# 递归算法分析中主定理的应用

李 卿

(安徽理工大学计算机科学与工程学院,安徽 淮南 232001)

**摘 要**:算法的时间和空间复杂度分析是计算机算法设计的重要内容,递归算法的时间复杂度分析尤为困难。给出了主定理的证明,并讨论了如何利用主定理来分析一类递归算法的时间复杂度,最后给出了主定理实用的范围。

**关键词**:递归;主定理;递归树;分治算法;算法分析

计算机的资源,最重要的是时间和空间资源,算法的时间复杂度是运行算法时所需要的时间资源的度量①。由于计算机资源有限,因此算法的复杂度分析,尤其是时间复杂度分析就显得的非常重要。递归算法是直接或间接调用自身的算法,在算法设计中使用递归,可使算法的描述更加简洁,但同时也增加了其时间复杂度分析的难度。

## 1 主定理的引入

主定理:设 a≥1,b>1 为常数, $f(n)$ 为函数,T(n)为递归式 $T(n)=aT(n/b)+f(n)$ …… 式①

对非负整数定义,其中 n/b 为 $\lfloor n/b \rfloor$ 或 $\lceil n/b \rceil$。则 T(n)可能有如下渐进界:

1.1 某常数 $\varepsilon>0$,有 $f(n)=O(n^{(\log_b a)-\varepsilon})$,则 $T(n)=\Theta(n^{\log_b a})$;

1.2 $f(n)=\Theta(n^{\log_b a})$,则 $T(n)=\Theta(n^{\log_b a}\lg n)$;

1.3 对某常数 $\varepsilon>0$,有 $f(n)=O(n^{(\log_b a)-\varepsilon})$,且对常数 c<1 与所有足够大的 n,有 a $f(n/b)\leqslant$c $f(n)$,则 $T(n)=\Theta(f(n))$。

在主定理中,通过将函数 $f(n)$ 与函数 $n^{\log_b a}$ 进行比较,得出 T(n)的渐进界由其中较大的函数决定。包括了三种情况:

情况一:若 $f(n)=O(n^{(\log_b a)-\varepsilon})$,即 $n^{\log_b a}$ 是 $f(n)$ 的上界,则 $T(n)=\Theta(n^{\log_b a})$;

情况二:若 $f(n)=\Theta(n^{\log_b a})$,即 $f(n)$ 和 $n^{\log_b a}$ 同样大小,则 $T(n)=\Theta(n^{\log_b a}\lg n)$;

情况三:若 $f(n)=\Omega(n^{(\log_b a)+\varepsilon})$,即 $n^{\log_b a}$ 是 $f(n)$ 的下界,则 $T(n)=\Theta(f(n))$。

## 2 主定理的证明

欲证明主定理的正确性,需分两部分进行:首先分析递归式①,将其简化成 $T(n)$ 仅定义在 $b>1$ 的整数幂上,即 $n=1,b,b^2,\ldots$;然后证明扩展到所有正整数 n 都成立。

2.1 三个引理

欲证明递归式①在 $n=b^i$ 正合幂上成立,需引入如下三个引理。

引理 1.设 a≥1,b>1 为常数, $f(n)$ 为定义在 b 的正合幂上的非负函数,定义递归式 $T(n)$ 为:

$$T(n)=\begin{cases}\Theta(1) & \text{若 } n=1\\ aT(n/b)+f(n) & \text{若 } n=b^i\end{cases}$$ ,i 为正整数 …… 式②

则有: $T(n)=\Theta(n^{\log_b a})+\sum_{j=0}^{\log_b n-1} a^j f(n/b^j)$

引理 2. 设 a≥1,b>1 为常数, $f(n)$ 为定义在 b 的正合整数幂上的非负函数,函数 g(n)有如下定义: $g(n)=\sum_{j=0}^{\log_b n-1} a^j f(n/b^j)$,对于 b 的整数幂,该函数可被渐进限界为:

