HW6_ minimum spanning tree problem

Kruskal’ s

程式摘要:

此程式由 C 語言所寫成，核心的觀念為 minimun spanning tree,做法是每次都選擇 weight 最小的 edge,直到每個 node 都被使用過

```
1. #include <stdio.h>
2. #include<stdlib.h>
3.
4.
5. struct vertex {
6.     vertex* root;
7.     int nuber;
8. };
9.
10.  struct edge{
11.      int weight;
12.      vertex* v1;
13.      vertex* v2;
14.      bool used;
15.  };
16.
17.  int comp(const void *a, const void *b){
18.      return (*(edge *)a).weight > (*(edge *)b).weight ?
   1 : -1;
19.  }
20.
21.  int main() {
22.
23.      int n;
24.      scanf("%d", &n);
25.
26.      while (n--) {
27.          int totalWeight = 0;
28.          int v, e, option;
```

```
29.         scanf("%d %d %d", &v, &e, &option);
30.
31.         vertex *vertexs = new vertex[v];
32.         for (int i = 0; i < v; i++) {//初始化
33.             vertexs[i].nuber = i;
34.             vertexs[i].root = NULL;
35.         }
36.
37.         edge *edges = new edge[e];
38.         for (int i = 0; i < e; i++) {//讀 edge 的側資
39.             int v1, v2;
40.             scanf("%d %d", &v1, &v2);
41.             edges[i].v1 = &vertexs[v1];
42.             edges[i].v2 = &vertexs[v2];
43.             scanf("%d", &edges[i].weight);
44.             edges[i].used = false;
45.         }
46.         qsort(edges, e, sizeof(edge), comp);//依照
  weight 做排序
47.
48.         for (int i = 0; i < e; i++) {
49.             vertex *root1 = edges[i].v1;
50.             vertex * root2= edges[i].v2;
51.             while (root1->root != NULL) {  //找該 node 的
  root
52.
53.                 root1 = root1->root;
54.             }
55.             while (root2->root != NULL) {  //找該 node 的
  root
56.                 root2 = root2->root;
57.             }
58.
59.             if (root1 != root2) {
60.                 root2->root = root1;
61.                 edges[i].used = true;
62.             }
63.         }
```

```
64.
65.         for (int i = 0; i < v; i++) {//Kruskal's algori
    thm
66.             if (edges[i].used) {
67.                 if (option == 1) {
68.                     if(edges[i].v1-
    >nuber<= edges[i].v2->nuber)
69.                         printf("%d %d\n", edges[i].v1-
    >nuber, edges[i].v2->nuber);
70.                     else
71.                         printf("%d %d\n", edges[i].v2-
    >nuber, edges[i].v1->nuber);
72.                 }
73.                 totalWeight += edges[i].weight;
74.             }
75.         }
76.         printf("%d\n", totalWeight);
77.     }
78.
79.
80.     return 0;
81. }
```

Pseudo code:

Sort(edge,依照 weight)

