作业反馈3-3

TC第21.1节练习2、3

TC第21.2节练习1、3、6

TC第21.3节练习1、2、3

TC第21章问题1

21.2-1

Write pseudocode for MAKE-SET, FIND-SET, and UNION using the linked-list representation and the weighted-union heuristic. Make sure to specify the attributes that you assume for set objects and list objects.

缺点什么？

```
MAKE-SET(x)
    Let Sx be a new set
    Sx.head = x
    Sx.tail = x
    x.next = NULL
    x.set = Sx
    Sx.weight = 1

FIND-SET(x)
    return x.set
```

```
UNION(x,y)
    Let Sx = x.set, Sy = y.set
    if(Sx.weight < Sy.weight)
        UNION(y,x)
    else
        Let tail = Sx.tail, head = Sy.head
        Let x = head
        tail.next = head
        while(x != NULL)
            x.set = Sx
            x = x.next
        Sx.tail = tail
        Sy.delete()
```

Update weight!

21.2-6

Suggest a simple change to the UNION procedure for the linked-list representation that removes the need to keep the *tail* pointer to the last object in each list. Whether or not the weighted-union heuristic is used, your change should not change the asymptotic running time of the UNION procedure. (*Hint:* Rather than appending one list to another, splice them together.)

- 将较小的链表接在较大链表的head 后面,
- 然后遍历较小的链表, 更改每个节点的元素的代表元, 并把最后一个节点的next 指向较大链表的第二个元素.

21.3-1

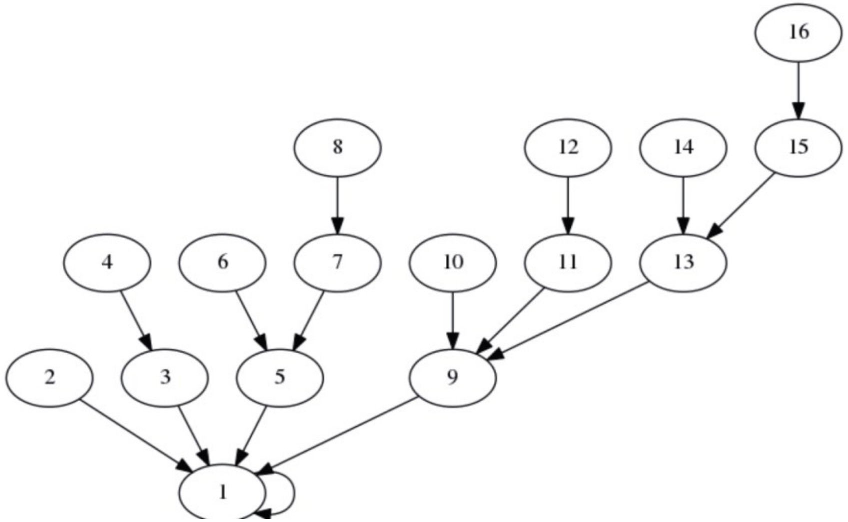
Redo Exercise 21.2-2 using a disjoint-set forest with union by rank and path compression.

```
for i = 1 to 16
    MAKE-SET(x_i)
for i = 1 to 15 by 2
    UNION(x_i, x_{i+1})
for i = 1 to 13 by 4
    UNION(x_i, x_{i+2})
UNION(x_1, x_5)
UNION(x_11, x_13)
UNION(x_1, x_10)
FIND-SET(x_2)
FIND-SET(x_9)
```

```
MAKE-SET(x)
    x.p = x
    x.rank = 0

UNION(x, y)
    LINK(FIND-SET(x), FIND-SET(y))
```

```
FIND-SET(x)
    if x ≠ x.p
        x.p = FIND-SET(x.p)
    return x.p
```



21.3-2

Write a nonrecursive version of FIND-SET with path compression.

- 需要记录什么？
  - 哪些节点的parent需要更新？
- 用什么记录？
  - 任意动态集合结构

```
Let v be a dynamic array
while(x.parent !=x)
        v.push_back(x)
        x = x.parent
for i = 1 to v.size
        v[i].p = x
```

21-1 *Off-line minimum*

The ***off-line minimum problem*** asks us to maintain a dynamic set T of elements from the domain $\{1, 2, \ldots, n\}$ under the operations INSERT and EXTRACT-MIN. We are given a sequence S of n INSERT and m EXTRACT-MIN calls, where each key in $\{1, 2, \ldots, n\}$ is inserted exactly once. We wish to determine which key is returned by each EXTRACT-MIN call. Specifically, we wish to fill in an array $extracted[1 \mathinner{..} m]$, where for $i = 1, 2, \ldots, m$, $extracted[i]$ is the key returned by the ith EXTRACT-MIN call. The problem is "off-line" in the sense that we are allowed to process the entire sequence S before determining any of the returned keys.

To develop an algorithm for this problem, we break the sequence S into homogeneous subsequences. That is, we represent S by

$$\mathrm{I}_1, \mathrm{E}, \mathrm{I}_2, \mathrm{E}, \mathrm{I}_3, \ldots, \mathrm{I}_m, \mathrm{E}, \mathrm{I}_{m+1},$$

where each E represents a single EXTRACT-MIN call and each I_j represents a (possibly empty) sequence of INSERT calls. For each subsequence I_j, we initially place the keys inserted by these operations into a set K_j, which is empty if I_j is empty.

b. Argue that the array *extracted* returned by OFF-LINE-MINIMUM is correct.

```
OFF-LINE-MINIMUM(m, n)
①
for i = 1 to n
    determine j such that i ∈ K_j
    if j ≠ m + 1
        extracted[j] = i
        let l be the smallest value greater than j
            for which set K_l exists
        K_l = K_j ∪ K_l, destroying K_j
②
return extracted
```

到底要证明什么？
循环不变式？

1. 每次循环开始前，1~i-1已被放入正确的提取位置

2. 每次循环结束后1~i已被放入正确的提取位置

T_j: 执行第j次E时的动态集合T

$c_j = \min(s \mid s \in T_j)$

$p_j = i;$

目标：$c_j = p_j;$

$I_1, E, I_2, E, I_3, \ldots, I_m, E, I_{m+1}$, **E, E, E … , $\mathbf{E_n}$**

T_j: 执行第j次E时的动态集合T

$c_j = \min(s | s \in T_j)$

$p_j = i;$

对于任意j(1<=j<=n)令:

$M_j = K_1 \cup K_2 \cup \cdots \cup K_j$

$X_j = \{s|s\text{在第j次E之前已经被抽取}\}$

$N_j = M_j - X_j$

$Q_j = \{s|s\text{在第j次E之前已经被抽取, 且}s < p_j\}$

$L_j = M_j - Q_j$

易证：$L_j \supseteq N_j = T_j$

显然，$p_j = \min\{s|s \in L_j\}$

所以，$p_j \leq \min(s|s \in N_j\} = c_j$

$$
\begin{array}{ccccccc}
c_1, & c_2, & \ldots & c_j, & \ldots, & c_n \\
\text{VI} & \text{VI} & & \text{VI} & & \text{VI} \\
p_1, & p_2, & \ldots & p_j, & \ldots, & p_n
\end{array}
$$

由于对所有j均成立，

又因为$C = \{c_j|j = 1\sim n\} = \{p_j|j = 1\sim n\} = P$

易证（反证）：

$$p_j = c_j$$