广西科学院学报
Journal of Guangxi Academy of Sciences
2006,22(4):342～345,351
Vol. 22,No. 4 November 2006

分划递推法在 Hanoi 塔问题上的应用

The Application of PAR Method in Solving the Problem of Tower of Hanoi

孙凌宇[1],冷 明[2]
SUN Ling-yu[1],LENG Ming[2]

(1. 井冈山学院计算机科学系,江西吉安 343009;2. 上海大学计算机学院,上海 200072)
(1. Department of Computer Science, Jinggangshan College,Ji'an,Jiangxi, 343009,China;
2. School of Computer Engineering and Science,Shanghai University,Shanghai,200072,China)

摘要:采用分划递推法通过功能归约变换,形式化推导和证明 Hanoi 塔问题中圆盘的移动规律,从而推导出结构清晰、可读性好、效率高、占用存储空间与圆盘个数无关的非递归算法,算法比较分析地显示出形式化推导在获得高效和正确性的算法程序中的作用. 相关算法在 UNIX 平台下用 C 语言进行实现.

关键词:分划递推法 Hanoi 塔 归约 变换 形式化推导 算法
中图法分类号:TP311 **文献标识码**:A **文章编号**:1002-7378(2006)04-0342-04

Abstract: The partition-and-recur method is used to derivate and prove the law of disk movement in the problem of Tower of Hanoi by transform of function specification and formalizing deduction. A non-recursive algorithm in which the occupied memory is unrelated with the number of disks is developed. The algorithm is readable, efficient and has clear structure. The comparison of algorithms shows the effect of the formalizing deduction in obtaining efficient and correct algorithm program. The related algorithm is implemented in UNIX platform with C language.

Key words: partition-and-recur method, Hanoi tower, specification, transformation, formalizing deduction, algorithm

Hanoi 塔问题是组合数学问题一个典型例子,对它很自然地采用递归程序进行求解. 但是,从软件工程及软件可靠性上看,递归程序相比非递归程序是一种容易出错的结构;从程序运行的效率上看,由于递归程序必须使用堆栈技术和穷举策略,所以递归程序要比非递归算法程序耗费更多的时间和空间. 本文将采用分划递推法,形式化地找出 Hanoi 塔问题中圆盘的移动规律[1],并且借助二进制表达形式将问题的递归求解转换为问题的非递归求解,从而推导出结构清晰、可读性好、占用存储空间与圆盘个数无关的非递归算法.

收稿日期:2005-11-30
修回日期:2006-04-17
作者简介:孙凌宇(1976-),女,江西永丰人,实验师,硕士,主要从事形式化方法和软件自动化研究。

1 相关介绍

1.1 分划递推法简介

薛锦云[2,3]通过研究算法程序设计中的固有规律和科学原理,把软件开发中尽可能多的创造性劳动转化为非创造性劳动,提出了一种简单实用的设计和证明算法的形式化方法——PAR(Partition-and-Recur). 在用 PAR 方法进行程序开发的实践中,可以看出它在形式化推导中的关键思想是功能归约变换[2,3].

PAR 方法使用的主要技术是分划、递推和量词变换. 分划是处理复杂对象常用的方法,典型地应用于分而治之等算法设计方法中;递推主要用于算法分析和动态规划法;量词变换在初等和高等数学中均有应用. PAR 方法的主要贡献在于把这些用于典型问题求解的普通技术结合在一起,经过扩充和改

造,揭示了算法程序设计的本质特征,形成了一种简单实用而又统一的算法程序设计和形式化技术[4,5].

1.2 Hanoi 塔组合数学问题

约在 19 世纪末,欧洲出现一种称为 Hanoi 塔的游戏. 据说这种游戏最早来源于布玛神庙里的教士. 游戏的装置是一块铜板,上面有三个金钢石的塔座 A、B、C,开始时在塔座 A 上有一叠共 n 个圆盘,这 n 个圆盘按从小到大编号为 $1,2,\cdots,n$, 且自下而上,由大到小地叠在一起. 游戏的目标是将塔座 A 上的这叠圆盘移到塔座 C 上,并仍按同样顺序叠置. 规定每次只能移动一个圆盘,任何时刻都不允许将较大的圆盘压在较小的圆盘之上[6].

