

# 研究型教学报告

|  |  |  |
| --- | --- | --- |
| 学 号： | 17281180 | |
| 姓 名： | 解仕奥 | |
| 专 业： | 计算机科学与技术 | |
| 学 院： | 计算机与信息技术学院 | |
|  |  |  |
|  |  |  |
| 提交日期： | 2019年06月7日 | |

摘要

本文对二叉平衡树的构建，插入，删除，可视化进行了研究，给出了相应的规则，通过递归算法，用c语言和c++编程，参考二叉排序树和红黑树给出了二叉平衡树的动画演示的可视化界面，便于理解。使用html&&javascript语言生成网页，用VS进行调试，实现可视化界面，并增加了交互的功能，动态过程说明详细。

关键词：二叉平衡树 插入 删除 可视化 html &&JS vs 交互

目录

[1.研究对象 - 1 -](#_Toc21228)

[2.研究方法 - 1 -](#_Toc25235)

[3.研究内容 - 1 -](#_Toc15595)

[3.1基本数据结构的确立 - 1 -](#_Toc7590)

[3.2二叉平衡树的构建 - 1 -](#_Toc18717)

[3.3二叉平衡树的四种不平衡类型 - 2 -](#_Toc18160)

[3.3.1 LL型（左孩子的左子树上添加） - 2 -](#_Toc20804)

[3.3.2 RR型（右孩子的右子树上添加） - 3 -](#_Toc22928)

[3.3.3 LR型（右子树的左孩子上加节点） - 5 -](#_Toc29376)

[3.3.4 RL型（左子树的右孩子上加节点） - 5 -](#_Toc22819)

[3.4 二叉平衡树的插入 - 5 -](#_Toc30611)

[3.5 二叉平衡树的删除 - 6 -](#_Toc12591)

[4. 二叉平衡树的可视化 - 9 -](#_Toc26758)

[4.1可视化方法：html和JS语言 - 9 -](#_Toc27947)

[4.2使用工具：VS2015 - 9 -](#_Toc6569)

[4.3界面展示： - 10 -](#_Toc30107)

[4.4功能介绍： - 10 -](#_Toc19316)

[5.研究成果呈现 - 10 -](#_Toc8969)

[6.研究结果分析和讨论 - 10 -](#_Toc3673)

[7.研究感悟 - 11 -](#_Toc15856)

