第16卷 第9期
2006年9月

计算机技术与发展
COMPUTER TECHNOLOGY AND DEVELOPMENT

Vol.16 No.9
Sep. 2006

基于分治策略求解方程根的个数

钱 萌,程玉胜,程树林,叶 敏,汪智华,齐 乐
(安庆师范学院 计算机与信息学院,安徽 安庆 246011)

摘 要:n 元高次方程根的个数求解常因问题的规模过大而使通常的算法时间复杂度过高。主要介绍了基于分治策略的二分思想来降低该问题的时间复杂度,并利用哈希技术和线性冲突解决方法进一步提高求解 n 元高次方程根个数的算法效率。

关键词:分治策略;哈希函数;二分法;查找
中图分类号:TP301.6 **文献标识码**:A **文章编号**:1673-629X(2006)09-0041-03

Seeking for Roots of Equation Based on Strategy of Divide and Conquer

QIAN Meng, CHENG Yu-sheng, CHENG Shu-lin, YE Min, WANG Zhi-hua, QI Le
(School of Computer & Information, Anqing Teachers College, Anqing 246011, China)

Abstract: The seeking for roots of n - members hyper - polynomial equation has large time complexity due to the size of n. In this paper, the efficiency of algorithm is improved by operating of binary based on the strategy of divide and conquer as well as using the Hash table and the linear method to resolve the conflicts.

Key words: strategy of divide and conquer; Hash function; binary operation; search

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,这些子问题互相独立且与原问题相同。二分思想是运用分治策略的一个典型例子,主要应用有大整数乘法、Strassen 矩阵乘法、棋盘覆盖等[1]。文中主要基于分治策略,利用二分思想,讨论了一种 n 元高次方程根的个数求解问题。

1 问题的提出

已知一个 n 元高次方程[2,3]:

$$k_1x_1^{p_1} + k_2x_2^{p_2} + \cdots + k_nx_n^{p_n} = 0$$

式中,$x_1, x_2, \cdots, x_n$ 是未知数,$k_1, k_2, \cdots, k_n$ 是系数,$p_1, p_2, \cdots, p_n$ 是指数,取值为正数;且方程中的所有数均为整数;假设未知数 $1 \leqslant x_i \leqslant M(i = 1,2,\cdots,n)$,求这个方程整数解的个数(方程整数解的个数小于 2^{31})。其中约束条件为

$$1 \leqslant n \leqslant 6; 1 \leqslant M \leqslant 150; |k_1M^{p_1}| + |k_2M^{p_2}| + \cdots + |k_nM^{p_n}| < 2^{31}$$

2 问题的时间复杂度分析

从方程的已知条件可知,此方程在最坏的情况下有 6 个未知数,且每个未知数的系数和指数由键盘输入决定,从而无法直接求出方程整数解的个数。由问题给定条件可知每个未知数 x_i 的取值范围最大为 M。如果采用简单的枚举方法来实现,那么它的时间复杂度是 $O(M^6)$,当 M 很大时无法承受。为了减小时间复杂度,必须采用缩小枚举范围的方法。基本思想是:将方程式左边分成代数式 A, B 两个部分,问题转化为计算 $A + B = 0$。A 式含前面几个未知数,B 式含后面几个未知数;然后分别枚举 A, B 式的取值,先计算出 A 式所有可能结果,存入 S_ data 数组中,然后再枚举计算出 B 式所有的取值并通过查找操作计算 $A + B = 0$ 是否成立。如果成立则表示求出方程的一组解。

下面考虑方程式等价划分问题,使得求解问题的时间复杂度尽可能小。假设 n 取值为 6,因此方程含有 6 个未知数。如果令 A 式只含一个未知数,B 式则含有 5 个未知数,那么此时枚举出 A 所有值的时间复杂度为 $O(M)$,枚举 B 式所有值的时间复杂度为 $O(M^5)$,因此该方法时间复杂度为 $O(M^5)$,降低幅度不是很明显。但能启发我们去大胆思考:如果将方程按未知数的个数分成大致相等的两半,结果会怎样呢?同前假设,使 A, B 分别含 3 个未知数,则枚举 A, B 所有值的时间复杂度分别是 $O(M^3)$,那么该方法的时间复杂度为 $O(M^3)$,从而进一步降低了问题的时间复杂度。

假设未知数个数为 n,未知数的值不超过 M,$F(n)$ 表示二分法后解的最大规模。下面进一步计算二分法求解该问题的时间复杂度。

该问题的时间复杂度主要集中在利用哈希技术存储

收稿日期:2006-01-06
基金项目:安徽省教育厅自然科学研究计划资助项目(2004kj265)
作者简介:钱 萌(1963-),男,安徽安庆人,副院长,副教授,研究方向为计算机仿真研究。