Hanoi 塔问题是求解对于任何一次圆盘移动步 x, 计算出需移动的圆盘号,而且能计算出它的源塔和目的塔.

2 分划递推法求解 Hanoi 塔组合数学问题步骤

2.1 构造问题的形式规范

PQ:令 n 个圆盘按从小到大编号为 $1,2,\cdots,n$. 令 T 表示 3 个塔座的集合,$s\in T$ 且 $d\in T$ 且 $s\neq d$, $T-s-d$ 表示除 s,d 之外的第三个塔座(即 $s,d,T-s-d$ 分别对应于原问题中的 A 塔、B 塔和 C 塔).

定义:令 $P(i,j,s,d)$ 表示完成将编号从 i 到 j 的圆盘从塔座 s 到 d 的任务,其中 $1\leqslant i\leqslant j\leqslant n$. 令 $Q(n)=Q(i,j:1\leqslant i\leqslant j\leqslant n:P(i,j,s,d))$ 表示完成将编号从 1 到 n 的圆盘从塔座 s 到 d 的任务所包含的一系列移动圆盘操作. 令 $m(k,s,d)=Q(i,j:k\leqslant i\leqslant j\leqslant k:P(i,j,s,d))=k/s/d$ 表示完成将编号为 k 的圆盘从塔座 s 到 d 的移动圆盘操作. 令 $m(k,h)$ 表示完成将编号为 k 圆盘的第 h 次的移动圆盘操作. 令 $m(x)$ 表示在 $Q(n)=Q(i,j:1\leqslant i\leqslant j\leqslant n:P(i,j,s,d))$ 的第 x 次移动圆盘操作.

PR:求 $Hanoi_n=Q(n)=Q(i,j:1\leqslant i\leqslant j\leqslant n:P(i,j,s,d))$ 所有的移动圆盘操作序列,即 $m(x)$ 在 $Q(n)$ 中编号为 k 的圆盘从塔座 s 到 d 的移动圆盘操作.

2.2 分划原问题

对后置断言分划原问题 $Q(n)$ 为要把编号从 1 至 n 号圆盘按要求从 s 塔移到 d 塔,它等价于先把编号从 1 号至 $n-1$ 号圆盘按要求从 s 塔移到 $T-s-d$ 塔,再把编号为 n 号圆盘从 s 塔移到 d 塔,最后把编号从 1 号至 $n-1$ 号圆盘按要求从 $T-s-d$ 塔到 d 塔.

2.3 构造递推关系

$$Q(n)=Q(i,j:1\leqslant i\leqslant j\leqslant n:P(i,j,s,d))=\{\text{范围分裂}\}Q(i,j:1\leqslant i\leqslant j\leqslant(n-1):P(i,j,s,T-S-d))+Q(i,j:n\leqslant i\leqslant j\leqslant n:P(i,j,s,d))+Q(i,j:1\leqslant i\leqslant j\leqslant(n-1):P(i,j,T-s-d,d));$$

$$=\{\text{单一范围}\}Q(i,j:1\leqslant i\leqslant j\leqslant(n-1):P(i,j,s,T-s-d))+M(n,s,d)+Q(i,j:1\leqslant i\leqslant j\leqslant(n-1):P(i,j,T-s-d,d)).\quad(\text{递推关系 1})$$

令 $C(n)=C(i,j:1\leqslant i\leqslant j\leqslant n:P(i,j,s,d))$ 表示在 $P(1,n,s,d)$ 所包含的一系列移动圆盘操作进行计数,则:

$$C(n)=C(i,j:1\leqslant i\leqslant j\leqslant n:P(i,j,s,d))=\{\text{范围分裂}\}C(i,j:1\leqslant i\leqslant j\leqslant(n-1):P(i,j,s,T-s-d))+C(i,j:n\leqslant i\leqslant j\leqslant n:P(i,j,s,d))+C(i,j:1\leqslant i\leqslant j\leqslant(n-1):P(i,j,T-s-d,d));$$

$$=\{\text{单一范围}\}C(i,j:1\leqslant i\leqslant j\leqslant(n-1):P(i,j,s,T-s-d))+C(M(n,s,d))+C(i,j:1\leqslant i\leqslant j\leqslant(n-1):P(i,j,T-s-d,d))=2\times C(n-1)+1.\quad(\text{递推关系 2})$$

因为 $C(1)=C(i,j:1\leqslant i\leqslant j\leqslant 1:P(i,j,s,d))=C(M(1,s,d))=1$,所以 $C(n)=2^n-1$.