1.研究对象

二叉平衡树的插入，删除以及可视化的实现

2.研究方法

（1）参考二叉排序树的实现过程，构建二叉平衡树，明确二叉平衡树不平衡的状态以及调整为平衡的方法。

（2）插入和删除二叉树采用递归的方法，插入和删除后进行递归判断是否平衡。

（3）二叉平衡树的可视化采用html和JS语言来实现，具有简便性，可视性较强。

3.研究内容

3.1基本数据结构的确立

根据平衡二叉树的特点，我们除了像其他的树一样声明节点的左孩子和右孩子外，因为二叉平衡树有计算每个节点不平衡因子的需要，因此我们还需声明父节点，同时更新树高。

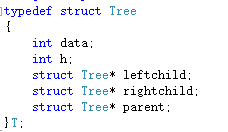


图1 节点的结构体

3.2二叉平衡树的构建

build函数递归构建二叉平衡树

根据二叉平衡树的特点为左孩子小于根节点，右孩子大于根节点，因此当插入的节点的值比根节点小时，递归左孩子，反之，右孩子。

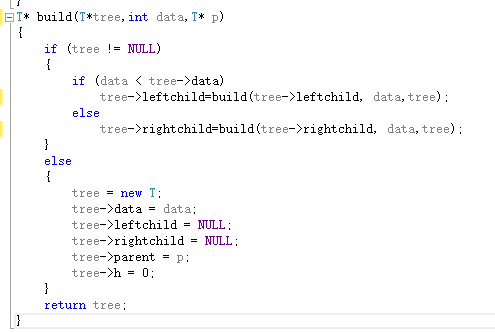


图2 build函数构建二叉平衡树

3.3二叉平衡树的四种不平衡类型

3.3.1 LL型（左孩子的左子树上添加）

**特点：**当前节点为不平衡的节点，并且当前节点的左孩子的深度比右孩子的深度大2，并且左孩子的左节点的深度大于其右孩子的深度，此时需要直接向右旋转。

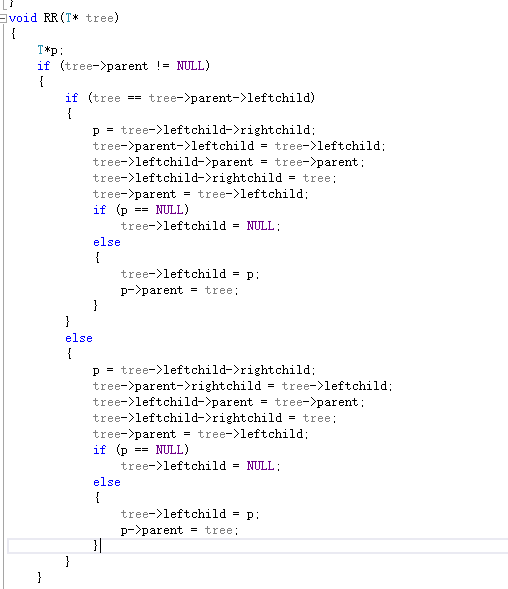
**方法：**

（1）判断是否为头节点

（2）若为头节点，将它的左孩子赋为头节点。

（3）判断其左孩子的右节点是否为空，保存其左孩子的右节点，将其作为当前不平衡节点的左孩子，完成重新调平衡的工作。

（4）若不为头节点，判断为父节点的左孩子还是右孩子，若为父节点的左孩子，则把父节点的左孩子换成当前节点的左孩子，并执行步骤3，否则，把父节点的右孩子换成当前节点的左孩子，并执行步骤3。



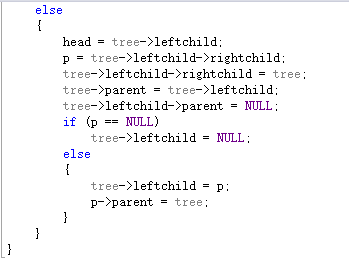


图3 右旋代码部分

3.3.2 RR型（右孩子的右子树上添加）

**特点：**当前节点为不平衡的节点，并且当前节点的右孩子的深度比左孩子的深度大2，并且右孩子的右节点的深度大于其左孩子的深度，此时需要直接向左旋转。

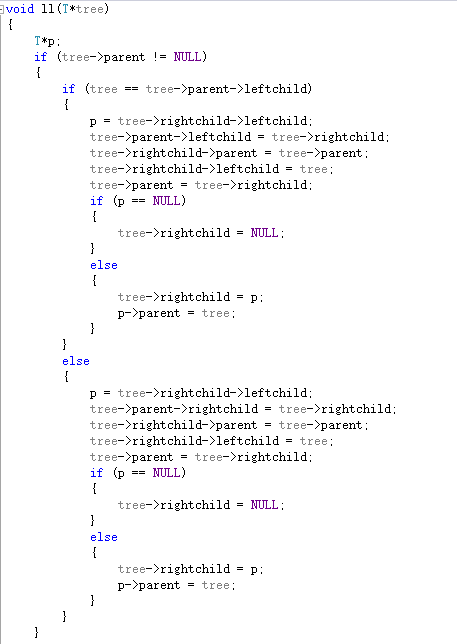
**方法：**

（1）判断是否为头节点

（2）若为头节点，将它的右孩子赋为头节点。

（3）判断其右孩子的左节点是否为空，保存其右孩子的左节点，将其作为当前不平衡节点的右孩子，完成重新调平衡的工作。

（4）若不为头节点，判断为父节点的左孩子还是右孩子，若为父节点的左孩子，则把父节点的右孩子换成当前节点的右孩子，并执行步骤3，否则，把父节点的左孩子换成当前节点的右孩子，并执行步骤3。



