

函数式编程原理

课程报告

**姓 名： 邹雅**

**学 号： U201915035**

**班 级： ACM1901**

**指导教师：** **顾琳**

**计算机科学与技术学院**

**2022 年 4月21日**

一、上机实验心得体会

一棵minheap树定义为：

1. t is Empty;
2. t is a Node(L, x, R), where R, L are minheaps and values(L), value(R) >= x (value(T)函数用于获取树T的根节点的值）

**1.1 编写SwapDown函数**

SwapDown函数给定的参数是一棵树（tree），该函数的作用是将其根节点的数据按照最小堆的定义移到所恰当的位置，返回值就是调整完的tree。

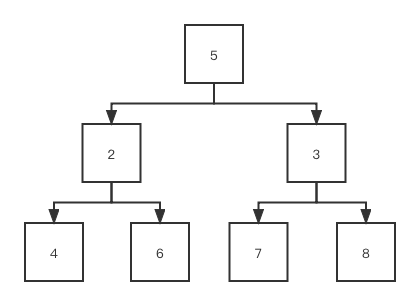


图1.1 tree的示意图

假如传入一棵树如上图所示，那么很明显，我们需要将根节点与其左右子树的根节点（也就是图中的5和2、3）去做比较。假如这里不是5而是1，那我们发现即使不用调整，该树本身就是最小堆。而此时5比2、3都要大，那么我们比较2和3，把更小的那个向上移动成为根节点，而原根节点则直接向下移动。以此类推。

在编写函数式语言的代码时，需要注意模式匹配需要完全，根据需要，得出代码如下所述。

fun SwapDown(Empty : tree) = Empty

| SwapDown(Br(Empty,x,Empty)) = Br(Empty,x,Empty)

| SwapDown(Br(Br(l1,v,r1),x,Empty)) =

if x <= v then Br(Br(l1,v,r1),x,Empty)

else Br(SwapDown(Br(l1,x,r1)),v,Empty)

| SwapDown(Br(Empty ,x, Br(l1,v,r1))) =

if x <= v then Br(Empty ,x, Br(l1,v,r1))

else Br(Empty ,v, SwapDown(Br(l1,x,r1)))

| SwapDown(Br(Br(l1,v1,r1) ,x, Br(l2,v2,r2))) =

if x <= v1 andalso x <= v2 then Br(Br(l1,v1,r1) ,x, Br(l2,v2,r2))

else if v1 <= v2 then Br(SwapDown(Br(l1,x,r1)) ,v1, Br(l2,v2,r2))

else Br(Br(l1,v1,r1) ,v2, SwapDown(Br(l2,x,r2)));

**1.2 编写heapify函数**

heapify函数的功能是将传入的参数tree调整为最小堆输出。

有了上面的SwapDown函数，我们可以发现heapify就是个壳，用处就是把一个个根节点拿出来进行SwapDown操作并记录。一般树的递归的实际操作是递归到空的时候才开始的，因为空就是最小堆或者有序树。这时再SwapDown一个个叶子节点，然后是上层节点，直到栈中递归保存到所有节点都被操作了，就可以得到最后的结果。

heapify函数代码如下所述。

fun heapify(Empty:tree) = Empty

| heapify(Br(l1,x,r1)) = SwapDown(Br(heapify(l1),x,heapify(r1)));

