

## 演算法

- (1) (20%) Is  $2^{n+1} = O(2^n)$ ? Is  $2^{2n} = O(2^n)$ ?
- (2) (20%) Analyze the performance of quicksort in worst-case, best-case and average case.
- (3) (20%) Suppose we use a hash function  $h$  to hash  $n$  distinct keys into an array  $T$  of length  $m$ . Assuming simple uniform hashing, what is the expected number of collisions? More precisely, what is the expected cardinality of  $\{\{k, l\}: k \neq l \text{ and } h(k) = h(l)\}$ ?
- (4) (20%) Write a recursive algorithm with dynamic programming technique to solve the knapsack problem. (Knapsack problem: A thief robbing a sage finds it filled with  $N$  types of items of varying size and value, but has only a small knapsack of capacity  $M$  to use to carry the goods. The knapsack problem is to find the combination of items that the thief should choose for the knapsack in order to maximize the total value of all the stolen items. )
- (5) (20%) A depth-first search algorithm is described as follows. Modify this algorithm to show that a depth-first search of an undirected graph  $G$  can be used to identify the connected components of  $G$ , and that the depth-first forest contains as many trees as  $G$  has connected components. More precisely, show how to modify depth-first search so that each vertex  $v$  is assigned an integer label  $cc[v]$  between 1 and  $k$ , where  $k$  is the number of connected components of  $G$ , such that  $cc[u] = cc[v]$  if and only if  $u$  and  $v$  are in the same connected component.

**Input:**  $G = (V, E)$ , directed or undirected. No source vertex given!

**Output:** 2 *timestamps* on each vertex:

- $d[v] = \text{discovery time}$
- $f[v] = \text{finishing time}$

**Pseudocode:** Uses a global timestamp *time*.

```
DFS(V, E)
for each  $u \in V$ 
  do  $color[u] \leftarrow \text{WHITE}$ 
 $time \leftarrow 0$ 
for each  $u \in V$ 
  do if  $color[u] = \text{WHITE}$ 
    then DFS-VISIT( $u$ )

DFS-VISIT( $u$ )
 $color[u] \leftarrow \text{GRAY}$     ▷ discover  $u$ 
 $time \leftarrow time + 1$ 
 $d[u] \leftarrow time$ 
for each  $v \in Adj[u]$     ▷ explore ( $u, v$ )
  do if  $color[v] = \text{WHITE}$ 
    then DFS-VISIT( $v$ )
 $color[u] \leftarrow \text{BLACK}$ 
 $time \leftarrow time + 1$ 
 $f[u] \leftarrow time$     ▷ finish  $u$ 
```