文章编号：2096-1472(2021)-10-41-03
DOI:10.19644/j.cnki.issn2096-1472.2021.010.010

层级结构型数据的联通性研究

贺军忠

(陇南师范高等专科学校，甘肃 成县 742500)
✉lnszhjztg@163.com

摘　要：针对结构复杂、联通算法难以实现的层级结构联通性问题，提出了利用树结构的最长路径算法解决层级结构型数据的方法。本研究以传统城堡防护问题为例，找出中间相隔的城墙数量最多的两个点，即层级结构的联通厚度。通过数据结构分析，将城堡各区域转化为树结构，验证了树结构的最长路径即为层级结构的联通厚度，最终巧妙实现将层级结构转换为树结构，并利用树结构的遍历分析寻找到树的最长路径递归算法，解决了树节点到节点的最长路径，最终实现层级结构各层的联通路径建设规划，为层级结构问题的解决找到突破口与参考性。

关键词：层级结构；生成树结构；联通算法；最长路径
中图分类号：TP399　**文献标识码**：A

Research on the Connectivity of Hierarchical Structured Data

HE Junzhong

(*Longnan Teachers College, Chengxian* 742500, *China*)
✉lnszhjztg@163.com

Abstract: Aiming at the problem of hierarchical structure connectivity that is difficult to achieve with the complex structure, this paper, proposes a method of using the longest path algorithm of the tree structure to solve the problem of hierarchical structure data. This research takes traditional castle protection problem as an example. It is necessary to find the two points, between which there is the largest number of walls, namely the connectivity thickness of the hierarchical structure. Through data structure analysis, each area of the castle is transformed into a tree structure to verify that the longest path of the tree structure is the connectivity thickness of the hierarchical structure. Finally, hierarchical structure is ingeniously transformed into tree structure, and the traversal of the tree structure is used to analyze and find the longest path recursive algorithm of the tree. It solves the longest path from the tree node to the node. Finally, the proposed method realizes the connection path construction planning of each layer in the hierarchical structure, and makes breakthroughs and provides references for solving hierarchical structure problems.

Keywords: hierarchical structure; spanning tree structure; connectivity algorithm; longest path

1 引言(Introduction)

在无法利用线性结构表示的数据中，层级结构最常见。数据间存在上下级关系或包含关系时，就会产生层级结构。世界杯决赛的对阵表、公司或学校的组织结构图、网店商品的分类标准等，都具有层级结构。表示层级结构数据的数据结构就是树(tree)。若某一概念包含另一概念，就将两个概念连接为上下级形式。这种从一个上级概念分割出许多从属概念的形状与树非常相似，都可以转化为树结构解决。

2 问题概述(Problem overview)

在古代，为保护城堡，防止敌人侵，更好地守卫和保护更多领地，城堡中心都建有多层城墙，如图1所示。其中主建筑的Strawgoh要塞达到了这种建城模式的极致。巨大的圆形外墙包含若干圆形内墙，所有城墙均未设置城门。因此，要想进出必须使用梯子，在要塞内部从某地移动到另一地也需

基金项目：2021年陇南市科技计划项目《乡村振兴视角下农村青年电商创业研究——以陇南地区为例》阶段性研究成果；2019校级教改项目《“1+X 证书制度”背景下网店运营推广人才培养模式研究与实践》阶段性研究成果(JXGG2019006)。

要耗费大量时间。为此，城主决定挖几条地道，以连接来往不便的几个地方。为了设计出合适的建设计划，需要找出中间相隔的城墙数量最多的两个点。例如，图1中的两个星号表示的地点相隔了5 层城墙[1]。

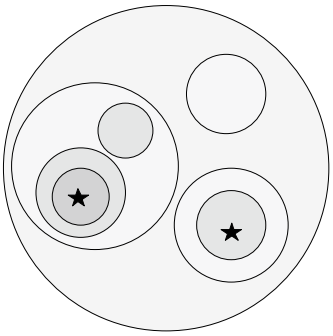
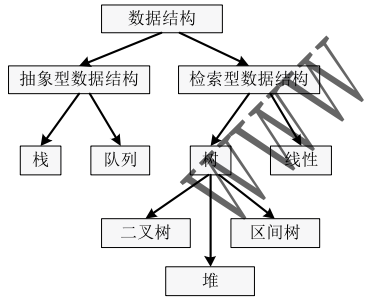


