

HW 5

肖桐 PB18000037

2020 年 11 月 17 日

解 1.

```
1 S *MAKESET(int x)
2 {
3     pSet = (S *)malloc(sizeof(S)); //pSet 指向一个新的集合头节点
4     pNew = (Node *)malloc(sizeof(Node)); //pNew 指向一个新的数据节点
5     pNew->Value = x;
6     pNew->pNext = NULL;
7     pNew->Set = pSet; //指向头节点
8     pSet->head = pNew;
9     pSet->tail = pNew;
10    pSet->Length = 1;
11    return pSet;
12 }
13
14 S *FIND_SET(Node *x)
15 {
16     return x->Set;
17 }
18
19 S *UNION(Node *p1, Node *p2)
20 {
21     S *pSet1 = FIND_SET(p1);
22     S *pSet2 = FIND_SET(p2);
23     if (pSet2->Length > pSet1->Length)
24     {
25         Node *z = pSet1->head;
26         while (z != NULL)
27         {
28             z->Set = pSet2;
29             z = z->pNext;
30         }
31         pSet1->tail->pNext = pSet2->head;
32         pSet2->head = pSet1->head;
33         pSet2->Length += pSet1->Length;
```

```

34     free(pSet1);
35     return pSet2;
36 }
37 else
38 {
39     Node *z = pSet2->head;
40     while (z != NULL)
41     {
42         z->Set = pSet1;
43         z = z->pNext;
44     }
45     pSet2->tail->pNext = pSet1->head;
46     pSet1->head = pSet2->head;
47     pSet1->Length += pSet2->Length;
48     free(pSet2);
49     return pSet1;
50 }
51 }

```

解 2. 设 $Value[i, j]$ 为选择前 i 个物品中的其中几个加入背包中, 且使得总重量不超过 j 所取得的最大价值.

初始化 $Value[n+1, W+1]$

设 n 件物品重量为 w_1, w_2, \dots, w_n , 对应的价值分别为 v_1, v_2, \dots, v_n .

则有最优子结构: $Value[i, j] = \max\{Value[i-1, j], Value[i-1, j-w_i] + v_i\}$.

理解为每次尝试将最新纳入考虑的第 i 件物品装入背包, 若将第 i 件物品装入背包能够使得总价值增加的话, 那便将第 i 件物品装入背包, 否则保持原来的状态.

因此采取自底向上的方法可以写出伪代码:

```

1  for (int i = 0; i < n + 1; i++)
2  { // 初始化
3      Value[i, 0] = 0;
4  }
5  for (int j = 0; j < W + 1; j++)
6  { // 初始化
7      Value[0, j] = 0;
8  }
9
10 for (int i = 1; i < n + 1; i++)
11 {
12     for (int j = 1; j < W + 1; j++)
13     {
14         Value[i, j] = max{Value[i - 1, j], Value[i - 1, j - w_i] + v_i};
15     }
16 }

```

因为内循环复杂度为 $O(nW)$, 因此算法总复杂度为 $O(nW)$.