①若对常数 $\varepsilon>0$,有 $f(n)=O(n^{(\log_b a)-\varepsilon})$,则 $g(n)=O(n^{\log_b a})$;

② $f(n)=\Theta(n^{\log_b a})$,则 $g(n)=\Theta(n^{\log_b a}\lg n)$;

③对常数 c<1 及所有 n≥b,有 a $f(n/b)\leqslant$c $f(n)$,则 $g(n)=\Theta(f(n))$。

引理 3.设 a≥1,b>1 为常数,$f(n)$为定义在 b 的整数幂上的非负函数,按式问定义递归式 T(n),则 T(n)可能有如下渐进界:

①若 $f(n)=O(n^{(\log_b a)-\varepsilon})$,$\varepsilon>0$,则 $T(n)=\Theta(n^{\log_b a})$;

②若 $f(n)=\Theta(n^{\log_b a})$,则 $T(n)=\Theta(n^{\log_b a}\lg n)$;

③若 $f(n)=\Omega(n^{(\log_b a)+\varepsilon})$,$\varepsilon>0$,且对常数 c<1 与所有足够大的 n,有 a $f(n/b)\leqslant$c $f(n)$,则 $T(n)=\Theta(f(n))$。

2.2 上取整和下取整时的证明

显然主定理在 n 取 b 的整数幂的范围内成立, 下面将 n 扩展到整个整数范围内,即主递归式中包含上取整函数或下取整函数的形式:

$T(n)=aT(\lceil n/b \rceil)+f(n)$ …… 式③

$T(n)=aT(\lfloor n/b \rfloor)+f(n)$ …… 式④

本文给出含上取整函数式④的证明, 下取整的情况式③类似,读者可自行证明之。

构造式④的递归数,如图 1 所示。其中,

$$n_j=\begin{cases}n & \text{若 } j=0\\ \lceil n_{j-1}/b \rceil & \text{若 } j>0\end{cases}$$

下面确定递归树的深度 k,由于 $\lceil x \rceil \leq x+1$,于是,

$$n_0\leq n, n_1\leq \frac{n}{b}+1, n_2\leq \frac{n}{b^2}+\frac{1}{b}+1, n_3\leq \frac{n}{b^3}+\frac{1}{b^2}+\frac{1}{b}+1,\ldots$$

一般地,有

$$n_j\leq \frac{n}{b^j}+\sum_{i=0}^{j-1}\frac{1}{b^i}<\frac{n}{b^j}+\sum_{i=0}^{\infty}\frac{1}{b^i}<\frac{n}{b^j}+\frac{b}{b-1}$$

当 $j=\lfloor \log_b n \rfloor$ 时,有

$$n_{\lfloor \log_b n \rfloor}<\frac{n}{b^{\lfloor \log_b n \rfloor}}+\frac{b}{b-1}<\frac{n}{b^{\log_b n-1}}+\frac{b}{b-1}=\frac{n}{n/b}+\frac{b}{b-1}=b+\frac{b}{b-1}=O(1)$$

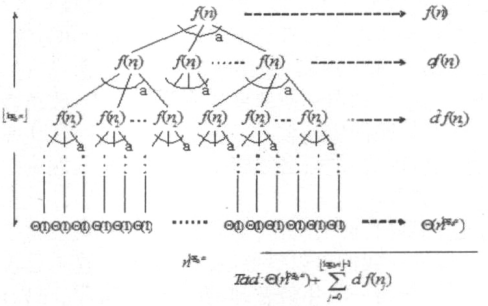

图 1 式 $T(n)=aT(\lfloor n/b \rfloor)+f(n)$ 递归树

由此可知,在深度为 $\lfloor \log_b n \rfloor$ 时,问题大小至多为常数。因此由图可得:

$$T(n)=\Theta(n^{\log_b a})+\sum_{j=0}^{\lfloor \log_b n \rfloor} a^j f(n_j)$$ ……式⑤