(规律 1)

令 $D_k(n)=D(i:1\leqslant i\leqslant n:M(k,s,d)\in Q(i))$ 表示编号为 $k(1\leqslant k\leqslant n)$ 的圆盘移动操作进行计数,则:

$$D_k(n)=D(i:1\leqslant i\leqslant n:M(k,s,d)\in Q(i))=\{\text{范围分裂}\}D(i:1\leqslant i\leqslant(n-1):M(k,s,d)\in Q(i))+D(i:n\leqslant i\leqslant n:M(k,s,d)\in P(i,i,s,d))+D(i:1\leqslant i\leqslant(n-1):M(k,s,d)\in Q(i));$$

$$=\{\text{单一范围}\}D(i:1\leqslant i\leqslant(n-1):M(k,s,d)\in Q(i))+D(M(k,s,d)\in M(n,s,d))+D(i:1\leqslant i\leqslant(n-1):M(k,s,d)\in Q(i))=2\times D_k(n-1).\quad(\text{递推关系 3})$$

因为 $D_k(k)=D(i:1\leqslant i\leqslant k:M(k,s,d)\in Q(k))=1$,得 $D_k(n)=2^{n-k}$. (规律 2)

$$C(n)=\sum_{k=1}^{n}D_k(n)=2^0+2^1+2^2+\cdots+2^{(n-1)}=2^n-1,$$

和由递推关系 2 所得结果一致.

$$Q(k+2)=Q(i,j:1\leqslant i\leqslant j\leqslant(k+2):P(i,j,s,d))=\{\text{范围分裂}\}Q(i,j:1\leqslant i\leqslant j\leqslant(k+1):P(i,j,s,T-s-d))+Q(i,j:(k+2)\leqslant i\leqslant j\leqslant(k+2):P(i,j,s,d))+Q(i,j:1\leqslant i\leqslant j\leqslant(k+1):P(i,j,T-s-d,d));$$

$$=\{\text{单一范围}\}Q(i,j:1\leqslant i\leqslant j\leqslant(k+1):P(i,$$

$j,s,T-s-d))+M(k+2,s,d)+Q(i,j:1\leqslant i\leqslant j\leqslant(k+1):P(i,j,T-s-d,d))$;

$=\{$**范围分裂**$\}Q(i,j:1\leqslant i\leqslant j\leqslant k:P(i,j,s,d))+Q(i,j:(k+1)\leqslant i\leqslant j\leqslant(k+1):P(i,j,s,T-s-d))+Q(i,j:1\leqslant i\leqslant j\leqslant k:P(i,j,d,T-s-d))+M(k+2,s,d)+Q(i,j:1\leqslant i\leqslant j\leqslant k:P(i,j,T-s-d,s))+Q(i,j:(k+1)\leqslant i\leqslant j\leqslant(k+1):P(i,j,T-s-d,d))+Q(i,j:1\leqslant i\leqslant j\leqslant k:P(i,j,s,d))$;

$=\{$**单一范围**$\}Q(i,j:1\leqslant i\leqslant j\leqslant k:P(i,j,s,d))+M(k+1,s,T-s-d)+Q(i,j:1\leqslant i\leqslant j\leqslant k:P(i,j,d,T-s-d))+M(k+2,s,d)+Q(i,j:1\leqslant i\leqslant j\leqslant k:P(i,j,T-s-d,s))+M(k+1,T-s-d,d)+Q(i,j:1\leqslant i\leqslant j\leqslant k:P(i,j,s,d))$;