左表达式所有可能取值到 $M^{F(n)}$ 数组对应位置,对应时间复杂度为 $O(M^{F(n)})$。而右表达式值利用哈希查找方法进行定位,时间复杂度为 $O(M^{F(n)})$。因此总的时间复杂度 $T(n)$ 为

$$O(M^{F(n)}) + O(M^{F(n)}) = 2 \times O(M^{F(n)})$$

而文中采用的是二分法,因此对规模为 n 的问题,有:

$$F(n) = n/2$$

所以总的时间复杂度 $T(n)$ 为:$2 \times O(M^{n/2})$。

通过分析,经过方程式变换技巧[4]便可实现上述方程的求解。考虑问题的复杂性,在实现时采用了哈希技术[1,5]来降低时间复杂度。以下给出方程的求解过程和系统的仿真结果[6]。

3 解答过程

首先由键盘输入 n,然后由二分法求出整数 t,$t = n$ DIV 2 (DIV 为整除);然后令 A 式包含方程的前 t 个未知数,B 式包含方程剩下的未知数。此时 A,B 式所含的未知数变为 $n/2$。由于问题中 $n = 6$,不是很大,因此只需一次划分方程式就能大幅度降低时间复杂度。

求 A 式:$k_1x_1^{p_1} + k_2x_2^{p_2} + \cdots + k_mx_t^{p_t}$ 的可能取值并利用哈希技术来存放 A 式的值 s。考虑到 s 在最坏情况下有 $150^3 = 3\ 375\ 000$ 个不同的取值,故定义长度为 3 375 000 的线性表来构造哈希表,并令哈希数取 3 375 000。并且考虑到相同的 s 值可能出现多次,为了统计次数将线性表的存储结构定义为两个域:一个存放 s 值,另一个统计这个值出现的次数。其存储结构如下所示:

```
Type Sqlist
    S_data As Double '存放哈希值为 x 的 S_data 的值
    Count As Double '记录这个值出现的次数
End Type
```

3.1 程序主要模块结构图及基本函数功能说明

程序主要模块结构如图 1 所示。

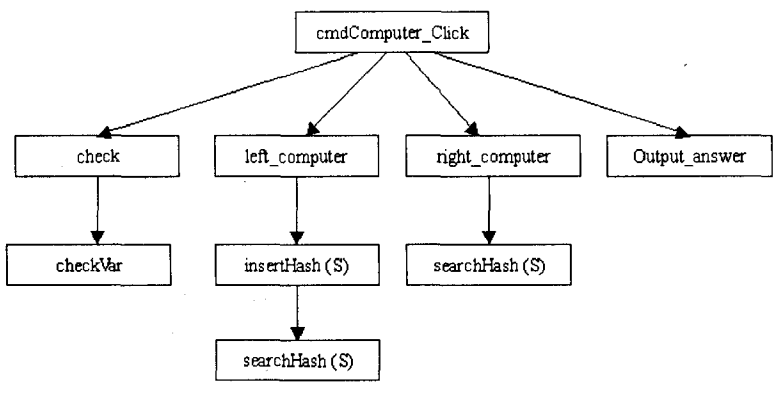

图 1 程序主要模块框架图

(1) 程序启动初始化(form_load):包括对指数、系数和部分控件的初始化。

(2) 开始计算(cmdComputer_Click):当用户数据输入完成后单击"开始计算"按钮触发 Click 事件执行 cmdComputer_Click 过程。

(3) 输入数据检测(check):检测输入数据是否有误或不满足约束条件。

(4) 查找元素(searchHash):该函数计算 B 式的值,然后在 A 式产生的哈希表中查找,查找过程中先查看 Count 域,若为 0 则此单元为空,即该值哈希表中没有,函数返回该单元的索引值。若 Count 不为 0 再将查找元素与 S_data 域中的值比较,如果相等则查找元素存在,然后返回索引值,否则依次探查 $(h(k) + i) \bmod p$,直到查找到与该元素相等或 Count 域为 0 的单元,并返回该单元的索引值。

(5) 插入元素(insertHash):先调用查找元素函数,再根据查找函数返回的索引值检测哈希表中该单元的 Count 域是否为 0,若为 0 说明该元素哈希表中不存在,将该元素赋值给该单元的 S_data 域并将 Count 值加 1,若不为 0 说明该元素在哈希表中已存在,只需将 Count 值加 1。

(6) 计算方程前半部分值(left_computer):该函数中使用一个选择语句判断 n 的值,从而决定 A 式包括几个未知数,枚举代数式 A 的所有值 s 并调用 insertHash 函数,将其插入哈希表中。