接着求和式⑥的值,

$$g(n)=\sum_{j=0}^{\lfloor \log_b n \rfloor} a^j f(n_j)$$ ……式⑥

情况一,证明类似引理 2 的证明略。

情况二,有 $f(n)=\Theta(n^{\log_b a})$,由于 $j\leq \lfloor \log_b n \rfloor$ 隐含 $b^j/n\leq 1$,界 $f(n)=O(n^{\log_b a})$ 隐含存在常数 $c>0$,使对足够大的 $n_j$,有:

$$\begin{aligned}f(n_j)&\leq c\left(\frac{n}{b^j}+\frac{b}{b-1}\right)^{\log_b a}\\ &=c\left(\frac{n^{\log_b a}}{a^j}\right)\left(1+\left(\frac{b^j}{n}\bullet\frac{b}{b-1}\right)\right)^{\log_b a}\\ &\leq c\left(\frac{n^{\log_b a}}{a^j}\right)\left(1+\frac{b}{b-1}\right)^{\log_b a}\\ &=O\left(\frac{n^{\log_b a}}{a^j}\right)\end{aligned}$$

于是情况②得证。

情况三,对 n>b+b/(b−1)有 $af(\lfloor n/b \rfloor)\leq cf(n), c<$为常数,则 $a^j f(n_j)\leq c^j f(n)$。于是

$$g(n)=\sum_{j=0}^{\lfloor \log_b n \rfloor} a^j f(n_j)\leq \sum_{j=0}^{\lfloor \log_b n \rfloor} c^j f(n)\leq f(n)\sum_{j=0}^{\infty} c^j=O(f(n))$$

此处 C 为常量,有 $g(n)=\Theta(f(n))$。至此主定理证明完毕。

## 3 主定理的应用

3.1 二路归并算法的分析

归并法(merge),是将两个或多个有序表合成一 (下转 83 页)

接受能力不一样,有的快,有的慢。如果仅就其学习过程进行评分,显然是不公平的。因此,我们在技能实习即将结束的前半天对学生在实习中的难点、以后工作中的重点,进行操作技能考核,是很有必要的。这一指标能够较准确地反映学生的动手能力,定量地、客观地反映学生的学习效果。这是科学性、公正性的重要指标,应当适当赋予较大权重。这也是中职教育的核心。

对于房产测绘的操作技能考核,应就其简易程度区分对待。房产调查相关内容就比较简单,要求学生能够正确的填写相关调查表,绘制草图等。面积量算是技能中的重点,要重点考核学生对相关规范的理解和应用程度。房产图的测绘是难点,学生实习过程中会差别较大,这也是实习内容的重点。因此,就技能的考核应进行详细的划分。

**3 模糊综合评价法的应用**

从上述分析可以看出,房产测绘成绩评定涉及众多指标,且相互影响相互制约,所以我们提出利用模糊数学的方法对实习成绩进行评定。

3.1 评价指标集合的建立

评价指标集合 V 包括如下评判元素

$$V=\{V_1,V_2,V_3,V_4,V_5\}$$

其中,$V_1$ 为平时成绩,$V_2$ 为理论知识考核成绩,$V_3$ 为房产调查技能考核成绩,$V_4$ 为面积测算技能考核成绩,$V_5$ 为房产图绘制技能考核成绩。

3.2 评价指标的模糊化

以模糊数学的方法对各项指标进行研究见表 1– 表 5.

