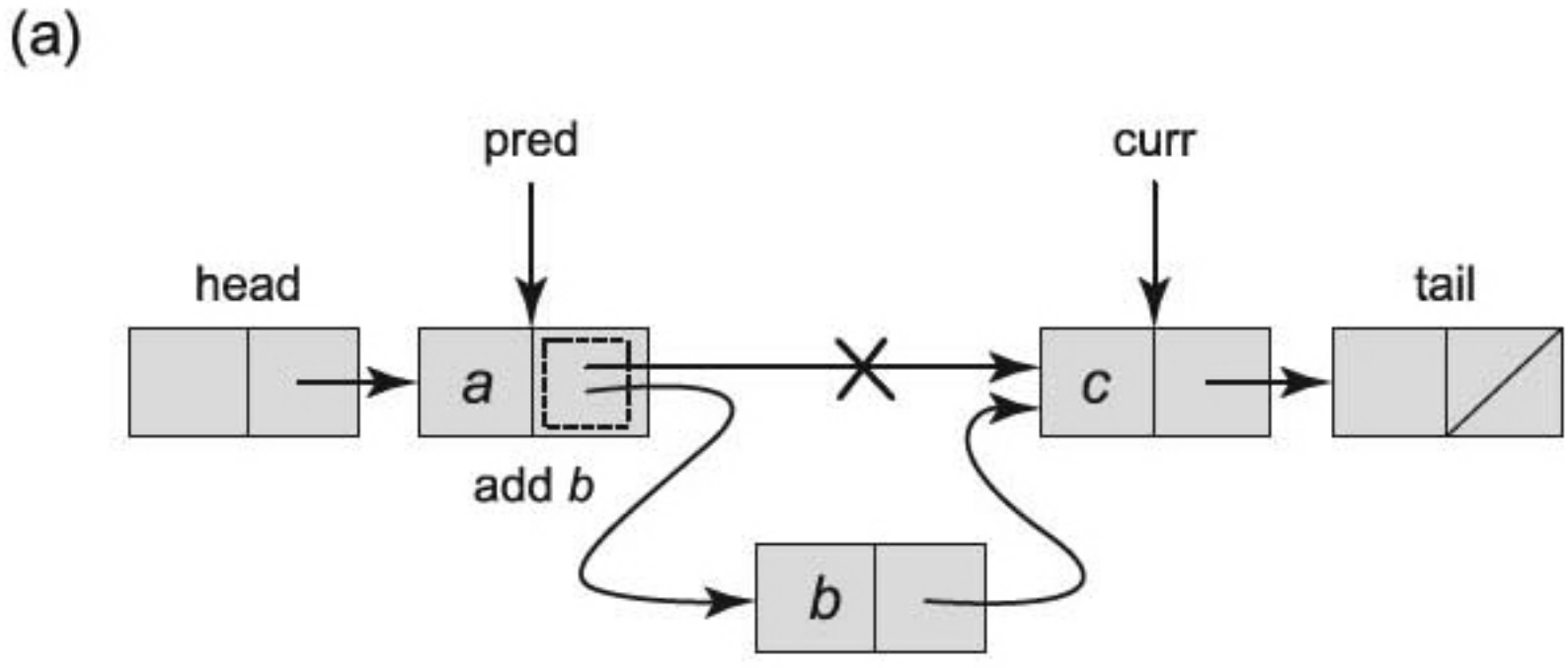

FIGURE 9.3A A sequential Set<> implementation: adding and removing nodes. In part (a), a thread adding a node *b* uses two variables: curr is the current node, and pred is its predecessor. The thread moves down the list comparing the keys for curr and *b*. If a match is found, the item is already present, so it returns *false*. If curr reaches a node with a higher key, the item is not in the set, so it sets *b*'s next field to curr, and pred's next field to *b*. In part (b), to delete curr, the thread sets prged's next field to curr's next field.