****

**算法分析与设计实验报告**

****

**学 院： 电子信息与人工智能学院**

**专业名称： 计算机科学与技术**

**班 级： 计算机203**

**学 号： 202007020625**

**姓 名： 於俊涛**

**任课老师： 齐 勇**

# 实验3 回溯算法实验

班级 计算机203 学号 202007020625 姓名 於俊涛 角色 designer

**一、实验目的**

1、掌握拓扑排序、最短路径、Dijkstra算法、Prim算法、Floyd算法相关知识、以及不同算法的性能。

2、了解各自算法的内容，尝试使用不同的编程语言来实现代码。

**二、实验环境**

线上平台sustoj.com

**三、实验内容**

**（一）算法效率分析**

**3.1 Prim算法**

Prim算法的具体效率，是与图的顶点和边数有关的。这取决于所选择的表示图本身的数据结构，以及表示集合V-Vr的优先队列的数据结构，集合中顶点的优先级就是到最近的树的顶点的距离。

假设总共有n个顶点，那么肯定有n次比较过程！而在第k次比较过程中，又有n-k个点和另外n-k个顶点相互比较，那么总结起来，那么算法复杂度几乎再n的立方级别。n³级别复杂度的确不是很高，但其实prim算法真正效率是介于n²和n³之间的，因为我们如上的分析是一种最坏的情况，就是每个点之间都存在一条路径。

在查找最小生成树这一问题中，Prim算法算是高效的算法，尽管算法的复杂度为n3。举个例子，如果试图用一种穷举法查找算法来构成一棵最小生成树，会遇到两个严重的阻碍，首先，随着图的规模的增长，生成树的数量呈指数增加（至少稠密图是如此）。其次，生成一个给定图的所有生成树也并非易事。事实上，这比用一些高效的算法来求加权图的最小生成树还要困难。

最小生成树算法Prim算法的时间复杂度为O（E\*lgV）。

而Prim算法若采用斐波那契堆实现的话，算法时间复杂度为O（E+V\*lgV），当|V|<<|E|时，E+V\*lgV是一个较大的改进。

**3.2 最短路径（两种算法的区别）**

Floyd 算法虽然总体上时间复杂度较高，但可以处理带负权边的图（但不能有负权回路），并且均摊到每一点对上，在所有的算法中还是属于比较优秀的算法。另外，floyd算法较小的编码复杂度也是一大优势，所以，如果要求的是所有点对间的最短路径，或者如果数据范围较小，则floyd算法比较合适。

Dijkstra算法最大的弊端就是他无法处理带有负权边以及负权回路的图，但是Dijkstra算法具有良好的可扩展性，扩展后可以适应很多问题。另外用堆优化的Dijkstra算法的时间复杂度可以达到O（M log N）。当边有负权，甚至存在负权回路时，需要使用Bellman-ford 算法或者队列优化的Bellman-ford算法，因此我们选择最短路径法时，根据实际的需求和每一种算法的特性，选择合适的算法来使用。

Dijkstra算法适用场景：

一般算法书中都说适合于有向无环图（DAG）。但是并不代表无向图就不行，只要将源点和目标点的权重相互之间赋值成一个值即可。如果是有环图的话，对于每个节点都是做了一层遍历，也不会出现死循环和重复计算的问题，因此迪杰斯特拉算法是可以用在无向图和有环图中的，适合于求单源最短路径。如果想适用于多源最短路径，就要将每个节点都进行遍历，那么时间复杂度就是O(n³)。

**3.3 拓扑排序（算法优化）**

先将入度为0的顶点放在栈或者队列中。当队列不空时，删除一个顶点v，然后更新与顶点v邻接的顶点的入度。只要有一个顶点的入度降为0，则将之入队列。此时，拓扑排序就是顶点出队的顺序。该算法的时间复杂度为O（V+E）。

**（二）方案设计**

**1、签到题之青蛙爬楼梯**

楼梯有n阶台阶，青蛙每次可以跳1~n阶台阶，问青蛙共有多少种上楼梯的方法。

方法一：可以看做是一个完全背包问题

1阶，2阶，….n阶就是物品，楼顶就是背包。每一阶可以重复使用，例如跳了一阶还可以继续跳一阶。问跳到楼顶有几种方法其实就是问装满背包有几种方法。

具体设计步骤：

1、确定dp数组以及下标的含义

dp[i]：爬到有i个台阶的楼顶，有dp[i]种方法

2、确定递推公式

dp[i]+=dp[i-j]

3、dp数组初始化

既然递归公式是 dp[i] += dp[i - j]，那么dp[0] 一定为1，dp[0]是递归中一切数值的基础所在，如果dp[0]是0的话，其他数值都是0了。

下标非0的dp[i]初始化为0，因为dp[i]是靠dp[i-j]累计上来的，dp[i]本身为0这样才不会影响结果

4、确定遍历顺序

这是背包里求排列问题，即：1、2 步 和 2、1 步都是上三个台阶，但是这两种方法不一样！

所以需将target放在外循环，将nums放在内循环。

每一步可以走多次，这是完全背包，内循环需要从前向后遍历。

C++代码如下：

class Solution {

public:

int climbStairs(int n) {

vector<int> dp(n + 1, 0);

dp[0] = 1;

for (int i = 1; i <= n; i++) { // 遍历背包

for (int j = 1; j <= m; j++) { // 遍历物品

if (i - j >= 0) dp[i] += dp[i - j];