$=\{$**范围分裂和单一范围**$\}Q(i,j:1\leqslant i\leqslant j\leqslant(k-1):P(i,j,s,T-s-d))+M(k,s,d)+Q(i,j:1\leqslant i\leqslant j\leqslant(k-1):P(i,j,T-s-d,d))+M(k+1,s,T-s-d)+Q(i,j:1\leqslant i\leqslant j\leqslant(k-1):P(i,j,d,s))+M(k,d,T-s-d)+Q(i,j:1\leqslant i\leqslant j\leqslant(k-1):P(i,j,s,T-s-d))+M(k+2,s,d)+Q(i,j:1\leqslant i\leqslant j\leqslant(k-1):P(i,j,T-s-d,d))+M(k,T-s-d,s)+Q(i,j:1\leqslant i\leqslant j\leqslant(k-1):P(i,j,d,s))+M(k+1,T-s-d,d)+Q(i,j:1\leqslant i\leqslant j\leqslant(k-1):P(i,j,s,T-s-d))+M(k,s,d)+Q(i,j:1\leqslant i\leqslant j\leqslant(k-1):P(i,j,T-s-d,d))$. (**递推关系** 4)

分析递推关系 4,**得到在完成** $P(1,k+2,s,d)$ **任务中圆盘移动操作呈现如下的规律**:

$$\begin{cases} m(k+2,1)=M(k+2,s,d), \end{cases}$$

$$\begin{cases} m(k+1,1)=M(k+1,s,T-s-d), \\ m(k+1,2)=M(k+1,T-s-d,d), \end{cases}$$

$$\begin{cases} m(k,1)=M(k,s,d), \\ m(k,2)=M(k,d,T-s-d), \\ m(k,3)=M(k,T-s-d,s), \\ m(k,4)=M(k,s,d). \end{cases}$$ (**规律** 3)

因为 $D_k(k)=2^{n-k}$, **令** $1\leqslant h\leqslant 2^{(n-k)}$ **且** $index_k(h)$ **表示** $m(k,h)$ **在所包含的一系列移动圆盘操作中每次编号为** k **的圆盘移动编号. 分析递推关系** 4,**得到计算** $Index_k(h)$ **等式.**

$C(i,j:1\leqslant i\leqslant j\leqslant(k-1):P(i,j,s,T-s-d))=2^{(k-1)}-1$,**则** $Index_k(1)=(2^{(k-1)}-1)+1=1\times 2^{(k-1)}=2^{(k-1)}\times[2\times 1-1]$;

$$\begin{cases} C(i,j:1\leqslant i\leqslant j\leqslant(k-1): \\ \quad P(i,j,T-s-d,d))+C(M(k+1,s,T-s-d))+C(i,j:1\leqslant i\leqslant j\leqslant(k-1):P(i,j,d,s))=(2^{(k-1)}-1)+1+(2^{(k-1)}-1)=2\times 2^{(k-1)}-1, \\ Index_k(2)-Index_k(1)=(2\times 2^{(k-1)}-1)+1=2\times 2^{(k-1)}=2^k \\ \quad \textbf{则}\ Index_k(2)=1\times 2^{(k-1)}+2\times 2^{(k-1)}=3\times 2^{(k-1)}=2^{(k-1)}\times[2\times 2-1]; \end{cases}$$

$$\begin{cases} C(i,j:1\leqslant i\leqslant j\leqslant(k-1): \\ \quad P(i,j,s,T-s-d))+C(M(k+1,s,d))+C(i,j:1\leqslant i\leqslant j\leqslant(k-1):P(i,j,T-s-d,d))=(2^{(k-1)}-1)+1+(2^{(k-1)}-1)=2\times 2^{(k-1)}-1, \\ Index_k(3)-Index_k(2)=(2\times 2^{(k-1)}-1)+1=2\times 2^{(k-1)}=2^k \\ \quad \textbf{则}\ Index_k(3)=3\times 2^{(k-1)}+2\times 2^{(k-1)}=5\times 2^{(k-1)}=2^{(k-1)}\times[2\times 3-1]; \end{cases}$$