3.3 评价指标权重的确定

确定评价指标权重的原则是对成绩影响大的元素给以较大权重。根据多年经验分析,将评价指标的权重确定如下:

$$P=(P_1,P_2,P_3,P_4,P_5)=(0.1,0.3,0.1,0.2.0.3)$$

3.4 评价标准的建立

成绩的评价标准见表 6。

3.5 成绩的综合评价

$$U=VP^{-1}=(V_1,V_2,V_3,V_4,V_5)(P_1,P_2,P_3,P_4,P_5)^{-1}$$

3.6 成绩评价实例

以我校 08 级房产测绘班 × × 同学为例,成绩评定如下:

$$U=VP^{-1}=(1,0.8,1,0.8,0.8)(0.1,0.3,0.1,0.2.0.3)^{-1}$$

计算的结果为 0.84,则该生的成绩为良好。

**4 结论**

运用模糊综合评价对房产测绘课程成绩进行评定, 克服了过去的主观、片面性,对其他课程成绩评定也有一定的借鉴意义。但任何一种方法都不是万能的 ,在具体运用时 ,可以考虑将其他方法与该方法综合运用 ,以提高评价的准确性和实用性。

**参考文献**

[1]陈琦,郭大鹏 .模糊综合评价法在干部教育培训评估中的应用[J].实践探索,2009(6).

[2]谌红.模糊数学在国民经济中的应用[M].武汉:华中理工大学出版社,1994.

[3]王庆东.综合评判在教学质量评估中的应用[J].郑州轻工学院学报,2000,(2)1.

[4]吴长悦,申立群,李小光.基于模糊数学的测量实习成绩评定[J].测绘通报,2004(1).

**作者简介**:王建设(1979~),男,河南焦作人,学士,讲师,主要从事工程测量的教学与研究工作。

(上接 193 页)

归并算法的思路是将两个有序表合并成一个有序表。对于拥有 n 个元素的表,其合并表的时间花费为 n,因此,二次归并算法的时间复杂度可使用递归式 $T(n)=2T(n/2)+n$ 表示。

使用主定理求解,$a=2,b=2,f(n)=n$,则 $n^{\log_b a}=n^{\log_2 2}=\Theta(n)$。第二种情况成立,故递归式的解为 $T(n)=\Theta(n^{\log_b a}\lg n)=\Theta(n\lg n)$ 。

3.2 大整数乘法的分析

大整数乘法是分治法的具体应用,分治法的基本思想是将一个规模为 n 的问题分解为 k 个规模较小的子问题,这些子问题相互独立且与原问题相同。设 X 和 Y 是 n 位的二进制整数,为求它们的乘积 XY,将 X 和 Y 分为 2 段,每段长为 n/2 位(假设 n 是 2 的幂),如图 2 所示。


**图 2 大整数 X 和 Y 的分段**

由此,$X=A2^{n/2}+B, Y=C2^{n/2}+D$,X 和 Y 的乘积为:

$$\begin{aligned}XY&=(A2^{n/2}+B)(C2^{n/2}+D)\\&=AC2^{n}+(AD+BC)2^{n/2}+BD\\&=AC2^{n}+((A-B)(D-C)+AC+BD)2^{n/2}+BD\end{aligned}$$

这样只需做 3 次 n/2 位乘法,因此,

$$T(n)=\begin{cases}O(1) & n=1\\ 3T(n/2)+O(n) & n>1\end{cases}$$

使用主定理求解,$a=3,b=2$,则 $n^{\log_b a}=n^{\log_2 3}=\Theta(n^{1.59})$,$f(n)=O(n^{\log_2 3-\varepsilon})$,其中 $\varepsilon=1$。第一种情况成立,故递归式的解为 $T(n)=\Theta(n^{\log_b a})=\Theta(n^{1.59})$。

**4 结论**

本文给出的基于主定理的递归算法分析能够很好的解决基于分治算法的递归问题分析。

**参考文献**

[1]王晓东.算法设计与分析[M].北京:清华大学出版社,2003,19.

[2]Thomas H.Cormen,Charles E.Leiserson,Ronald L. Rivest,Clifford Stein. 潘金贵等译. 算法导论第二版 [M]. 北京: 机械工业出版社,2010,43.

[3]方贤进,潘地林,管建军.用母函数理论分析递归算法的时间复杂度[J].南京师范大学学报,2005,5;1.

**作者简介**:李卿(1987~),女,江西吉安人,安徽理工大学计算机科学与工程学院硕士研究生,计算机应用技术专业。