圖形的走訪

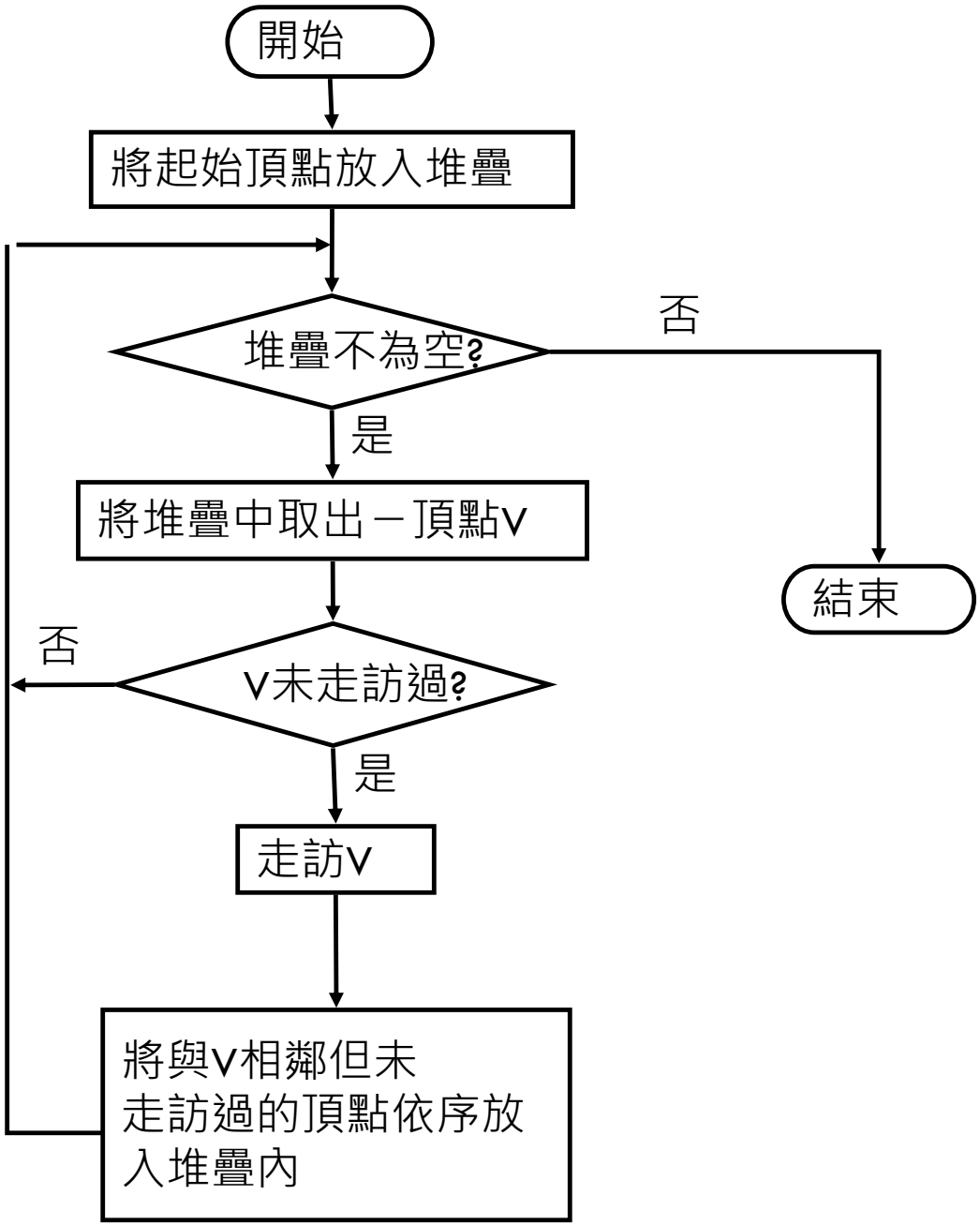
資料結構
鍾宜玲

深度優先搜尋DFS (Depth First Search)

- 任選一個起始頂點v開始走訪
- 接著走訪與v相鄰但未走訪的任一頂點，設為 v_i ，並由頂點 v_i 繼續深度優先搜尋
- 當走訪完某一頂點 v_i ，若 v_i 的所有相鄰頂點皆已走訪，則退回到 v_i 的前一個走訪頂點，繼續深度優先走訪。

