EECS 340: Assignment 4

Shaochen (Henry) ZHONG, sxz517
Yuhui ZHANG, yxz2052

Due and submitted on 03/19/2020
EECS 340, Dr. Koyutürk

Problem 1

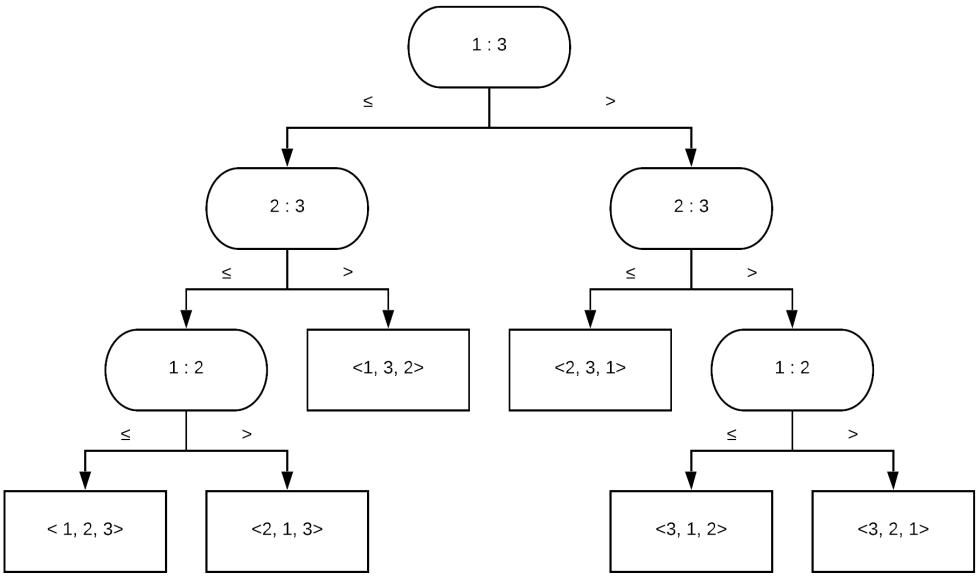

Figure 1: Decision Tree of QUICKSORT

Problem 2

Procedure PREPROCESS(A)

Algorithm 1 PreProcess(A)

```
1: procedure PREPROCESS(A, p, r)
2:     Let C[0, 1, ..., k] be an arry of 0.
3:     for i ← 0 to n do
4:         C[A[i]] = C[A[i]] + 1
5:     for j ← 1 to k do
6:         C[j] = C[j] + C[j − 1]
7:     return C
```

Procedure QUERY(A, a, b)

Algorithm 2 Query(A, a, b)

```
1: procedure QUERY(A, a, b)
2:     if a == 0 then
3:         return A[b]
4:     else
5:         return A[b] − A[a − 1]
```

Problem 3

Algorithm 3 Sparse-Transpose(R, C, V, m, n, k)

```
1: procedure SPARSE-TRANSPOSE(A, a, b)
2:     Let CH[0, 1, ..., n] be an arry of 0.
3:     Let VH[0, 1, ..., n] be an arry of 0.
4:     for i ← 0 to n do
5:         start ← R[i]
6:         end ← R[i + 1] − 1
7:         Let CPR[ ] be an empty array.
8:         Let VPR[ ] be an empty array.
9:         for j ← start to end do
10:            CPR[i].append(C[j])
11:            VPR[i].append(V[j])
12:        CH[i].append(CPR)
13:        VH[i].append(VPR)
14:    Let R[ ] be an empty array.
15:    Let C[ ] be an empty array.
16:    Let V[ ] be an empty array.
17:    V.append(0)
18:    Let len_max to be the maximum length amoung all elements in CH.
19:    for i ← 0 to len_max do
20:        POP-SMALLEST(R, C, V, CH, VH, len_max)
21:    for i ← 1 to n do
22:        R[i] = R[i] + R[i − 1]
23:    return R, C, V
```

Algorithm 4 Pop-Smallest(R, C, V, CH, VH, len_max)

```
1: procedure POP-SMALLEST(R, C, V, CH, VH, len_max)
2:     min_value ← CH[0][0]
3:     min_i ← 0
4:     min_j ← 0
5:     for i ← 0 to len_max do
6:         R_counter ← 0
7:         for j ← 0 to n do
8:             if CH[j][i] < min_value or (CH[j][i] == min_value and j < min_j) then
9:                 min_value ← CH[j][i]
10:                min_i ← i
11:                min_j ← j
12:                continue
13:            if CH[j][i] == min_value then
14:                R_counter ← R_counter + 1
15:                if j < min_j then
16:                    min_value ← CH[j][i]
17:                    min_i ← i
18:                    min_j ← j
19:        CH[min_j].remove(min_i)
20:        C.append(min_j)
21:        V.append(VH[min_j][min_i])
22:        VH[min_j].remove(min_i)
23:        R.append(R_counter)
```