图1 城堡防卫结构图

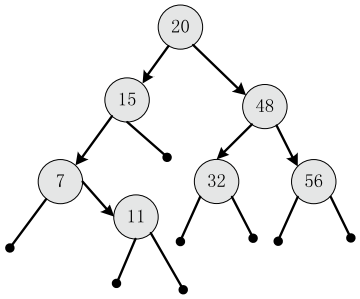
Fig.1 Castle defense structure diagram

给出城墙信息时，编写程序计算出从某地移动到另一地需要翻过的城墙最多会有几层。此问题表面上看起来与树结构并无直接关系，但由“所有城墙不会相交或相接”可知，城墙有很规则的层次结构，层级关系较为明确，从而可以将城墙间的包含关系转化为树结构求解。

图2(a)是表示各数据结构关系的树结构图。结构图中的每个方框表示1 种数据结构，上面的方框表示上级概念，下面的方框表示从属概念。若某一概念包含另一概念，就将两个概念连接为上下级形式。这种从一个上级概念分割出许多从属概念的形状与树非常相似(虽然上下颠倒)，故称为“树”，如图2(b)所示。层级结构与人类的生活关系非常密切，所以得到了广泛应用。



(a)数据结构图

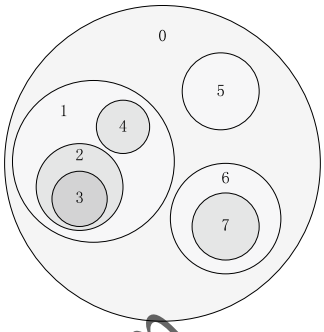


(b)二叉搜索树

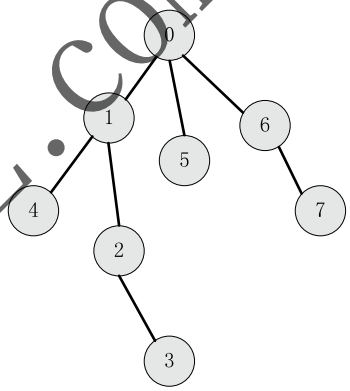
图2 数据结构树与二叉搜索树对比

Fig.2 Comparison of data structure tree and binary search tree

为此，给城墙分割的各个“区域”标上序号。将围绕当前区域的城墙序号用作该区域序号，得到图3(a)。通过这种连接就能把包含关系表示为树结构，如图3(b)所示。如果两个区域不被同一城墙分割，那么它们不会在树结构中相互连接。例如，1 号区域间接包含3 号区域，所以并没有在树结构中直接相连。



(a)城堡各区域图



(b)树结构图

图3 城堡各区域转化的树结构

Fig.3 The transformed tree structure of each area of the castle

转化为树结构后，从一个区域移动到相邻其他区域的过程将对应于树结构中沿着树的边从一个节点移动到另一个节点的过程。因此，两个区域间经过的最多城墙数问题就能变换成在树结构中寻找相隔最远的两个节点的问题。此时，连接两个节点的各边就是树的路径[1]。

3 树的最长路径(The longest path of the tree)

上述建模过程将问题变换为“找出给定树结构中最长路径”的问题。而求树的最长路径并不难，求解关键在于需要明白最长路径的两端必须是根节点或叶节点。

假设，既不是根节点也不是叶节点的内部节点(Internal Node)是路径终点，而内部节点总是连接两条以上的边。因为此节点不是根节点，所以会有连向父节点的边；又不是叶节点，所以会有连向子节点的边。因此，以内部节点为终点的路径必定至少存在1条未使用的边。利用此边就能生成更长的路径，所以它不可能是最长路径[2]。

由此可知，最长路径会是如下两种路径中的较大值：最长的根—叶路径长度和最长的叶—叶路径长度。

此时，最长的根—叶路径长度等于树的高度，我们已经知道求解方法。现在只要求出最长的叶—叶路径长度，就能得出答案。

叶—叶路径的形式总是从叶节点到达某个节点后，又回到另一个叶节点。例如，图2(b)中连接3和4的路径，首先从3到达1之后，又返回4；而连接3和7的路径，首先从3到达0之后，又返回7。这种先上升后下降的路径中，处在上下转换位置的节点就是此路径的顶端节点。因此，在树的遍历过程中找出将各节点用作顶端节点的最长叶—叶路径，选择其中的最大值即可[3]。