流程圖

(DFS利用堆疊)


開始
將起始頂點放入堆疊
堆疊不為空?
否
是
結束
將堆疊中取出一頂點v
否
v未走訪過?
是
走訪v
將與v相鄰但未
走訪過的頂點依序放
入堆疊內

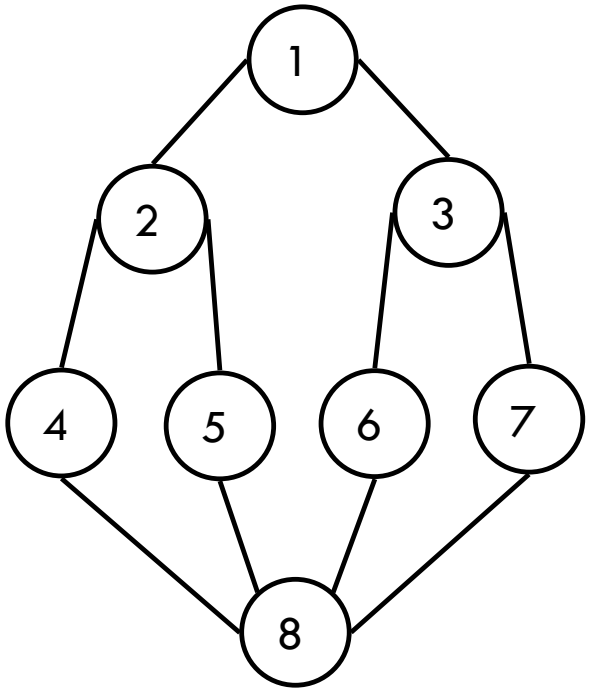
DFS：利用堆疊

S為一個空堆疊

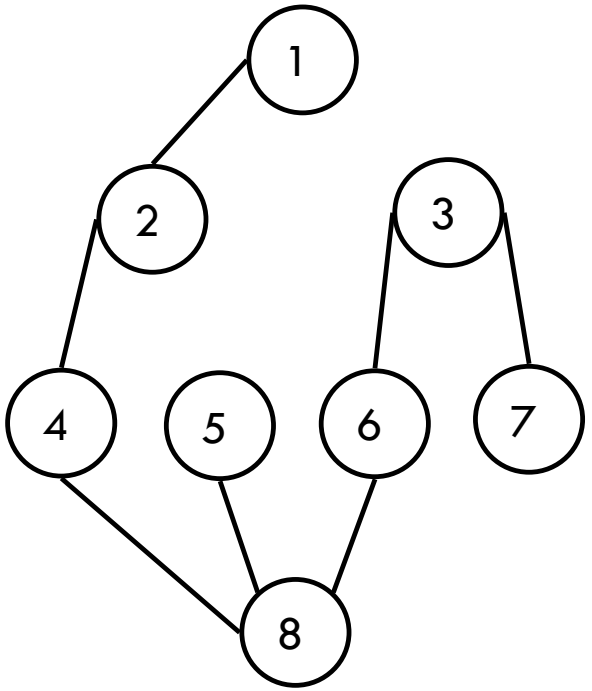
1. 將起始頂點 v 放入 (push) S 中
2. 若 S 不是空的，則執行下列步驟，否則跳到步驟 3
   - 2.1 從 S 取出(pop)一頂點 w
   - 2.2 若w未走訪過，走訪w
     否則跳回步驟 2
   - 2.3 將與 w 相鄰且尚未拜訪過的所有頂點依序放入 (push) S 中
   - 2.4 回到步驟 2
3. 結束