**1.3 运行结果**

直接在命令行中进行SML测试。

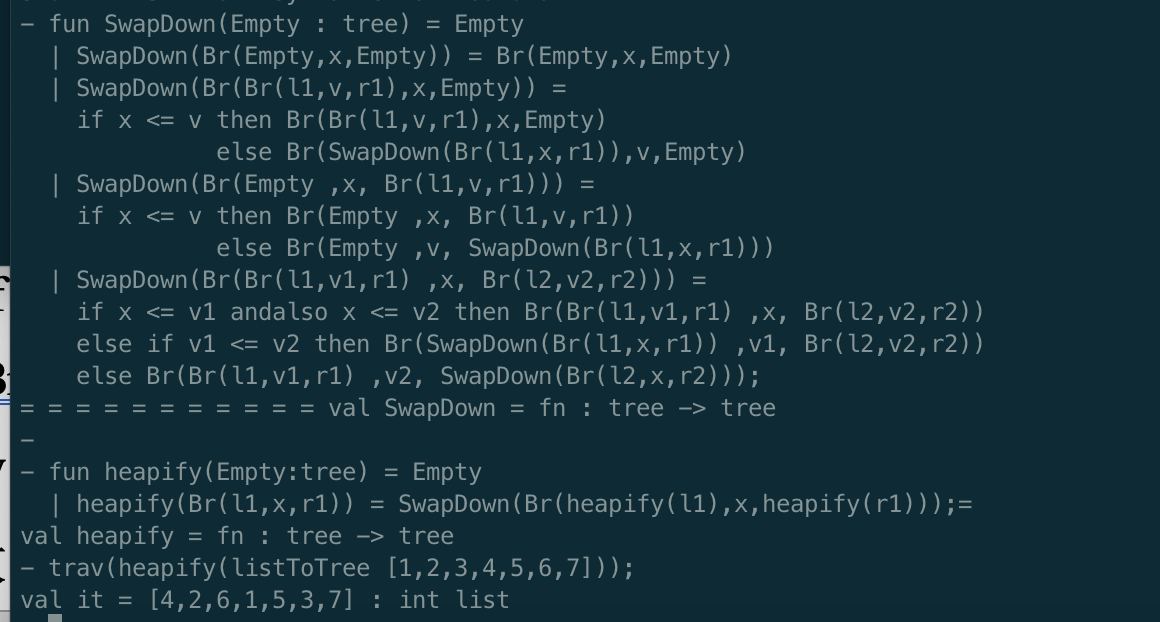


图1.2 运行结果

**1.4 性能分析**

* work分析

我们首先来分析SwapDown函数的work。我们已知树的节点为n个，树的深度则为.

所以SwapDown的work为.

接着其上来分析heapify的work。

根据推导可以得到heapify的work为.

* span分析

同样地，我们来分析SwapDown函数的work。

所以SwapDown的span为.

接着上面的基础来分析heapify的span。

根据推到可以得到heapify的span为.

**1.5 遇到的问题以及解决方式**

一开始写SwapDown函数的时候想用上在实验中提到并完成的treecompare，但是发现遇到最大的问题是无法将左右子树的根节点取出来进行下一步的根节点上移和递归下移原根节点的操作。在这条路上卡了一段时间， 没有想到解决方式，于是直接采用了变量模式匹配。在匹配完整性上又花了一段时间来进行考虑。

此外，本题的性能分析也是比较难的。一开始有一个大概想法，直觉能猜出work和span，但是就是很难理论去推导得到。后来又回归老师上课讲过的内容，模仿PPT上的性能分析列出公式。再一步步推导可以得到最后的复杂度。

**1.6 上机心得体会**

虽然因为疫情影响，没有去到实验室内完成实验，但是自己独立完成实验的过程中，还是能实践了函数式语言的使用。

整体来说，实验的难度适中，可以在不算很长的时间内完成。但是基本是围绕着老师上课的内容进行展开实践，还是很有收获的。特别是对于heapify这道题的完成和后续理解思考，更能让我进一步去考虑函数式编程语言编写方式其中的细节。

二、课程建议和意见

函数式编程是一种编程模型，它将计算机运算看作是数学中函数的运算，并且避免了状态以及变量的概念。函数式编程中任何函数的值只取决于它的参数的值，而与求得参数值的先后或调用函数的路径无关。

在本课程中，我们学习了SML相关的语法，包括表达式和类型、声明和作用域、模式和匹配、ML中的等式、规则说明等内容；我们了解了如何对SML语言进行代码说明，以及程序证明、近似运行时间和递推分析；此外我们还一起学习了一种新的数据结构tree，并用tree类型进行编写程序和相关的并行性能分析；最后我们知道了类型检测、多态的特点、类型推导的方式以及多态的特征。

课程从函数式编程介绍的角度，内容完整，结构清晰。但总体来说编程量偏少，涉及数据结构和算法的部分也比较少。

在头歌实验平台的部署方面，相对来说不是很完善。我在写第一次实验的时候就遇到因为空格和空行始终无法通过实验的问题。建议在实验发布前增加测试或者使用更加完善的部署平台。