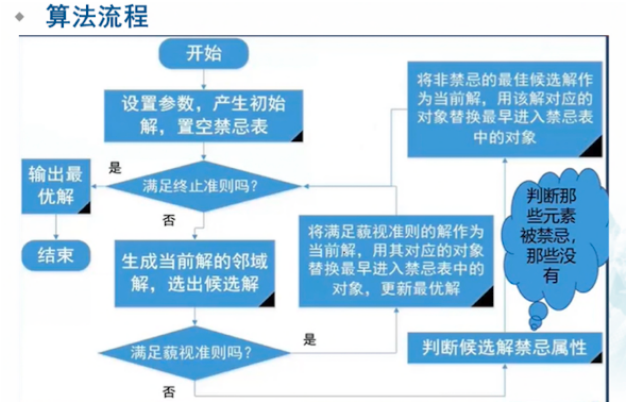
禁忌搜索算法解TSP问题

算法实现

1. 初始化最优解，禁忌表，藐视准则
2. 迭代
    1. 遍历邻域
       在禁忌表中，更新藐视最优解
       不在禁忌表中，更新迭代的最优解
    2. 判断藐视最优解是否满足条件
    3. 对迭代最优解进行转移，并更新禁忌表
3. 输出全局最优解



关键代码

- 读入城市数据，并计算城市距离

```
void readAndInit(int n){
    for(int i=0;i<n*2;i++)
        city[i/2][i%2] = coordinate[i];
        for(int i=0;i<n;i++)
            for(int j=0;j<n;j++)
                adj[i][j]=sqrt((city[i][0]-city[j][0])*(city[i][0]-city[j][0])+(city[i][1]-city[j][1])*(city[i][1]-city[j][1]));


}
```

- 初始化禁忌表

```
//初始化禁忌表
int now=0;
for(int i=0;i<n;i++)
    for(int j=i+1;j<n;j++){
        TabuList[now][0]=i;
        TabuList[now][1]=j;
        TabuList[now][2]=0;
        now++;
    }
```

- 转移最优解，对每次迭代后的最优解进行更新

```
for(int i=0;i<K;i++){//值得注意 每次搜索禁忌表是共享的 也就是新的小搜索不会重置禁忌表
    int smallSearchDis=smallSearch(n);
    if(finalDis>smallSearchDis){//如果此次小型搜索的最优解优于finalDis,则更新
        finalDis=smallSearchDis;
        memcpy(finalBestPath,nowPath,sizeof (nowPath));//路径复制
    }
```

- 邻域搜索，从初始解的邻域开始，进行局部搜索；在邻域搜索中，检查当前解是否在禁忌表中，若在表中，则检查是否可以被特赦；若不在表中，则更新迭代最优解

```
getRandomOrder(nowPath,n);
double bestDis=getPathValue(nowPath,n);//初始化小型搜索最优解

int pardon[2], curBest[2];//特赦最优解和搜索最优解
pardon[0]=pardon[1]=curBest[0]=curBest[1]=INF;//初始化
int LNum=n*(n-1)/2;//邻域数量
for(int i=0;i<ITERATIONS;i++){//迭代
    for(int j=0;j<LNum;j++){//领域搜索
        swap(nowPath[TabuList[j][0]],nowPath[TabuList[j][1]]);
        double tmpDis=getPathValue(nowPath,n);
        if(TabuList[j][2]==0){//没有被禁忌
            if(tmpDis<curBest[1]){
                curBest[0]=j;
                curBest[1]=tmpDis;
            }
        }
        else{//被禁忌
            if(tmpDis<pardon[1]){
                pardon[0]=j;
                pardon[1]=tmpDis;
            }
        }
        swap(nowPath[TabuList[j][0]],nowPath[TabuList[j][1]]);
    }
```

- 更新禁忌表，将本次更新后的迭代最优解加入禁忌表中，并对已在表中的解进行更新

```
if(curBest[1]<bestDis){
    bestDis=curBest[1];
    //交换位置
    swap(nowPath[TabuList[curBest[0]][0]],nowPath[TabuList[curBest[0]][1]]);
    //更新禁忌表
    TabuList[curBest[0]][2]=TABU_SIZE;
    for(int j=0;j<LNum;j++)
        if(TabuList[j][2]>0)
            TabuList[j][2]--;
}
```

运行结果

最优解如下图：

最短距离为：16784.6

```
路线长度： 16784.6
路线： ->16->18->23->24->11->13->0->14->12->5->6->9->8->7->3->1->4->15->22->10->28->30->29->26->27->25->19->20->21->17->2
```

问题与收获

在学习该算法的过程中，对相关参数概念有点模糊不解，参考网上前辈的思路才清晰一点，对比不同的实现代码发现对最终的结果会有较大差异，不太清楚其中原因，但路线长度的范围大致在15000-20000之间。

禁忌搜索算法是一种模拟人记忆的启发式算法，在对邻域搜索时，对局部最优解进行标记即加入禁忌表，然后在后续搜索中尽量避免这些局部最优解从而尽量达到全局最优，在有限时间内给出最优解。