$$\begin{cases} C(i,j:1\leqslant i\leqslant j\leqslant(k-1): \\ \quad P(i,j,d,s))+C(M(k+1,T-s-d,d))+C(i,j:1\leqslant i\leqslant j\leqslant(k-1):P(i,j,s,T-s-d))=(2^{(k-1)}-1)+1+(2^{(k-1)}-1)=2\times 2^{(k-1)}-1, \\ Index_k(4)-Index_k(3)=(2\times 2^{(k-1)}-1)+1=2\times 2^{(k-1)}=2^k \\ \quad \textbf{则}\ Index_k(4)=5\times 2^{(k-1)}+2\times 2^{(k-1)}=7\times 2^{(k-1)}=2^{(k-1)}\times[2\times 4-1]; \end{cases}$$

……

$Index_k(h)-Index_k(h-1)=(2\times 2^{(k-1)}-1)+1=2\times 2^{(k-1)}=2^k$,**则** $Index_k(h)=2^{(k-1)}\times[2\times h-1],(1\leqslant h\leqslant 2^{(n-k)})$. (**规律** 4)

综合以上递推关系和分析结果,可以得到如下圆盘移动规律.

规律 1:$C(n)$ **表示** $P(1,n,s,d)$ **所包含移动圆盘移动操作的总次数为** 2^n-1.

规律 2:$D_k(n)$ 表示在 $P(1,n,s,d)$ 所包含的一系列移动圆盘操作中编号为 k 的圆盘移动次数为 2^{n-k}.

规律 3:编号为 k 的圆盘下一次被移动时,是从该圆盘前一次被移动的目的塔移到该圆盘前一次被移动中没有使用过的塔上.编号为 k 的圆盘第一次被移动是在总移动中的第 2^{k-1} 次,并且当 $n-k$ 为偶数时,它是从 A 塔移到 C 塔;当 $n-k$ 为奇数时,它是从 A 塔移到 B 塔.

规律 4:$Index_k(h)$ 表示 $m(k,h)$ 在所包含的一系列移动圆盘操作中每次编号为 k 的圆盘移动编号为 $2^{(k-1)}\times[2\times h-1]$.编号为 k 的圆盘前后两次被移动时,在总的移动次数中相差 2^k 次.

根据上述 4 条规律求解 $m(x)$ 在 $Q(n)$ 中编号为 k 的圆盘从塔座 s 到 d 移动圆盘操作 $(k/s/d)$.首先由规律 1 可知 $1\leqslant x\leqslant 2^n-1$.将其展开有:$x=\sum_{i=0}^{n-1}(a_i\times 2^i)=(a_{n-1}\times 2^{n-1})+(a_{n-2}\times 2^{n-2})+\cdots+(a_k\times 2^k)+(a_{k-1}\times 2^{k-1})+\cdots+(a_1\times 2^1)+(a_0\times 2^0)$,而 $Index_k(h)=2^{(k-1)}\times[2\times h-1]=(a_{n-1}\times 2^{n-1})+(a_{n-2}\times 2^{n-2}+\cdots+(a_{k-1}\times 2^{k-1})+\sum_{i=0}^{k-2}(0\times 2^i)=((a_{n-1}\times 2^{n-k})+\cdots+(a_k\times 2^1)+(a_{k-1}\times 2^0))\times 2^{k-1}$.

由规律 4 可得:$h=[(a_{n-1}\times 2^{n-k})+\cdots+(a_k\times 2^1)+(a_{k-1}\times 2^0)+1]/2$,其中 $1\leqslant h\leqslant 2^{(n-k)}\wedge a_{k-1}=1\wedge\forall(j:0\leqslant j\leqslant(k-2);a_j=0)$.因为 $a_{k-1}=1$,所以该 h 表达式中的分母肯定可以被 2 整除,从而结果肯定是整数,将 $a_{k-1}=1$ 代入表达式中得到以下结果.