範例

如下圖(a) 的圖形，若由頂點1開始走訪，則走訪的順序為 1, 2, 4, 8, 5, 6, 3, 7，所產生的DFS擴展樹如圖(b)。

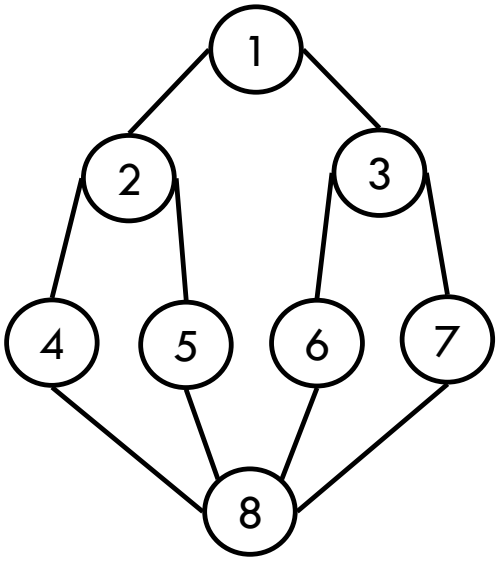


(a) 原圖形



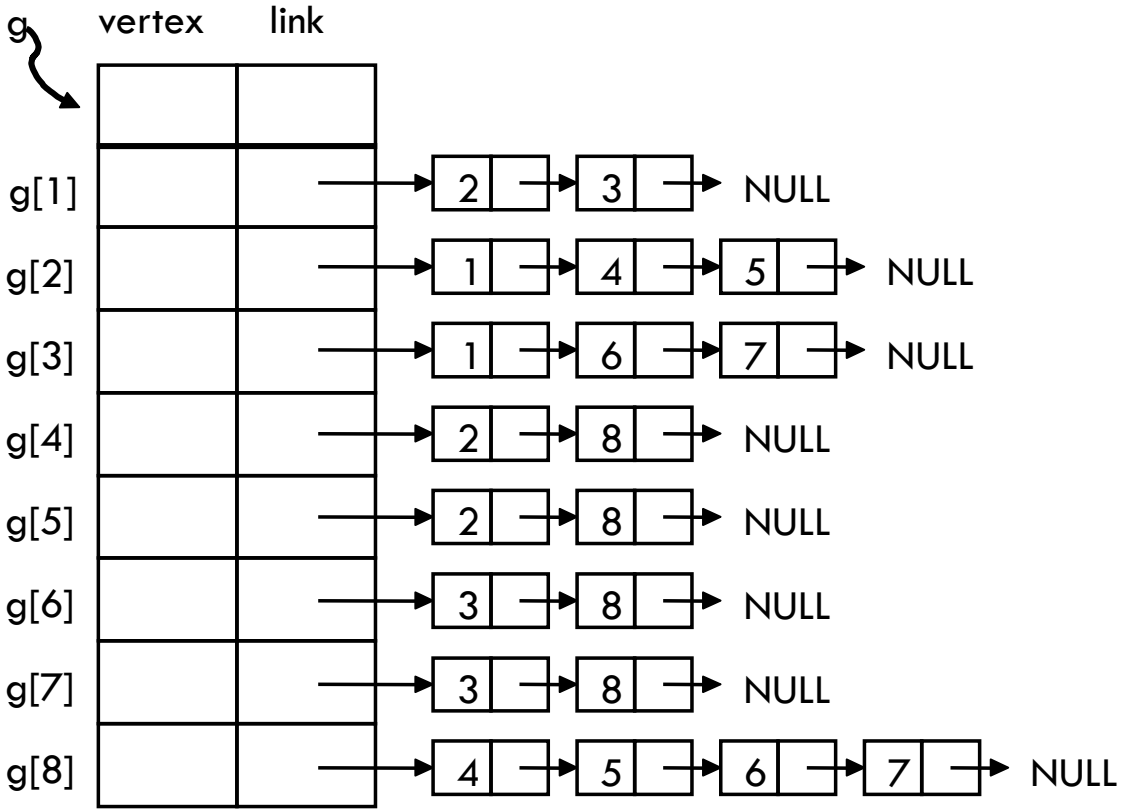
(b) DFS擴展樹

圖形的頂點結構



若圖形的頂點結構如下：

```
typedef  struct  gnode {
     int  vertex;
     struct  gnode  *link;
}  GNODE;
GNODE  *g;
```



DFS：利用堆疊

```
void dfs (int i)
{
 GNODE *p;
 int v;
 push(i);                                  //1. 將起始頂點放入堆疊內
 while (top !=NULL) {                      //2. 若堆疊不為空，則
  v=pop();                                 //   2.1從堆疊中取出頂點v
  if (g[v].vertex !=1) {                   //   2.2若頂點v未走訪過
     printf ("%3d",v);                     //         走訪頂點v
     g[v].vertex=1;                        //         走訪記錄設為1
     for(p=g[v].link;p;p=p->link)          //  2.3與v相鄰的所有頂點
       if(q[p->vertex].vertex !=1)         //  找出尚未走訪過的頂點
          push (p->vertex);                //  依序放入等待走訪堆疊中
      }
   }                                       // 2.4 回到步驟 2
} /*end of bfs()*/                         // 3. 結束
```