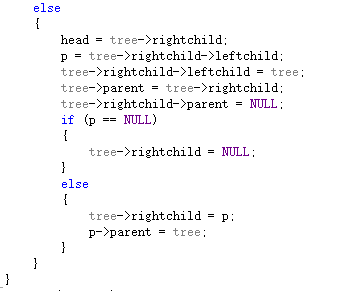


图4 左旋代码

3.3.3 LR型（右子树的左孩子上加节点）

**特点：**当前节点为不平衡的节点，并且当前节点的左孩子的深度比右孩子的深度大2，但是左孩子的右节点的深度却大于其左孩子的深度，此时需要先向左旋转，再向右旋转，左旋转和右旋转的代码可以直接调用前面LL和RR的代码。

**方法：**

当前不平衡节点的右子树左旋后，以当前节点为根节点的树右旋。

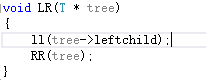


图5 先向左旋后右旋代码

3.3.4 RL型（左子树的右孩子上加节点）

**特点：**当前节点为不平衡的节点，并且当前节点的右孩子的深度比左孩子的深度大2，但是右孩子的左节点的深度却大于其右孩子的深度，此时需要先向右旋转，再向左旋转，左旋转和右旋转的代码可以直接调用前面LL和RR的代码。

**方法：**

当前不平衡节点的左子树右旋后，以当前节点为根节点的树左旋。

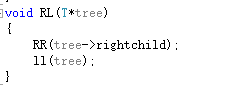


图6 先向右旋后左旋代码

3.4 二叉平衡树的插入

Search函数递归

**思路：**首先把新节点按照二叉排序树的规则插入当前二叉平衡树，然后通过递归算法找到该节点，逐步向上判断是否出现不平衡，属于哪一种不平衡，并根据前面四种类型的判断规则进行调整。



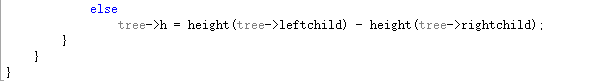


图7 search递归插入

3.5 二叉平衡树的删除

Delete递归删除

**节点删除按照删除节点的类型分为三种情况：**

**1.右子树为空**：

若当前节点的左子树不为空，直接用左孩子来代替当前节点的位置，并删除当前节点，否则，置为空。

**2.左子树为空：**

若当前节点的右子树不为空，直接用右孩子来代替当前节点的位置，并删除当前节点，否则，置为空。

**3.左子树和右子树均不为空：**

此时分为两种情况：

(1)左子树高度大于右子树

用待删除节点的左子树中的最大值来代替当前节点的位置，并把最大值的节点删除。

(2)右子树的高度大于左子树

用待删除节点的右子树中的最小值来代替当前节点的位置，并把最小值的节点删除。

**步骤：**

1. 首先通过递归找到要删除的节点p;
2. 找到要删除的节点p后，判断该节点属于哪种类型，是左子树为空，还是右子树为空，或两者均不为空。
3. 若左子树为空或右子树为空，判断是否有父节点，若父节点为空，直接让头节点为右子树节点或左子树节点，并删除要删除的节点p;若父节点不为空，直接让父节点的左孩子或者右孩子为p的不为空的字数节点即可；若左子树右子树均为空，直接让父节点的左子树和右子树均为空。
4. 若删除节点p的左子树和右子树均不为空，此时需比较一下，左子树和右子树的深度，若左子树深度大于右子树，那么找到左子树的最大节点leftmax代替删除节点p，并删除leftmax，此时再次调用delete递归。同理，若右子树深度大于左子树深度，找到右子树的最小节点rightmin代替删除节点p，并删除rightmin，此时再次调用delete递归。

