数据结构作业13

P305的习题

文本, 信件

描述已自动生成

Kruskal算法和Prim算法求最小生成树

#include <iostream>  
#include <algorithm>  
#include <string>  
#include <vector>  
const int INF = **INT\_MAX**;  
using namespace std;  
vector<vector<int>> graph = **{** {INF, 28, INF, INF, INF, 10, INF},  
 {28, INF, 16, INF, INF, INF, 14},  
 {INF, 16, INF, 12, INF, INF, INF},  
 {INF, INF, 12, INF, 22, INF, 18},  
 {INF, INF, INF, 22, INF, 25, 24},  
 {10, INF, INF, INF, 25, INF, INF},  
 {INF, 14, INF, 18, 24, INF, INF}  
**}**; *//Adjacent Matrix*void Prim(vector<vector<int>> g){  
 int n = g.size();  
 vector<int> lowcost(n, INF);  
 vector<int> closet(n, -1);  
 vector<bool> visited(n, false);  
 lowcost[0] = 0;  
 for(int i = 0; i < n; i++){  
 int v = -1;  
 for(int j = 0; j < n; j++){  
 if(!visited[j] && (v == -1 || lowcost[j] < lowcost[v])){  
 v = j;  
 }  
 }  
 if(lowcost[v] == INF){  
 cout << "没有最小生成树" << endl;  
 return;  
 }  
 visited[v] = true;  
 for(int u = 0; u < n; u++){  
 if(g[v][u] < lowcost[u]){  
 lowcost[u] = g[v][u];  
 closet[u] = v;  
 }  
 }  
 }  
 for(int i = 0; i < n; i++){  
 cout << closet[i] << " - " << i << " : " << lowcost[i] << endl;  
 }  
}  
void Kruskal(vector<vector<int>> g){  
 int n = g.size();  
 vector<int> lowcost(n, INF);  
 vector<int> closet(n, -1);  
 vector<bool> visited(n, false);  
 lowcost[0] = 0;  
 for(int i = 0; i < n; i++){  
 int v = -1;  
 for(int j = 0; j < n; j++){  
 if(!visited[j] && (v == -1 || lowcost[j] < lowcost[v])){  
 v = j;  
 }  
 }  
 if(lowcost[v] == INF){  
 cout << "没有最小生成树" << endl;  
 return;  
 }  
 visited[v] = true;  
 for(int u = 0; u < n; u++){  
 if(g[v][u] < lowcost[u]){  
 lowcost[u] = g[v][u];  
 closet[u] = v;  
 }  
 }  
 }  
 for(int i = 0; i < n; i++){  
 cout << closet[i] << " - " << i << " : " << lowcost[i] << endl}  
  
}  
int main(){  
 Prim(graph);  
 Kruskal(graph);  
 return 0;  
}