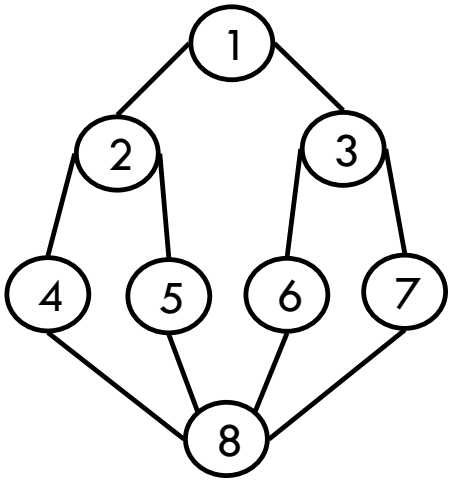
遞迴程式製作深度優先搜尋

```
/*深度優先搜尋：利用遞迴觀念*/
void dfs(int i)
{
 GNODE  *p;
 int d;

 printf("%3d", i);          /*走訪頂點i*/
 g[i].vertex = 1;           /*將走訪記錄設為1*/

 for(p=g[i].link; p !=NULL; p=p->link) /*找一個與頂點i相鄰*/
     if( g[p->vertex].vertex != 1)          /*且未走訪過的頂點*/
         dfs(p->vertex);              /*繼續深度優先搜尋*/
}
```

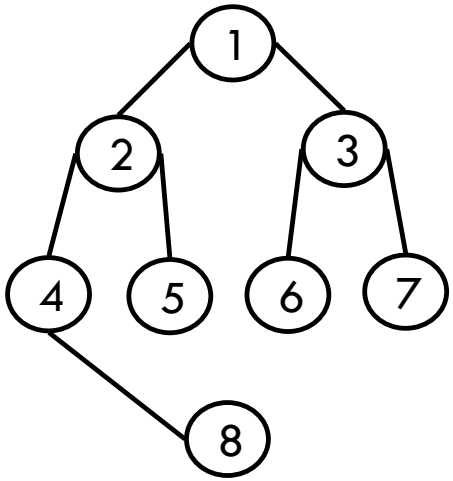
廣度優先搜尋BFS (BREADTH FIRST SEARCH)

