### TSP问题

1. **问题描述：**

旅行商从驻地出发，经过每个需要访问的城市一次且只有一次，并最终返回出发点。如何安排路线，使旅行总路程最短？

1. **问题形式化定义：**

n个城市组成的带权无向图G=(V,E)，顶点V对应于城市，边E对应于城市间路径，要求找出一条旅行线路，每个城市只经历一次

1. **问题解决：**

1

3

2

4

30

6

10

20

5

4

解决如图所示的旅行商问题

思路：

回溯法排列树问题：

因为TSP是一个哈密顿回路，所以只要假定一个起点，对结果无影响

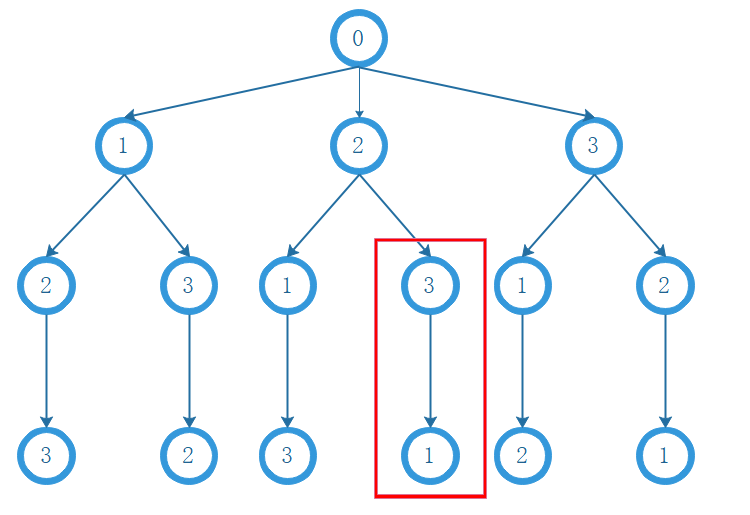
假定1为起点，维护一个flag数组，表示该顶点是否被使用过，使用递归的方法，有四个参数，分别表示当前花费，第几个节点

和从哪个顶点来以及当前路径，使用排列树的思想来解决

画出以1为起点的排列树并剪枝

剪枝条件是：

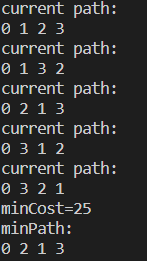
如果当前cost大于minCost，放弃搜索



代码：

1. #include <cstdio>
2. #include <iostream>
3. #define POINT\_NUM 4
4. **using** **namespace** std;
6. **int** minCost=(1<<30)-1;
7. **bool** flag[POINT\_NUM];
8. **int** graph[POINT\_NUM][POINT\_NUM]={0,30,6,4,
9. 30,0,5,10,
10. 6,5,0,20,
11. 4,10,20,0};
12. **int** minPath[POINT\_NUM];
13. **void** TSP(**int** cost,**int** x,**int** from,**int** path[]){
14. **if**(x==POINT\_NUM){
15. cost+=graph[from][0];
16. cout<<"current path:"<<endl;
17. **for**(**int** i=0;i<POINT\_NUM;i++)
18. cout<<path[i]<<" ";
19. cout<<endl;
20. **if**(minCost>cost){
21. minCost=cost;
22. **for**(**int** i=0;i<POINT\_NUM;i++)
23. minPath[i]=path[i];
24. }
25. }
26. **if**(cost>minCost)
27. **return**;
28. **for**(**int** i=0;i<POINT\_NUM;i++){
29. **if**(!flag[i]){
30. flag[i]=**true**;
31. path[x]=i;
32. TSP(cost+graph[from][i],x+1,i,path);
33. flag[i]=**false**;
34. }
35. }
36. }
37. **void** solution(){
38. **for**(**int** i=0;i<POINT\_NUM;i++)
39. flag[i]=**false**;
40. flag[0]=**true**;
41. **int** path[POINT\_NUM];
42. path[0]=0;
43. TSP(0,1,0,path);
44. cout<<"minCost="<<minCost<<endl;
45. cout<<"minPath:"<<endl;
46. **for**(**int** i=0;i<POINT\_NUM;i++)
47. cout<<minPath[i]<<" ";
49. }
50. **int** main(){
51. solution();
52. }

输出：



可见0->2->3->1被剪枝，最短哈密顿回路是0->2->1->3。