problem_6_union_and_intersection

March 11, 2020

0.0.1 Analyze:

I need to find a way to loop the linked List and check the element in another linked list. but it will take O(n^m) so I define a set to check already fund element.

For "Union" operation. I need a new Linked list to union two LinkedList. becuase, don't change the original LinkedList.

- The function **intersection()** '2 while loop' takes less then O(n^m) time complexity.
- The function **union()** 'while loop' takes O(n)+O(m) time complexity.
- I think **intersection()** pointers take O(1) space complexity.
- I think **union()** takes O(n) because I don't want to change the original LinkedList. so create a new Linked list to record the union of llist_1 and llist_2.

[20]:
```
class Node:
    def __init__(self, value):
        self.value = value
        self.next = None

    def __repr__(self):
        return str(self.value)

    def __eq__(self, other):
        if self and other:
            return self.value == other.value
```

[21]:
```
class LinkedList:
    def __init__(self):
        self.head = None

    def append(self, value):

        if self.head is None:
            self.head = Node(value)
            return

        node = self.head
        while node.next:
```

```
            node = node.next

        node.next = Node(value)

    def size(self):
        size = 0
        node = self.head
        while node:
            size += 1
            node = node.next

        return size

    def __str__(self):
        cur_head = self.head
        out_string = ""
        while cur_head:
            if cur_head.next:
                arrows = " -> "
            else:
                arrows = ""
            out_string += str(cur_head.value) + arrows
            cur_head = cur_head.next
        return out_string
```

[22]:
```
def union(llist_1, llist_2):

    union_llist = LinkedList()

    p1 = llist_1.head
    p2 = llist_2.head

    while p1:
        union_llist.append(p1.value)
        p1 = p1.next

    while p2:
        union_llist.append(p2.value)
        p2 = p2.next

    return union_llist
```

[24]:
```
def intersection(llist_1, llist_2):

    p1 = llist_1.head
    llist_set = set()
```

```
    while p1:
        if not p1.value in llist_set:
            p2 = llist_2.head
            while p2:
                if p1 == p2:
                    llist_set.add(p1.value)
                p2 = p2.next
        p1 = p1.next

    inter_llist = LinkedList()
    for i in llist_set:
        inter_llist.append(i)

    return inter_llist
```

[25]:
```
# Test case 1

llist_1 = LinkedList()
llist_2 = LinkedList()

element_1 = [4,1,8,4,5]
element_2 = [5,0,1,8,4,5]

for i in element_1:
    llist_1.append(i)

for i in element_2:
    llist_2.append(i)

print('------Linked Lists------')
print(element_1)
print(element_2)

result1 = union(llist_1,llist_2)
print('------Union result------')
print(result1)

result2 = intersection(llist_1,llist_2)
print('------Intersection result------')
print(result2)
```

```
------Linked Lists------
[4, 1, 8, 4, 5]
[5, 0, 1, 8, 4, 5]
------Union result------
4 -> 1 -> 8 -> 4 -> 5 -> 5 -> 0 -> 1 -> 8 -> 4 -> 5
------Intersection result------
```

```
8 -> 1 -> 4 -> 5
```

[26]:
```
# Test case 2

linked_list_1 = LinkedList()
linked_list_2 = LinkedList()

element_1 = [3,2,4,35,6,65,6,4,3,21]
element_2 = [6,32,4,9,6,1,11,21,1]

for i in element_1:
    linked_list_1.append(i)

for i in element_2:
    linked_list_2.append(i)


print('------Linked Lists------')
print(element_1)
print(element_2)

result1 = union(linked_list_1,linked_list_2)
print('------Union result------')
print(result1)

result2 = intersection(linked_list_1,linked_list_2)
print('------Intersection result------')
print(result2)
```

```
------Linked Lists------
[3, 2, 4, 35, 6, 65, 6, 4, 3, 21]
[6, 32, 4, 9, 6, 1, 11, 21, 1]
------Union result------
3 -> 2 -> 4 -> 35 -> 6 -> 65 -> 6 -> 4 -> 3 -> 21 -> 6 -> 32 -> 4 -> 9 -> 6 -> 1
-> 11 -> 21 -> 1
------Intersection result------
4 -> 21 -> 6
```

[27]:
```
# Test case 3

linked_list_3 = LinkedList()
linked_list_4 = LinkedList()

element_3 = [3,2,4,35,6,65,6,4,3,23]
element_4 = [1,7,8,9,11,21,1]

for i in element_3:
```

```
    linked_list_3.append(i)

for i in element_4:
    linked_list_4.append(i)

print('------Linked Lists------')
print(element_3)
print(element_4)

result1 = union(linked_list_3,linked_list_4)
print('------Union result------')
print(result1)

result2 = intersection(linked_list_3,linked_list_4)
print('------Intersection result------')
print(result2)
```

```
------Linked Lists------
[3, 2, 4, 35, 6, 65, 6, 4, 3, 23]
[1, 7, 8, 9, 11, 21, 1]
------Union result------
3 -> 2 -> 4 -> 35 -> 6 -> 65 -> 6 -> 4 -> 3 -> 23 -> 1 -> 7 -> 8 -> 9 -> 11 ->
21 -> 1
------Intersection result------
```

[ ]: