

#### 11.2-4.

Suggest how to allocate and deallocate storage for elements within the hash table itself by linking all unused slots into a free list. Assume that one slot can store a flag and either one element or a pointer or two pointers. All dictionary and free-list operations should run in  $O(1)$  expected time. Does the free list need to be doubly linked, or does a singly linked free list suffice?

**Answer.**

#### Data structures

The flag in a slot indicates whether the slot is free or not.

- Free slots are organized in a doubly linked free list. So a free slot consists of a flag as well as two pointers.
- Besides the flag, a used slot contains an element and a pointer to its successor (possibly NIL) which contains the next element hashes to this slot.

#### Operations

- **Insertion:**
  - If the new element  $e$  hashes to a free slot  $s$ , store the  $e$  as well as a pointer to NIL to  $s$  and remove  $s$  from the free list. The free list must be doubly linked so that the deletion can be done in  $\Theta(1)$  time.
  - If the new element  $e$  hashes to a used slot  $u$ , which contains an element  $d$ . Check to see if  $d$  belongs to  $u$ , that is, whether it hashes to  $u$  or not.
    - If  $u$  is the slot  $d$  hashes to, add  $e$  to the chain of elements in  $u$ . To do so, allocate a slot  $v$  for  $e$  from the free list and add  $v$  at the head of chain pointed to by  $u$ .
    - If not,  $d$  is part of another slot's chain. Transplant  $d$  and all the pointers related to it to a newly allocated slot  $v$ . Then insert  $e$  to  $u$  as if it is an empty slot. (To update the pointer to  $u$ , go down from the slot  $d$  hashes to find its predecessor.)
- **Deletion:** Let  $u$  denote the slot to which the to be deleted element  $e$  hashes.
  - If  $e$  is the only element hashes to  $u$ , just free the slot  $u$ , returning it to the head of the free list.
  - If  $e$  is the first one in the chain of elements that hashes to  $u$ , transplant its successor to  $u$  and free its slot.
  - If  $e$  is not the first one in the chain of elements that hashes to  $u$ , update the pointer to  $e$  slot to point to  $e$ 's successor and free  $e$ 's slot.
- **Searching:** Check and follow the chain of pointers from the slot to which the key hashes until the desired element was found.

---

\*. This solution is digested from selected solution posted on the website of MIT Press, for more information see [http://mitpress.mit.edu/sites/default/files/titles/content/9780262033848\\_Solutions\\_to\\_Exercises\\_and\\_Problems.pdf](http://mitpress.mit.edu/sites/default/files/titles/content/9780262033848_Solutions_to_Exercises_and_Problems.pdf)

All these dictionary and free-list operations take average-case time  $O(1)$ . The reason is similar to that version in the textbook: The expected time to search the chain is  $O(1 + \alpha)$  in which  $\alpha \leq 1$ , as all the elements are stored in the table.