- 任選一個起始頂點v開始走訪
- 接著一一走訪完與頂點v相鄰但未走訪的頂點。

圖形(a)若由頂點1開始走訪，走訪順序為1, 2, 3, 4, 5, 6, 7, 8



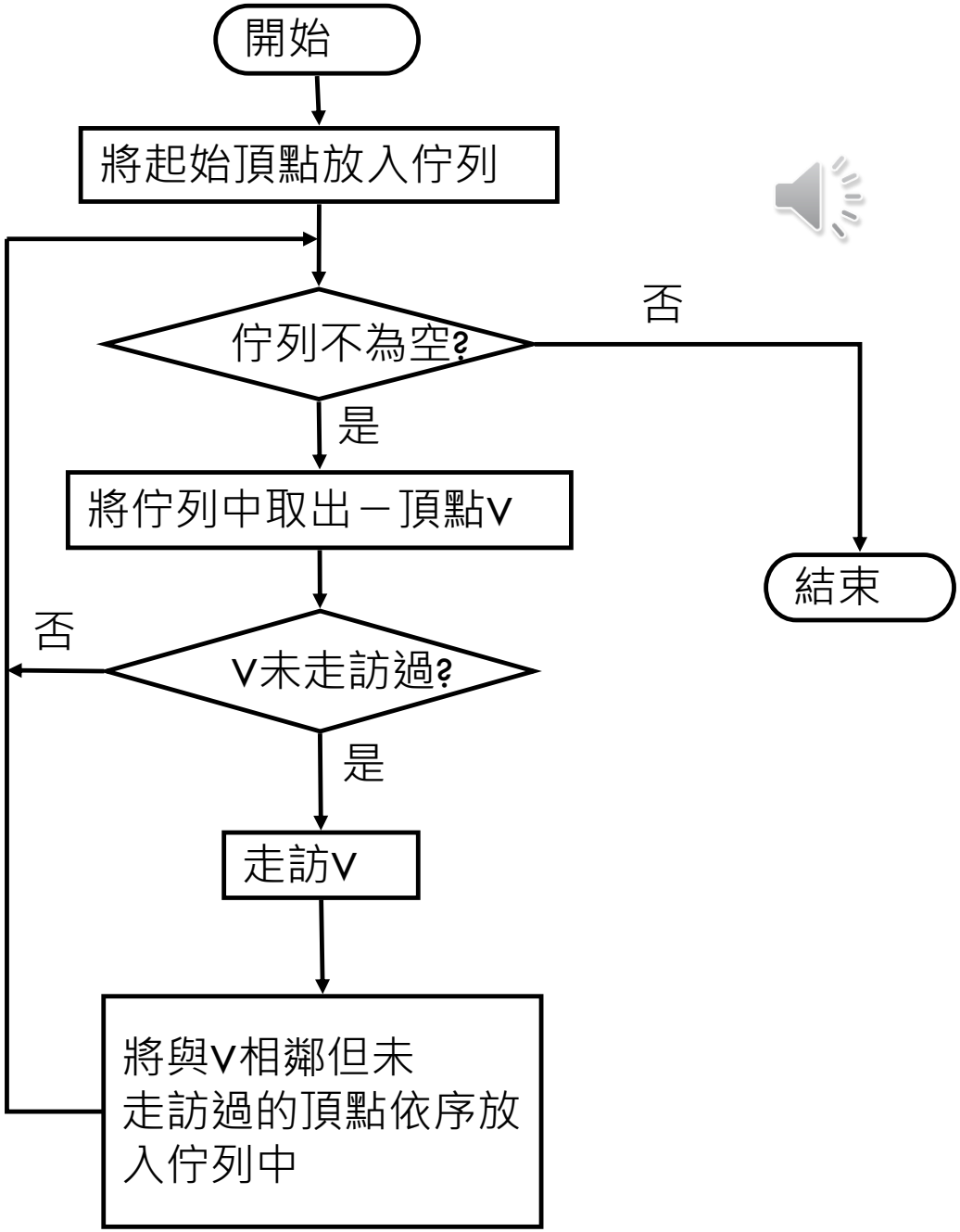
(a) 原圖形



(b) BFS擴展樹

流程圖

(BFS利用佇列)


開始
將起始頂點放入佇列
佇列不為空?
否
是
將佇列中取出一頂點V
結束
否
V未走訪過?
是
走訪V
將與V相鄰但未
走訪過的頂點依序放
入佇列中

BREADTH-FIRST SEARCH (BFS)

Q為一個空佇列

1. 將起始頂點 v 加入Q中（Enqueue）
2. 若Q不是空的，則執行下列步驟，否則跳到步驟 3
   - 2.1 從Q取出一頂點 w（Dequeue）
   - 2.2若w未走訪過，走訪w
     否則跳回步驟 2
   - 2.3 將與 w 相鄰且尚未走訪過的所有頂點依序加入Q中（Enqueue）
   - 2.4 回到步驟 2
3. 結束

廣度優先搜尋：利用佇列

假設圖形以相鄰串列表示，且頂點節點結構如下：

```
typedef  struct  gnode {
      int  vertex;
      struct  gnode  *link;
 }  GNODE;
 GNODE  *g;
```

廣度優先搜尋的函數BFS()

```
/*廣度優先搜尋：利用佇列*/
void bfs (int i)
{
 GNODE *p;
 int v;
 add_queue(i);                          /*將起始頂點放入佇列中*/
 while (front !=NULL) {                 /*若佇列不為空，則*/
    v=delete_queue();                           /*從佇列中取出頂點v*/
    if (g[v].vertex !=1) {              /*若頂點v未走訪過*/
       printf("%3d",v);                         /*走訪頂點v*/
       g[v].vertex=1;                           /*走訪記錄設為1*/
       for(p=g[v].link; p ;p=p->link)           /*與頂點v相鄰的所有頂點*/
          if(g[p->vertex].vertex !=1)           /*找出尚未走訪過的頂點*/
             add_queue(p->vertex);/*依序放入等待走訪的佇列中*/
    }
 }
} /*end of bfs()*/
```