精選例題 19

(1) 若經常需要找出某一頂點的所有相鄰頂點，採用那一種圖形表示法較適合。

(2) 若經常需要兩個頂點是否相鄰，採用那一種圖形表示法較適合。

94暨南資工系－資料結構與演算法

解答：(1) Adjacency List

(2) Adjacency Matrix

8-3 圖形搜尋法 (Graph Searching Methods)

✧ 要點：深度優先追蹤 DFS(Depth-First-Search)

1. 由最後追蹤的頂點繼續追蹤未追蹤過的相鄰頂點。
2. 如果相鄰頂點都已經追蹤過了，則回溯至前一頂點繼續追蹤。
3. DFS 遞迴演算法

```
DFS(v)
  visited[v]←true;
  for each <v,w> do
      if not visited[w] then DFS(w);
```

DFS 遞迴演算法時間複雜度

(1) 圖形使用鄰接串列表示時爲 O(n+e) or O(e)。

(2) 圖形使用鄰接矩陣表示時爲 $O(n^2)$。

4. DFS 非遞迴演算法

```
DFS(v)
  push(v);
  while stack not empty do
  begin
      u←pop();
      if not visited[u] then
      begin
          visited[u]←true;
```

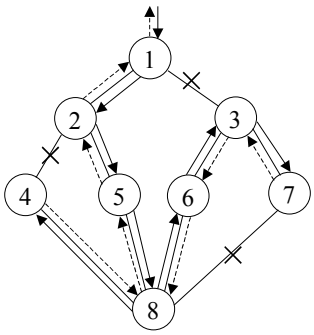
```
            for each <u,w> do
                if not visited[w] then push(w);
        end
    end;
```

DFS 非遞迴演算法時間複雜度

(1) 圖形使用鄰接串列表示時爲 O(n+e) or O(e)。

(2) 圖形使用鄰接矩陣表示時爲 $O(n^2)$。

➢ **範例：**如圖 G2 的 DFS 追蹤，以頂點 1 爲起點，其中一種追蹤次序爲 1,2,5,8,6,3,7,4(非唯一)。



✧ 要點：廣度優先追蹤 BFS(Breadth-First Search)

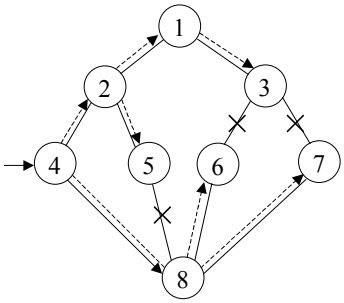
1. 自起點開始，由近而遠追蹤圖形所有頂點。
2. 若 vi 比 vj 先被追蹤到，則 vi 的直接後繼頂點就必須比 vj 的直接後繼頂點先追蹤。
3. BFS 演算法

```
visited[v]←true;
init_queue(q);
add_queue(q,v);
while q not empty do
begin
    v←del_queue(q);
    for each <v,w> do
```

```
                    if not visited[w] then
                    begin
                        add_queue(q,w);
                        visited[w]←true
                    end
                end
```

4. BFS 演算法時間複雜度
   (1) 圖形使用鄰接串列表示時爲 O(n+e) or O(e)。
   (2) 圖形使用鄰接矩陣表示時爲 $O(n^2)$。

➢ **範例：**如圖 G2 的 BFS 追蹤，以頂點 4 爲起點，其中一種追蹤次序爲 4,2,8,1,5,6,7,3(非唯一)。



✧ 要點：找出圖形的每個連通單元(Connected Components)

1. 使用 DFS 或 BFS 即可用來找出連通單元。
2. 方法如下

```
for i←1 to n do
    if not visited[i] then
    begin
        DFS(i);     // 可以替換爲 BFS(i)
        output the component;
    end;
```

3. 時間複雜度

(1) 圖形使用鄰接串列表示時爲 O(n+e) or O(e)。
(2) 圖形使用鄰接矩陣表示時爲 $O(n^2)$。

✧ 要點：後序深度優先追蹤(Post-Order Depth-First-Search)

1. 每個頂點皆有三種可能的狀態：

$$visited[v]=\begin{cases} not\_visited & ,\text{頂點v尚未追蹤到} \\ being\_visited & ,\text{頂點v正在被追蹤} \\ finish & ,\text{頂點v已追蹤完成} \end{cases}$$

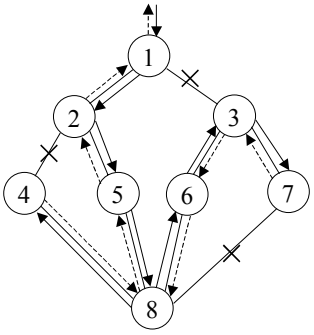
2. 演算法：

```
for each u ∈ V(G) do
    visited[u]←unvisited;
for each u ∈ V(G) do
    if visited[u]=not_visited then
        DFS(u);

DFS(u)
    visited[u]←being_visited;
    for each (u,v) ∈ E(G) do
        if visited[v]= not_visited then
            DFS(v);
    visited[u]←finish;
```

3. DFS 追蹤：依每個頂點由 not_visited 改變爲 being_visited 的次序追蹤，即爲 DFS 追蹤。
4. 後序 DFS 追蹤：依每個頂點由 being_visited 改變爲 finish 的次序追蹤，即爲後序 DFS 追蹤。
5. 時間複雜度：
   (1) 圖形使用鄰接串列表示時爲 O(n+e) or O(e)。
   (2) 圖形使用鄰接矩陣表示時爲 $O(n^2)$。

➢ **範例：**如圖 G2 的後序 DFS 追蹤，以頂點 1 爲起點，其中一種追蹤次序爲 7, 3, 6, 4, 8, 5, 2, 1（非唯一）。



精選例題 20

圖2爲一個圖型（graph）。

(1) 請利用廣度優先搜尋法（breadth-first search, BFS），從節點c開始，列出此圖型的所有節點。請將字元較小的節點優先列出。

(2) 請利用深度優先搜尋法（depth-first search, DFS），從節點c開始，列出此圖型的所有節點。請將字元較小的節點優先列出。

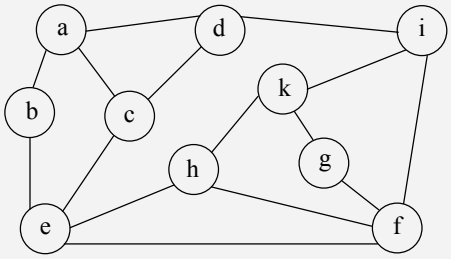


圖2

100年公務人員、關務人員薦任升等

解答：(1) c,a,d,e,b,i,f,h,k,g

(2) c,a,b,e,f,g,k,h,i,d