该如何求将给定节点用作顶端节点的最长叶—叶路径呢？这种路径的形态是：当前节点的后代节点中，从一侧上升到该节点，然后从另一侧下降。此时，上升部分和下降部分的长度等于将各自的后代节点用作根节点的子树高度加1。因此，先计算出各子树的高度后，找出最高的两个子树就能求出最长叶—叶路径。

这些过程相当于求树的高度的运算过程。因此，变换一下求树高的递归函数就能求出此题。代码1表示解决此问题的递归调用函数。代码中，已找出的最长路径的长度会保存到全局变量longest，而height()函数通过更新此变量完成操作。另外，利用最高的两个子树的高度求出将当前节点用作顶端节点的叶—叶路径中的最长路径[4]。

代码1：找出树结构中最长路径的递归算法。

```
struct Treellode {
  vector<Treenode*> children;
};
// 保存已找出的最长叶—叶路径的长度
int longest;
//返回将root用作根节点的子树的高度
int height(TreeNode* root){
//计算将各子节点用作根节点的子树高度
  vector<int> heights;
  for(int i=0;i < root->children.size();++i)
    heights.push_back(height(root->children[i]));
//没有子节点就返回0
  if(heights.empty()) return 0;
  sort(heights.begin(),heights.end());
//考虑将root用作顶端节点的路径
  if(heights.size()>=2)
    longest=max(longest,2+heights[heights.size()-2]+
          heights[heights.size()-1];
//将子树高度加1以算出树的高度
    return heights.back()+1;
}
//计算两个节点之间的最长距离
int solve(TreeNode* root){
  longest=0;
//选择树的高度和最长叶—叶路径中更大的值
  int h=height(root);
  return max(longest,h);
}
```

其中height()函数在处理整个树结构的过程中会耗费时间，若是忽略子树高度的排序时间O(nlgn)，就会与树的遍历时间相同，都是O(n)[5]。

4　算法的实现(Algorithm implementation)

此问题的实际实现方法可分为两部分：从输入值生成树结构；在生成的树结构中求出最长路径。求最长路径的部分如前所述，接下来介绍树结构生成过程。

最直观的生成树结构的方法是从树的根节点开始生成。第0 号城墙是包含其他所有城墙的外墙，所以它会成为树的根节点。找出第0 号城墙的下一层城墙，并通过递归调用生成将各城墙用作根节点的子树。代码2是将给定序号的城墙用作根节点并生成树结构的getTree()函数[6]。

代码2：生成表示给出城墙中各区域的树结构。

```
//生成将root城墙用作根节点的树
TreeNode* getTree(int root){
  TreeNode* ret=new Treenode();
  for(int ch=0;ch<n;++ch)
//如果ch城墙直接包含于root城墙，那么生成树后再添加到后代目录
  if(isChild(root,ch))
    ret->children.push_back(getTree(ch));
  return ret;
}
```

getTree()中的isChild()函数可用多种方法实现。不过，实现此函数最简单的方法是，根据其他城墙的信息判断是否存在于两个城墙之间。例如，图2(a)中，第0 号城墙和第4 号城墙之间存在第1 号城墙，因此，第4 个区域并没有直接包含于第0 号区域。代码3表示实现这种简单方法的isChild()函数[7]。

代码3：确认某个城墙是否包含且直接包含于另一城墙的函数。

```
//输入数据
int n,y[100],x[100],radius[100];
//返回x^2
int sqr(int x){
  return x*x;
}
//返回两个城墙a和b的圆心点距离的平方值
```

```
int sqrdist(int a,int b){
  return sqr(y[a]-y[b])+sqr(x[a]-x[b]);
}
//确认城墙a是否包含城墙b
bool encloses(int a,int b){
  return radius[a]>radius[b] &&
    sqrdist(a,b)<sqr(radius[a]-radius[b]);
}
//在"城墙"树结构中确认parent是否为child的父节点
// parent必须直接包含 child
bool iscChild(int parent,int child){
  if(!encloses(parent,child)) return false;
    for(int i=0;i<n;++i)
        if(i !=parent && i !=child &&
            encloses(parent,i)&& encloses(i,child))
            return false;
    return true;
}
```