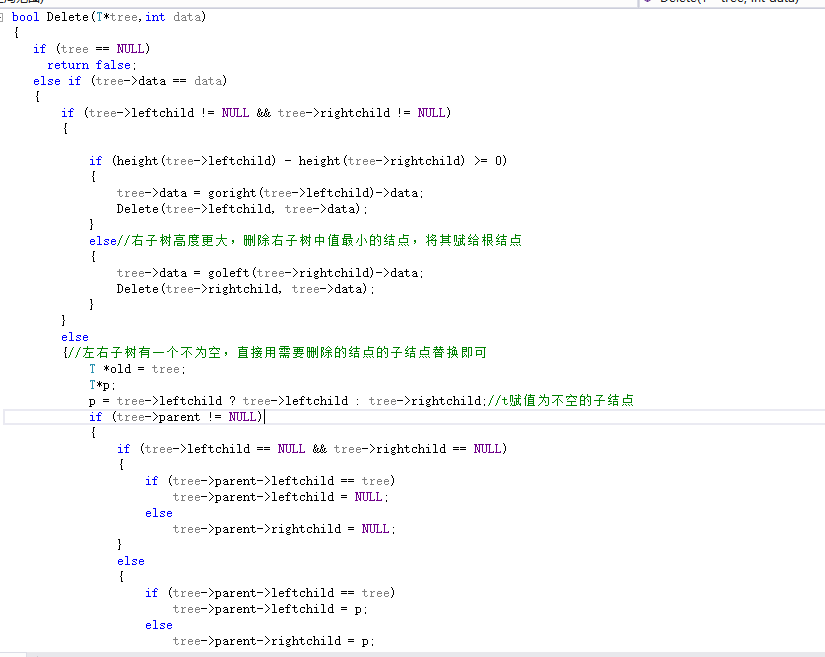






图8 delete递归删除节点

4. 二叉平衡树的可视化

4.1可视化方法：html和JS语言

4.2使用工具：VS2015

4.3界面展示：

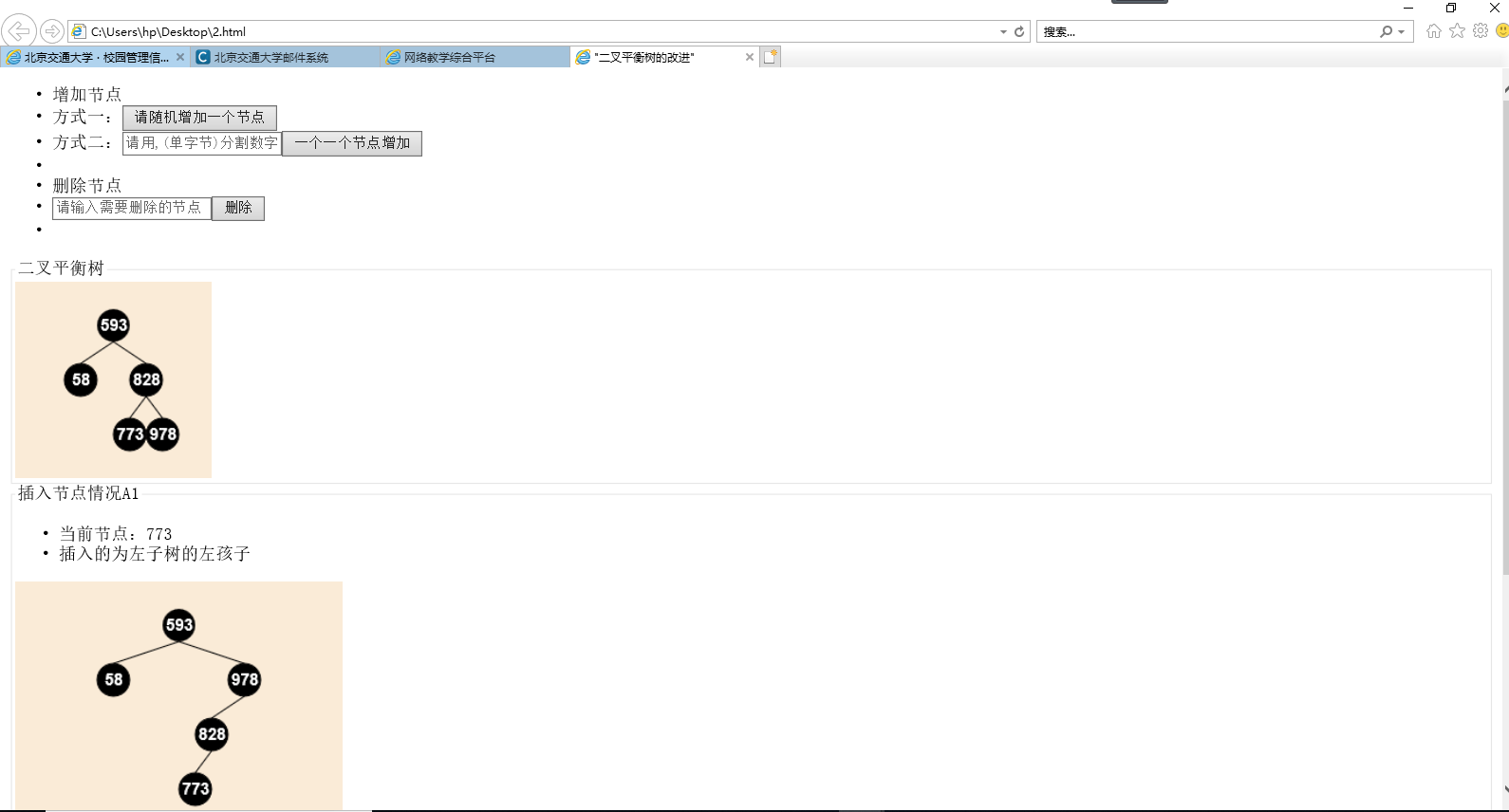


图9 二叉平衡树可视化页面展示1



图10 二叉平衡树可视化界面2

4.4功能介绍：

1. 随机添加节点
2. 通过字符串指定节点
3. 删除指定的节点
4. 添加和删除的方法实时说明，便于理解

5.研究成果呈现

相关代码见附件一

网页链接：2.html

6.研究结果分析和讨论

**优点：**

（1）算法简单易理解，可行性强

（2）网页界面简单清晰，动画展示说明详细，步骤说明容易理解

（3）交互性强

**缺点：**

（1）多处递归，算法效率不高，可用循环改进

（2）Html动画切换比较快

7.研究感悟

通过本次对二叉平衡树的研究，我对课本上的二叉平衡树有了更深的理解，并且自己写代码实现了更深一层的插入和删除等操作，我觉得还是得进行实践操作，因为在具体写代码的过程中会遇到很多想不到的困难，可视化的实现使我对从c到JS的转化，以及与html接口有了更深一步的理解，总体来说这个小项目使我受益很多。

参考文献

【1】平衡二叉树的实现

【2】二叉平衡树的插入，删除

【3】二叉平衡树的可视化