$$h=[(a_{n-1}\times 2^{n-k})+\cdots+(a_k\times 2^1)+(1\times 2^0)+1]/2=[(a_{n-1}\times 2^{(n-k)})+\cdots+(a_k\times 2^1)]/2+1=\sum_{j=k}^{n-1}(a_j\times 2^{(j-k)})+1.$$

$m(x)=m(k,s,d)=k/s/d$ 的圆盘号 $k=\max(b:1\leqslant b\leqslant n;x \bmod 2^{(b-1)}=0)$,即 k 为 x 二进制表达形式中从左到右的第一个值为 1 的二进制数位置.由此,当已知 x 时可以求出对应的圆盘号 k,并严格符合规律 4 即编号为 k 的圆盘前后两次被移动时,在总的移动次数中相差 2^k 次.当 x、k 均已知时,$m(x)=m(k,h)$ 算出是第 h 次移动 k 号圆盘,h 表达式中的 $\sum_{j=k}^{n-1}(a_j\times 2^{(j-k)})$ 值为 x 整除 2^k 之后的余数,则 $h=\frac{x}{2^k}+1$,并且因为 $1\leqslant x\leqslant 2^n-1$,得到 $\max(h)=[\frac{2^n-1}{2^k}+1]=2^{(n-k)}$,严格符合规律 2 中描述的在 $P(1,n,s,d)$ 所包含的一系列移动圆盘操作中编号为 k 的圆盘移动次数为 2^{n-k}.

规律 3 中描述编号为 k 的圆盘第一次被移动是在总移动中的第 2^{k-1} 次,并且当 $n-k$ 为偶数时,它是从 A 塔移到 C 塔;当 $n-k$ 为奇数时,它是从 A 塔移到 B 塔.

$$m(k,1)=k/s/d=\begin{cases}s=A\wedge d=B,\text{if}(n-k)\bmod 2\neq 0\\ s=A\wedge d=C,\text{if}(n-k)\bmod 2=0\end{cases}.$$

规律 3 中描述编号为 k 的圆盘下一次被移动时,是从该圆盘前一次被移动的目的塔移到该圆盘前一次被移动中没有使用过的塔上.

$$m(k,h)=\begin{cases}k/s/d,if\ h \bmod 3=1,\\ k/d/(T-s-d),\text{if } h \bmod 3=2,\\ k/(T-s-d)/s,\text{if } h \bmod 3=0.\end{cases}$$

2.4 构造 Hanoi 塔问题非递归算法

```
program hanoi;
var k,n,x,h:integer;s,d,m:char;
begin
1.  k:=1;
2.  s:='A';
3.  writeln("求 hanoi 问题——要求 hanoi 问题规模即圆盘数?");
4.  read(n);
5.  writeln("求 hanoi 问题——要求 hanoi 问题操作步?");
6.  read(x);
7.  do x mod 2 =0
8.    x:=x/2;
9.    k:=k+1;
10. od;
11. if(((n-k) mod 2)==0) →d:='C';m:='B';
12. [](((n-k) mod 2)<>0) →d:='B';m:='C';
13. fi;
14. h:=(x+1)/2;
15. if(h mod 3==0) →writeln(k,m,s);
16. [](h mod 3==1) →writeln(k,s,d);
17. [](h mod 3==2) →writeln(k,d,m);
18. fi;
    end.
```

(下转第 351 页)

表1 *MSE* 和 *ISNR* 评价数据

插值方法	直接复原		插值复原	
	MSE	*ISNR*	*MSE*	*ISNR*
双线性插值	0.0126	1.1111	0.0129	0.9890
双三次插值	0.0127	0.6803	0.0126	0.7067

