差分约束算法总结

差分约束系统

一、概念

如果一个系统由 n 个变量和 m 个约束条件组成，形成 m 个形如 ai-aj≤k 的不等式 (i,j∈[1,n],k 为常数), 则称其为差分约束系统。

二、引例

给定 n 个变量和 m 个不等式，每个不等式的形式为 x[i] - x[j] <= a[k] (0 <= i, j < n, 0 <= k < m， a[k] 已知)，求 x[i] - x[j] 的最大值。

例如当 n = 4， m = 5，给出如下图所示的不等式组，求 x3 - x0 的最大值。

$$X_1-X_0<=1 \qquad (1)$$
$$X_2-X_0<=2 \qquad (2)$$
$$X_3-X_0<=4 \qquad (3)$$
$$X_2-X_1<=3 \qquad (4)$$
$$X_3-X_2<=1 \qquad (5)$$

一般思路：我们可以尝试把几个不等式组合得到最后我们要求的式子，于是这些式子里最小的那个就是答案。

比如，在这个例子中：

（3） ==》 x3 - x0<=4；

（1）、（4）、（5） ==》 x3 - x0<=5；

（2）、（5） ==》 x3 - x0<=3;

所以最后结果就是 3.

在这个过程中，我们是否想到这种方法与我们已经学的一种算法有所联系，是的，就是最短路算法。

三、差分约束与最短路模型

1、与最短路模型的联系

先给出结论：求解差分约束系统，都可以转化成图论的单源最短路径（或最长路径）问题。

我们观察上面例子中的不等式，都是 x[i] - x[j] <= a[k]，可以进行移项，成为 x[i] <= x[j] + a[k]，我们令 a[k] = w(j, i)，dis[i]=x[i], 并使i=v,j=u, 那么原始就变为：dis[u]+w(u,v)>=dis[v], 于是可以联想到最短路模型中的一部分代码

```
if(dis[u]+w(u,v)<=dis[v])
{
    dis[v]=dis[u]+w(u,v);
}
```

这不正与松弛操作相似吗?

但是好像不等号方向刚好相反，但其实这并不矛盾

上面的代码要实现的是使 dis[u]+w(u,v)>dis[v], 而对于不等式，我们进行建边的操作：对于每个不等式 x[i] - x[j] <= a[k]，对结点 j 和 i 建立一条 j -> i 的有向边，边权为 a[k]，求 x[n-1] - x[0] 的最大值就是求 0 到 n-1 的最短路，两者刚好吻合。所以求解差分约束问题就转化为了最短路问题。

2. 问题解的存在性

　由于在求解最短路时会出现存在负环或者终点根本不可达的情况，在求解差分约束问题时同样存在

（1）、存在负环

如果路径中出现负环，就表示最短路可以无限小，即不存在最短路，那么在不等式上的表现即 X[n-1] - X[0] <= T 中的 T 无限小，得出的结论就是 X[n-1] - X[0] 的最大值不存在。**在 SPFA 实现过程中体现为某一点的入队次数大于节点数。（貌似可以用 sqrt(num_node) 来代替减少运行时间）**

（2）、终点不可达

这种情况表明 X[n-1] 和 X[0] 之间没有约束关系，X[n-1] - X[0] 的最大值无限大，即 X[n-1] 和 X[0] 的取值有无限多种。在代码实现过程中体现为 dis[n-1]=INF。

3、不等式组的转化

做题时可能会遇到不等式中的符号不相同的情况，但我们可以对它们进行适当的转化

（1）方程给出：X[n-1]-X[0]>=T , 可以进行移项转化为： X[0]-X[n-1]<=-T。

（2）方程给出：X[n-1]-X[0]<T, 可以转化为 X[n-1]-X[0]<=T-1。

（3）方程给出：X[n-1]-X[0]=T，可以转化为 X[n-1]-X[0]<=T&&X[n-1]-X[0]>=T，再利用（1）进行转化即可

4、应用

对于不同的题目，给出的条件都不一样，我们首先需要关注问题是什么，如果需要求的是两个变量差的最大值，那么需要将所有不等式转变成 "<=" 的形式，建图后求最短路；相反，如果需要求的是两个变量差的最小值，那么需要将所有不等式转化成 ">="，建图后求最长路。