```
107   boolean find(T x, Node<T>[] preds, Node<T>[] succs) {
108     int bottomLevel = 0;
109     int key = x.hashCode();
110     boolean[] marked = {false};
111     boolean snip;
112     Node<T> pred = null, curr = null, succ = null;
113     retry:
114       while (true) {
115         pred = head;
116         for (int level = MAX_LEVEL; level >= bottomLevel; level--) {
117           curr = pred.next[level].getReference();
118           while (true) {
119             succ = curr.next[level].get(marked);
120             while (marked[0]) {
121               snip = pred.next[level].compareAndSet(curr, succ,
122                                                     false, false);
123               if (!snip) continue retry;
124               curr = pred.next[level].getReference();
125               succ = curr.next[level].get(marked);
126             }
127             if (curr.key < key){
128               pred = curr; curr = succ;
129             } else {
130               break;
131             }
132           }
133           preds[level] = pred;
134           succs[level] = curr;
135         }
136         return (curr.key == key);
137       }
138   }
```

FIGURE 14.13 The LockFreeSkipList class: a more complex find() than in LazySkipList.