totalWeight=0

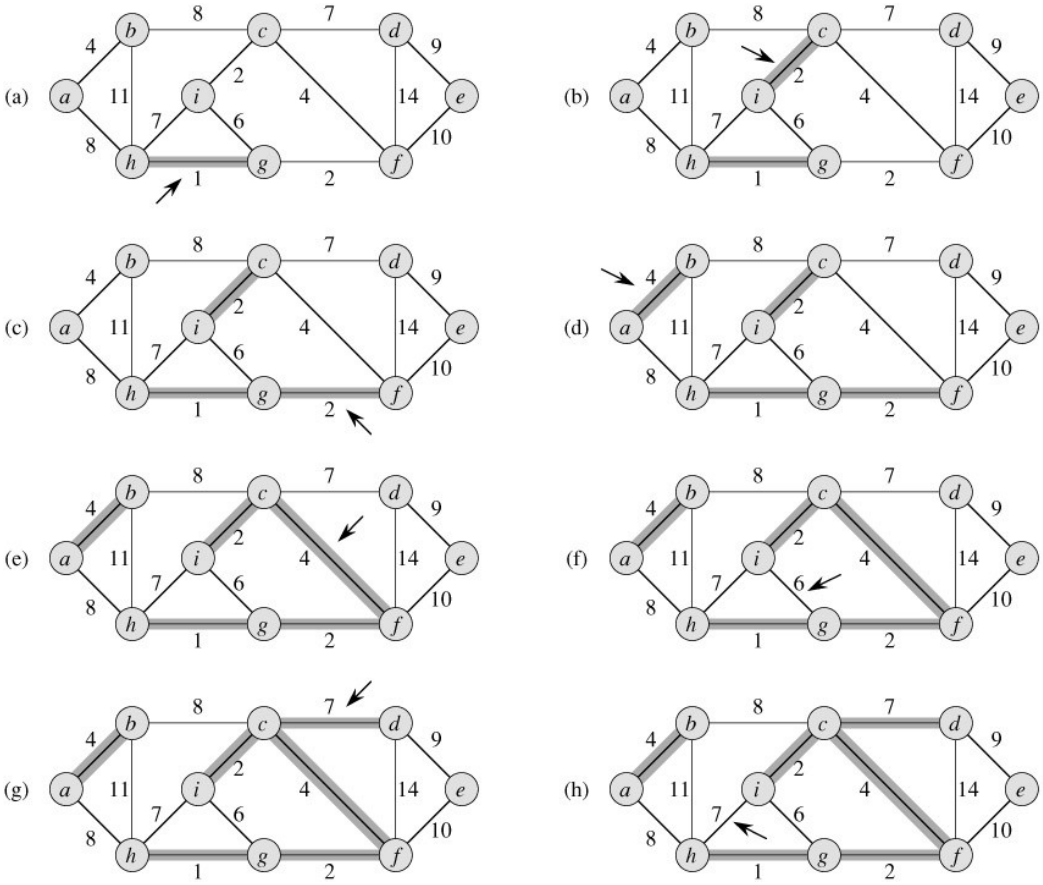
For i=0 to vertice 數量

    If(edge[i]連接的兩個 node 其中一個的 color 是 1)

        將該 node 的 color 改為 2

        totalWeight+= edge[i].weight

圖解:



如圖:

每次都選擇 weight 最小的 edge

直到每個 node 都被使用過

Prim's

程式摘要:

此程式由 C 語言所寫成，核心的觀念為 minimun spanning tree,做法是先隨機選一個 node,將該 node 鄰近的 edge 的 color 標為 1,並將該 node 的 color 標為 2;然後將 edge color 為 1 且 weight 最小的取出,不斷執行以上步驟 v-1 次

```
1. #include <stdio.h>
2. #include<stdlib.h>
3.
4.
5. struct vertex {
6.     vertex* parent;
7.     int nuber;
8.     int color;
9.
10.  };
11.
12.  struct edge {
13.      int weight;
14.      vertex* v1;
15.      vertex* v2;
16.      int color;
17.  };
18.
19.  int main() {
20.
21.      int n;
22.      scanf("%d", &n);
23.
24.      while (n--) {
25.          int totalWeight = 0;
26.          int v, e, option;
27.          scanf("%d %d %d", &v, &e, &option);
28.
29.          vertex *vertexs = new vertex[99];
30.          for (int i = 0; i < v; i++) {//初始化
```

```
31.             vertexs[i].nuber = i;
32.             vertexs[i].parent = NULL;
33.             vertexs[i].color = 0;
34.         }
35.         edge *edges = new edge[9999];
36.         for (int i = 0; i < e; i++) {//讀 edge 的側資
37.             int v1, v2;
38.             scanf("%d %d", &v1, &v2);
39.             if (v1 < v2) {
40.                 edges[i].v1 = &vertexs[v1];
41.                 edges[i].v2 = &vertexs[v2];
42.             }
43.             else {
44.                 edges[i].v2 = &vertexs[v1];
45.                 edges[i].v1 = &vertexs[v2];
46.             }
47.
48.             scanf("%d", &edges[i].weight);
49.             edges[i].color = 0;
50.         }
51.
52.         vertex *selectVertice = &vertexs[0];
53.         v--;
54.         while (v--) {
55.
56.             selectVertice->color = 2;
57.
58.             for (int i = 0; i < e; i++) {
59.                 if (edges[i].v1 == selectVertice) {
60.                     if (edges[i].v2->color == 0)
61.                         edges[i].v2->color = 1;
62.                     if (edges[i].color == 0)
63.                         edges[i].color = 1;
64.                 }
65.                 else if (edges[i].v2 == selectVertice)
   {
66.                     if (edges[i].v1->color == 0)
67.                         edges[i].v1->color = 1;
```

```
68.                     if (edges[i].color == 0)
69.                         edges[i].color = 1;
70.                 }
71.                 if (edges[i].v1-
   >color == 2 && edges[i].v2->color == 2) {
72.                     edges[i].color = 2;
73.                 }
74.             }
75.
76.             edge*selectEdge = NULL;
77.
78.             for (int i = 0; i < e; i++) {
79.                 if (selectEdge == NULL)
80.                     if (edges[i].color == 1)
81.                         selectEdge = &edges[i];
82.                     else
83.                         continue;
84.                 else if (edges[i].color == 1 && edges[i
   ].weight < selectEdge->weight) {
85.                     selectEdge = &edges[i];
86.                 }
87.             }
88.
89.             if (selectEdge != NULL) {
90.                 selectEdge->color = 2;
91.                 if (selectEdge->v1->color == 1)
92.                     selectVertice = selectEdge->v1;
93.                 else
94.                     selectVertice = selectEdge->v2;
95.
96.                 totalWeight += selectEdge->weight;
97.                 if (option == 1)
98.                     printf("%d %d\n", selectEdge->v1-
   >nuber, selectEdge->v2->nuber);
99.             }
100.            else
101.                break;
102.        }
```

```
103.
104.         printf("%d\n", totalWeight);
105.     }
106.
107.     //system("pause");
108.     return 0;
109. }
```