}

}

return dp[n];

}

};

方法二：暴力解法，用回溯算法进行暴力搜索，枚举所有的情况。

由于时间复杂度过高，放弃这个方案。

方法三：简单数学找规律，当n=1时，有一种解法（1）

当n=2时，有两种解法（1,1）（2）

当n=3时，有4种解法（1,1,1）（1,2）（2,1）（3）

以此类推，当有n阶楼梯时，有2（n-1）种方法。

具体方案步骤：

直接使库函数：pow即可。

当n<=50时，long long result=pow(2,(n-1))

**2、排名**

有 N 个比赛队（1<=N<=500），编号依次为 1，2，3……N 进行比赛，比赛结束后，裁判委员会要将所有参赛队伍从前往后依次排名，但现在裁判委员会不能直接获得每个队的比赛成绩，只知道每场比赛的结果，即 P1 赢 P2，用 P1，P2 表示，排名时 P1 在 P2之前。现在请你编程序确定排名。

符合条件的排名可能不是唯一的。此时要求输出时编号小的队伍在前。输入数据保证是正确的，即输入数据确保一定能有一个符合要求的排名。

方案如下：由题意可知这是拓扑排序，但是它要求输出时小的队伍在前面。这样就用到了优先级队列的知识，可以使用STL的priority\_queue容器。优先队列是队列的一种，不过它可以按照自定义的一种方式（数据的优先级）来对队列中的数据进行动态的排序，每次的push和pop操作，队列都会动态的调整，以达到我们预期的方式来存储。

**涉及到的相关知识**

①优先级队列

自定义编号小的队伍在前

priority\_queue<int,vector<int>,greater<int> >q;//优先队列，数据越小，优先级越高

②图的拓扑排序

给出有向图G=(V,E)， 对于V中的顶点的线性序列(v1,v2,...,vn)， 如果满足如下条件： 若在G中从顶点 vi 到vj有一条路经，  则在序列中顶点vi必在顶点vj之前； 则称该序列为G的一个拓扑序列。

构造有向图的一个拓扑序列的过程称为拓扑排序。

**具体设计步骤：**

1.统计每个结点的入度，将度为0的结点编号放入队列（此题放入优先队列中）中。

2.进行循环：

①取出队头结点，视作边的起点。

②删除与该点相连的边，即将这个图中的该边另一个结点（即终点）的入度减一。

③如果减一以后，终点的入度变为了0，那么将终点的编号入队列。

④判断队列是否为空，若不为空回到①。

**3、系兄弟就来砍我**

渣渣灰因为一句“大家好，我系渣渣辉，系兄弟就来砍我”引得众粉丝纷纷拿两米长的大刀寻找。现有n个据点，编号（1~n），有m条单向路使据点相连。每个据点仅有一个人。这n个人中有k个粉丝。其中渣渣灰在s据点处。请问这k个粉丝到渣渣灰的最短距离是多少。

方案一：

根据题意可知，是典型的单源求最短路径的问题，采用Dijkstra算法。

**具体方案步骤**：

①Dijkstra算法求单源最短路裸题，因为要求k个粉丝（起点）到 S 终点的最短距离，如果正向建边，求k次最短路会超时

②所以考虑逆向建边，求S点到所有点的最短距离，储存在dis数组中

③既然要求k个点到s点的最短路，我们可以反过来求s到这k个点的最短路。这样就变成了单源最短路问题，dijkstra算法和spfa算法都可以做。由于图为有向图，我们在存图时反向存图即可，原本a[i][j]表示i->j，我们可以将它重新定义为j->i，或者存图时直接写成a[j][i]即可。

时间复杂度分析：

在核心代码部分，最复杂的是 while 循环和 for 循环嵌套的部分，while 循环最多循环 v 次（v 为顶点个数），for 循环执行次数与边的数目有关，假设顶点数 v 的最大边数是 e。

for 循环中往优先队列中添加删除数据的复杂度为O(log v)。

综合上述两部分，最终 Dijkstra 算法的时间复杂度是O(e·logv)

方案二：分支限界法求单源最短路径。

用优先队列式分支限界法（最小耗费优先法）设计并实现算法，求单源最短路径问题。

**4、逛街**

假设渣渣灰有一个女朋友，他的女朋友要他陪着一起去公园。由于渣渣灰不喜欢运动，所以他想找一条最短的路到达公园。由于途中会有许多消费点，而每到一个消费点女朋友就要购物，而渣渣灰比较抠，所以假如有多条最短路，则他会选择途中消费点最便宜的。给你n个点，m条无向边，每条边都有长度d和花费p，给你起点s终点t，要求输出起点到终点的最短距离及其花费，如果最短距离有多条路线，则输出花费最少的。

方案如下：

dijkstra算法求单源最短路简单扩展，我们再创建一个value数组储存花费情况。在松弛时对value进行改变。

具体方案设计：

松弛成功则value（s->i）=value(s->k->i)。

若最短路相等则对value值进行比较，即value（s->i）=min(value(s->k->i),value（s->i))。

s为源点，i为当前终点，k为中间点。最终输出最短路及对应value值即可。

**四、实验小结**

本次实验，掌握了拓扑排序、最短路径、Dijkstra算法、Prim算法、Floyd算法相关知识、以及不同算法的性能。