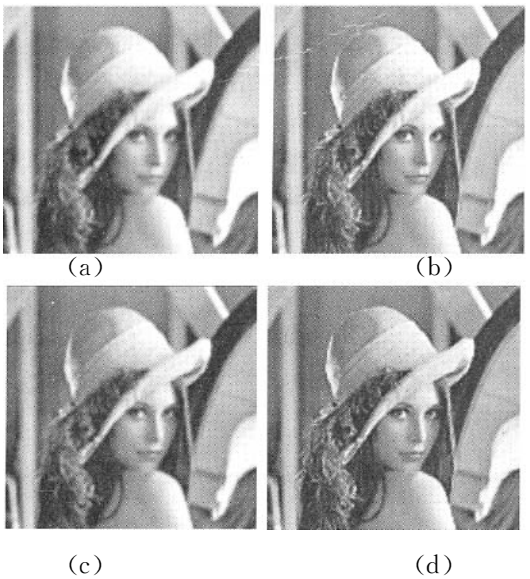

图4 插值和复原图像

(a)双线性插值 g1(256×256);(b)g1的复原图像(256×256);(c)双三次插值 g2(256×256);(d)g_2的复原图像(256×256)

3 结论

本文采用直接复原方法和插值复原方法对单幅欠采样低分辨率模糊图像进行复原实验。直接方法是直接采用 PML 法进行图像复原,插值复原方法是先对要复原的图像进插值预处理,提高图像采样分辨率,然后再采用 PML 法进行图像复原。实验结果显示,两种方法对欠采样模糊图像的复原,效果相同,主观视觉和评价数据十分接近。这表明,通常用于单画幅欠采样模糊图像复原预处理的线性插值没有起到提高复原效果的作用。

参考文献:

[1] 苏秉华. 超分辨力图像复原方法研究[D]. 北京:北京理工大学博士论文,2002.

[2] CASTLEMAN K R. 数字图像处理[M]. 北京:电子工业出版社,2002.

[3] 江铭炎,李兴江,袁东风. 图像插值放大处理的方法[J]. 山东大学学报:理学版,2003,38(3):79-81.

[4] LUCY L B. An iterative technique for the rectification of observed distributions [J]. The Astronomical Journal,1974,9(6):745-765.

(责任编辑:邓大玉 凌汉恩)

(上接第345页)

3 算法比较分析

尽管递归过程结构清晰和可读性强,但根据圆盘移动规律可以推断出递归求解算法时间复杂度为 $O(2^n)$,同时因为递归程序借助堆栈实现参数保存,参数传递等工作导致递归求解算法空间复杂度较高。例如当求解该问题的圆盘数 n 达到64,至少需要移动 $2^{64}-1=1.8\times10^{19}$ 次,若用现代电子计算机1微秒移动一次,运行时间大致为100万年,需占用最大递归层次为64的存储空间。而我们推导出的非递归算法第8步和第9步最坏情况下执行了 n 次,从而对应的时间复杂度为 $O(n)$。因为该非递归算法仅仅使用变量寄存结果信息,并且与问题规模 n 无关,从而占用存储空间少。本文中的非递归算法和递归算法在 UNIX 平台下用 C 语言全部实现。在一定问题规模下非递归算法和递归算法的运行结果一致,反映出了非递归算法的正确性,而且在对两个算法运行时间和消耗内存评估中,实验数据反映出非递归算法的效率高、占用存储空间少的特点。

4 结束语

本文采用分划递推法通过功能归约变换,形式化推导和证明 Hanoi 塔问题中圆盘的移动规律,从而推导出占用存储空间与圆盘个数无关的非递归算法,不仅效率高、而且占用存储空间少,同时也保证了该算法的正确性,显示了形式化推导在获得高效和正确性的算法程序中的作用。

参考文献:

[1] 宁爱兵,黄明和. Hanoi 塔问题非递归算法的形式推导[J]. 计算机工程与科学,2003,25(3):66-68.

[2] XUE JINYUN. A unified approach for developing efficient algorithmic programs[J]. Journal of computer Science and Technology,1997,12(4):314-329

[3] XUE JINYUN. A Practicable approach for formal development of algorithmic programs: proceedings of the International Symposium on Future software Technology (ISFST'99)[C]. Published by Software Engineers Associations of Japan,1999.

[4] 薛锦云. 算法程序形式化开发研究[J]. 云南大学学报,1997,19(增刊):283-288.

[5] 李云清. 分划递推法及其应用[J]. 计算机工程与应用,2001,17:77-79.

[6] 卢开澄. 组合数学[M]. 第3版. 北京:清华大学出版社,2002.

(责任编辑:韦廷宗)