通过这种实现方法，isChild()函数能够在O(n)的时间内完成运算。树结构生成过程中，每访问1 个节点，就会调用n 次isChild()。因此，整个算法的时间复杂度为$O(n^2)$[8]。

5 结论(Conclusion)

本研究根据城堡防卫结构图及联通性需求，利用层级结构和树结构关系，将不容易直接解决的层级结构数据转化为树结构，从而将城堡的联通性结构设计转化成了树结构的最长路径问题来解决。通过寻找树结构的最长路径优化算法，最终实现城堡的最大联通值。同时，通过解决城堡防卫联通性问题，得出利用树结构解决层级结构的方法，为解决层级结构问题的相关研究提供了依据。

参考文献(References)

[1] 计算思维百科.“要塞”的版本间的差异[EB/OL].(2016-06-02)[2021-06-05]. https://wiki.jsswsq.com/index.php?diff=prev&oldid=3711&title=%E8%A6%81%E5%A1%9E.

[2] RAI A K, KUMAR N. Intra-block correlation based reversible data hiding in encrypted images using parametric binary tree labeling[J]. Symmetry, 2021, 13(6):187-189.

[3] 杨迪,马金全,岳春生,等.面向异构处理平台的最长路径列表调度算法[J].信息工程大学学报,2021,22(02):136-141,214.

[4] 王前东.一种带匹配路径约束的最长公共子序列长度算法[J].电子与信息学报,2017,39(11):2615-2619.

[5] 潘静静.林区轮伐期优化的隐式图建模和最长路径算法[J].河南科技大学学报(自然科学版),2016,37(01):73-77,8-9.

[6] 刘剑,宋莹,邓立军.基于GA与最长路径并联通路法优化通风网络图绘制[J].中国安全生产科学技术,2014,10(11):77-83.

[7] 王防修,周康.基于最长公共子序列的随机路径选择算法设计[J].计算机工程与设计,2014,35(06):2170-2173.

[8] 张琦,金胤丞,李苗,等.Trie树路由查找算法在网络处理器中的实现[J].计算机工程,2014,40(1):98-102.

作者简介：

贺军忠(1982-)，男，硕士，副教授/网络工程师.研究领域：网络组建与信息安全，网络营销.

(上接第53页)

程,2017,38(19):185-189.

[2] 崔洁,杨凯,肖雅静,等.步进电机加减速曲线的算法研究[J].电子工业专用设备,2013,42(08):45-49.

[3] 范立云,高明春,马修真,等.电控单体泵高速电磁阀电磁力关键影响因素[J].内燃机学报,2012,30(04):359-364.

[4] 史小露,郑友胜,张磊.基于ROS的智能代步车嵌入式运动控制系统[J].软件工程,2016,19(06):48-51.

[5] JEON J W, HA Y Y. A generalized approach for the acceleration and deceleration of industrial robots and CNC machine tools[J]. IEEE Transactions on Industrial Electronics, 2000, 47(1):133-139.

[6] LU T C, CHEN S L. Genetic algorithm-based S-curve acceleration and deceleration for five-axis machine tools[J]. The International Journal of Advanced Manufacturing Technology, 2016, 87(1):219-232.

[7] LIU Q F, YAN B H, LI W. Research of step motor on acceleration and deceleration control method[J]. Electric Machines & Control Application, 2017, 44(3):36-39.

[8] 罗钧,汪俊,刘学明,等.基于S型加减速的自适应前瞻NURBS曲线插补算法[J].计算机集成制造系统,2013,19(01):55-60.

[9] 李娟,张波,丘东元.电能质量监测系统中基于Modbus RTU的多机通信[J].电力自动化设备,2007(01):93-96.

[10] 赵晓军,曹建坤,李可一,等.基于CAN总线的数据臂通信设计[J].自动化仪表,2010,31(05):13-15,18.

作者简介：

郑明航(1996-)，男，硕士生.研究领域：智能检测与应用.

彭来湖(1980-)，男，博士，副教授.研究领域：智能装备与嵌入式控制技术，工业互联网通信.

史伟民(1965-)，男，博士，教授.研究领域：纺织机械自动控制，轻工机械.