Homework4

软件13 杨楠

1

a.

用数组表示d叉堆，元素按顺序存储。树的根节点是 $A[1]$ ，给定一个节点的下标i，其父节点为

$$PARENT(i) = A[(i+d-2)/d]$$

第k个子节点为

$$CHILD(i,k) = A[d(i-1)+k+i]$$

b.

高度为h的d叉堆，其元素个数n的最大值为

$$1+d+d^2+\ldots+d^h = \frac{d^{h+1}-1}{d-1}$$

最小值为

$$1+d+d^2+\ldots+d^{h-1}+1 = \frac{d^h-1}{d-1}+1$$

从而有

$$\frac{d^h-1}{d-1}+1 \le n \le \frac{d^{h+1}-1}{d-1}$$

即

$$d^h \le d^h+d-2 \le n(d-1) \le d^{h+1}-1 \le d^{h+1}$$

取对数，有

$$h \le \log_d(n(d-1)) \le h+1$$

所以包含n个元素的d叉堆的高度为

$$h = \lfloor \log_d(n(d-1)) \rfloor$$

c.

先给出d叉堆的MAX-HEAPIFY维护最大堆性质的函数的伪代码。

```
MAX-HEAPIFY(A, i)
    largest = i
    for k = 1 to d
        child = CHILD(i, k)
        if child <= A.heap-size and A[child] > A[largest]
            largest = child
        else
            break
    if largest != i
        exchange A[i] with A[largest]
        MAX-HEAPIFY(A, d, largest)
```

再给出EXTRACT-MAX函数的伪代码。

```
EXTRACT-MAX(A)
    if A.heap-size < 1
        error "heap underflow"
    max = A[1]
    A[1] = A[A.heap-size]
    A.heap-size = A.heap-size - 1
    MAX-HEAPIFY(A, 1)
    return max
```

对于MAX-HEAPIFY函数，递归次数最多为d叉堆的高度h，每次调用中，进行了d次比较，其时间复杂度为 $\mathrm{O}(dh) = \mathrm{O}(d\lfloor \log_d(n(d-1)) \rfloor) = \mathrm{O}(d \log_d n)$

从而，对于EXTRACT-MAX函数，时间复杂度由MAX-HEAPIFY函数决定，也为 $\mathrm{O}(d \log_d n)$

d.

伪代码如下。

```
INSERT(A, k)
    A.heap-size = A.heap-size + 1
    A[A.heap-size] = -∞
    INCREASE-KEY(A, A.heap-size, k)
```

时间复杂度由INCREASE-KEY函数决定，在第e问中会解释，其时间复杂度为 $\mathrm{O}(log_d n)$ ，从而时间复杂度为 $\mathrm{O}(log_d n)$

e.

伪代码如下。

```
INCREASE-KEY(A, i, k)
    if k < A[i]
        error "new key is smaller than current key"
    A[i] = k
    while i > 1 and A[PARENT(i)] < A[i]
        exchange A[i] with A[PARENT(i)]
        i = PARENT(i)
```

函数中，while循环次数最大为d叉堆的高度h，从而时间复杂度为 $\mathrm{O}(h) = \mathrm{O}(log_d n)$

2

a.

在此条件下，随机化快速排序的每次递归的划分都是最不平衡的，有 $T(n) = T(n-1) + \Theta(n)$ ，从而 $T(n) = \Theta(n^2)$

b.

修改后的PARTITION'代码如下。

```
PARTITION'(A, p, r)
    x = A[r]
    i = p - 1
    j = p - 1
    for k = p to r-1
        if A[k] < x
            i = i + 1
            j = j + 1
            exchange A[i] with A[k]
            if i != j
                exchange A[j] with A[k]
        else if A[k] == x
            j = j + 1
            exchange A[j] with A[k]
    return i + 1 and j + 1
```

c.

修改后的RANDOMIZED-PARTITION'代码如下。

```
RANDOMIZED-PARTITION'(A, p, r)
    i = RANDOM(p, r)
    exchange A[r] with A[i]
    return PARTITION'(A, p, r)
```

修改后的QUICKSORT'代码如下。

```
QUICKSORT'(A, p, r)
    if p < r
        q, s = RANDOMIZED-PARTITION'(A, p, r)
        QUICKSORT'(A, p, q - 1)
        QUICKSORT'(A, s + 1, r)
```

d.

不假设所有元素都是互异。将集合 Z_{ij} 中的 z_i 定义为A中第i小的元素中的一个。后面的分析照常。