Pseudo code:

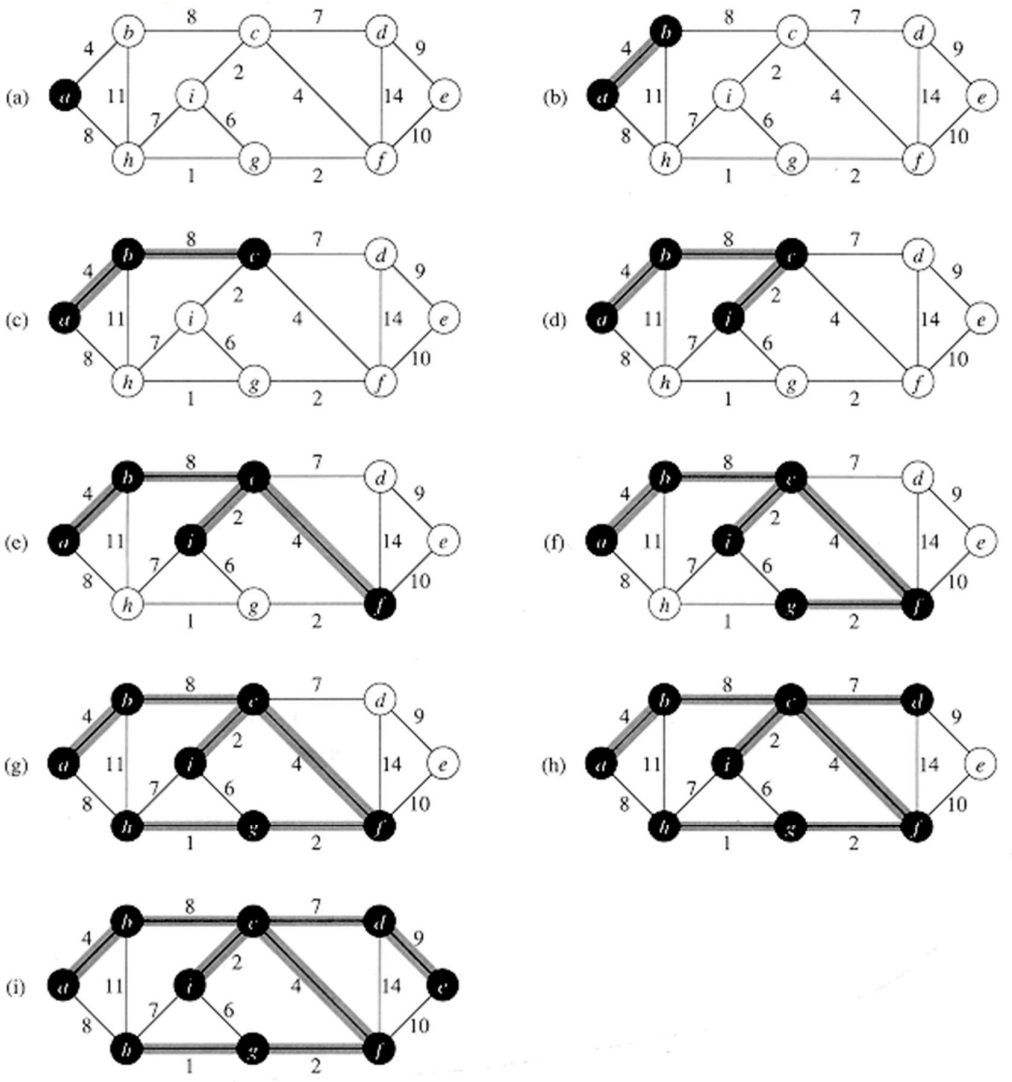
Select=number 為 0 的 node

While(node 數量—)

將 select 鄰近的 edge 的 color 標為 1

選出 edge color 為 1 且 weight 最小的 edge

圖解:



如圖

將 select 鄰近的 edge 的 color 標為 1

選出 edge color 為 1 且 weight 最小的 edge