山东大学 计算机科学与技术 学院

可视化技术 课程实验报告

|  |  |  |  |
| --- | --- | --- | --- |
| 学号：201900130176 | 姓名： 李伟国 | | 班级： 智能 |
| 实验题目： RT算法实现radial tree layout | | | |
| 实验学时： | | 实验日期： 2021/10/23 | |
| 实验目的： | | | |
| 硬件环境：  处理器：AMD Ryzen 5 3600 6-Core Processor 3.60 GHz  Ram 16.0 GB | | | |
| 软件环境： | | | |
| 实验步骤与内容：  Tree 的表示方法：   1. Identation（windows 文件夹） 2. 节点表示法（根节点，子节点，叶子节点） 3. 闭包：表示层次结构 4. 分层法（相对位置和对齐）   Layout：  Naïve recursive Layout:按照叶子的数量不断的划分子树空间  Reingold & Tiflord layout：空间的利用率会很高，密度最大，对称性很强，compact space.  首先后序遍历到”shifts”,然后将右子树想左子树靠近，这样递归的不断进行下去，最后再来一遍前序遍历去设置breadth dimension的坐标。  详细步骤：假设要绘制的原始的tree 的结构如下：     1. First Traversal of Tree:首先run 一次post-order traversal（后序遍历），意味着是从bottom 向up，从左到右来绘制tree 的。原图中A~G就表示了后序遍历的顺序 2. Assigning initial X values：对每一个node都计算local X的值。这是仅仅只关于子节点的X 值。赋值的规则是，如果该节点左边没有其兄弟节点，那么就赋值为0，否则的话，就是其左兄弟节点的值加一。初始化赋值后如下图：     此时如果直接将初始的X值作为各个节点在X axis的坐标绘制，就会出现很多的节点之间相互overlap（重叠），如下图所示。所以下一步要做的就是将子节点放置在其各自的父节点之下。     1. The mod property   剩余的步骤涉及到将node 和其子节点位移很多次。为了能够这样做，通常必须遍历该子树中的所有节点并且增加其X的值。但这样的性能不好，尤其是那种很大的树。为了避免这个问题，我们使用Mod property 去告诉node，它应该将其子节点移动多远，之后我们将会第二次遍历（前序遍历）整个树去决定每个节点的最终的X的值。  去移动一个节点和其子节点在这次迭代中，增加节点的X的值和Mod property 的值，增加的量随我们。   1. Positioning Child Nodes under parents   这一步是将子节点剧中置于父节点下。  首先，找到一个X的值，使得该节点能够位于其子节点的中心。   1. 如果节点只有一个孩子，那么渴望的X值就是跟其子节点的X的值一样 2. 如果该节点有多余一个的孩子，那么渴望的X的值就是第一个孩子和最后一个孩子的中点的X的值。   接下来，检查该父亲节点的左边是否有其兄弟节点，可以通过检查该节点是否是其父节点的第一个节点。   1. 如果该节点是其父节点的第一个子节点，就将该节点的X的值设置为渴望的X的值， 2. 如果不是的话，就将该节点的Mod 值改成node.x – 渴望的X的值。来移动父节点下的子节点。   Reingold-Tilford algorithm  Radial Tree：  使用的是极坐标系，极径代表了树的深度，而极点代表了树的根部，角度扇区分配给了子树，这样的结构是递归的布局。可以使用RT algorithm 去布局 | | | |
| 结论分析与体会： | | | |

附录：程序源代码