10.2-6.

The dynamic-set operation UNION takes two disjoint set S_1 and S_2 as input, and it returns a set $S = S_1 \cup S_2$ consisting of all the elements of S_1 and S_2. The sets S_1 and S_2 are usually destroyed by the operation. Show how to support UNION in $O(1)$ time using a suitable list data structure.

Answer.

The simplest way to join two sets is to combine two lists of their elements. As we know, combining two lists requires to connect one list's tail to the other one's head, and retrieving the tail element is crutial to this process. So how to get the tail of either one as fast as possible? Well, recall that in a circular doubly linked list, one can always reach the tail element in constant time. We thus choose to implement the UNION operation on the circular doubly linked list.

The procedure LIST-UNION modifies the boundary pointers of two lists L_1 and L_2 to produce their combination.

```
LIST-UNION($L_1, L_2$)
    $L_2.nil.next.pre = L_1.nil.pre$
    $L_1.nil.pre.next = L_2.nil.next$
    $L_2.nil.pre.next = L_1.nil$
    $L_1.nil.pre = L_2.nil.pre$
```

Figure 1 shows the effects of LIST-UNION on two sets S_1: $\{25, 9, 16\}$ and S_2: $\{4, 1\}$ representing as circular, doubly linked lists with sentinels. The running time for LIST-UNION on two lists of length m and n elements is $O(1)$.

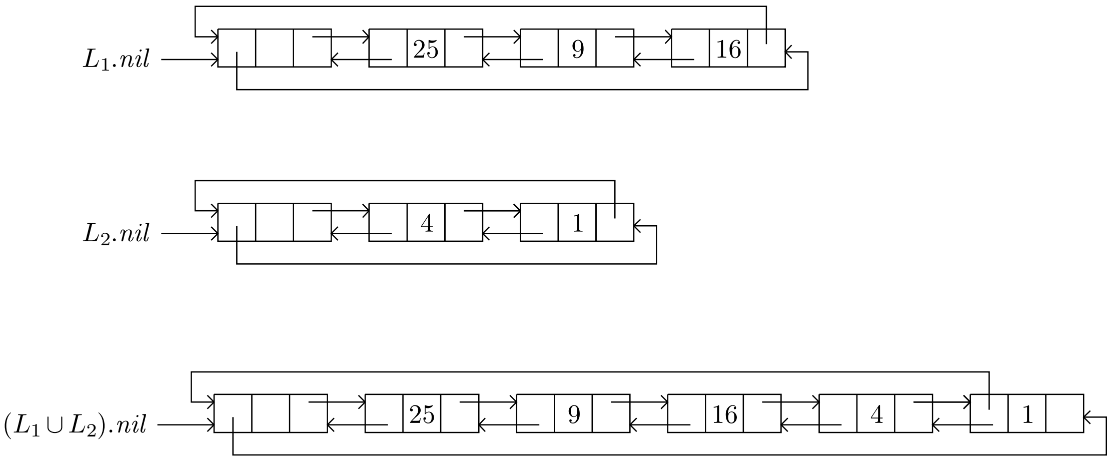

Figure 1. The result of LIST-UNION on two sets sets S_1: {25, 9, 16} and S_2: {4, 1} representing as circular, doubly linked lists with sentinels.

Email address: informlarry@gmail.com