(7) 计算方程后半部分值(right_computer):该函数中同样使用一个选择语句判断 n 的值,从而决定 B 式包括几个未知数,然后枚举代数式 B 所有的值,对每次枚举的值都调用 searchHash 函数查找存储在哈希表中的 A 的值,若 $A + B = 0$ 则将方程的解的个数 result 加上哈希表中该单元的 Count 值,依次类推计算出方程解的个数。

(8) 输出函数(Output_answer):输出计算的结果。若方程解的个数$\geqslant 2^{31}$,则弹出提示。

3.2 构造哈希表的基本操作

3.2.1 哈希函数构造

除余法:选择一个适当的正整数 p,由于哈希表的长度为3 375 000,因此可取 $p = 3\ 375\ 000$,令 $h(k) = k \bmod p$。

3.2.2 哈希表初始化

初始化时将哈希表的所有 S_data 和 Count 域使用系统默认值 0,在查找哈希表时先检测 Count 域,若为 0 表示此单元 S_data 域为空。

3.2.3 冲突处理

当不同的 s 值具有相同的哈希值时,便会产生冲突,可采用线性重新散列技术解决冲突。令数组元素个数为 p,则当 $h(k)$ 位置已存储有元素时,依次探查 $(h(k) + i) \bmod p$,$i = 1,2,3\cdots$,直到找到空的存储单元为止(若从 $h(k)$ 开始扫描一圈仍未发现空单元,则表示哈希表已经满了,发生溢出错误)。

3.3 程序流程说明

首先启动程序,执行 form_load 函数,该函数将会对所有变量初始化,包括对运行界面的初始化。当用户输入数据后单击"开始计算"按钮,程序执行 cmdComputer_Click 函数,该函数在执行时先调用 check 函数检测输入数

据是否合法和满足约束条件。若输入的数据不合法或不满足约束条件时,则提示用户检查数据并更正,再单击"开始计算"按钮重新检测计算。若输入的数据合法,则 cmdComputer_Click 开始调用 left_computer 函数计算 A 式的值 s 并调用 insertHash (S)函数将它们插入哈希表,在 insertHash (S)函数中先调用 searchHash (S)查找 s 是否已在哈希表中,若在则不必插入,否则插进哈希表。当 left_computer 执行完毕后,cmdComputer_Click 函数调用 right_computer 函数,该函数先枚举出 B 代数式的所有取值,每计算一个值调用一次 searchHash (S) 函数查找哈希表中是否有使得 $A + B = 0$ 的值,并统计结果。right_computer 函数执行完后,cmdComputer_Click 函数调用 output_answer 函数输出结果。

当 Output_answer 函数执行完毕,此时界面将会出现一个"重新计算"按钮,若单击该按钮,程序将继续按上述步骤执行一次。若用户单击"退出"按钮,程序将结束。

4　计算机仿真结果

例 1　给定 $n = 4, M = 150$, $k_1 = k_3 = 1, k_2 = k_4 = -1, p_i = 2(i = 1,2,3), p_4 = 3$, 该 n 元高次方程解的个数通过计算机仿真后,结果为 5 167,如图 2 所示。

图 2　一组测试数据

例 2　给定 $n = 6, M = 150, k_1 = k_2 = k_3 = 1, k_4 = k_5 = k_6 = -1$, $p_i = 1$, 解的个数超出约束范围($<2^{31}$),如图 3 所示。

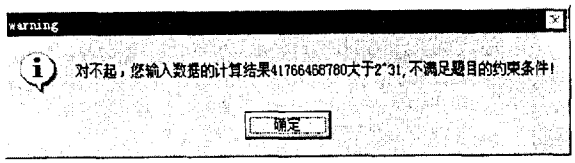

图 3　第三组测试数据

表 1 给出该方程所有可能含对约束条件检测的仿真结果。

表 1　计算机仿真结果

参数组号	N	M	k_1, p_1	k_2, p_2	k_3, p_3	k_4, p_4	k_5, p_5	k_6, p_6	结果	备注
1	3	150	1,2	-1,2	1,2				178	正常
2	4	150	1,2	-1,2	1,2	-1,3			5167	正常
3	6	150	1,1	1,1	1,1	-1,1	-1,1	-1,1		解的个数超范围
4	3	150	1,6	1,1	1,1					各项绝对值和超范围
5	2	150	1,3	-1,1	1,1					系数、指数个数大于 N
6	3	151	1,6	-1,1	1,1					M 的值不符合约束条件
7	2.1	150	1,3	1,2						N 的值不符合约束条件
8	2	8.5	1,2	1,2						M 的值不符合约束条件
9	3	150	2.5,2	1,2	1,2					系数值不符合约束条件
10	3	150	1,2	1,2.5	1,2					指数值不符合约束条件

5　结　论

主要考虑的是如何降低程序执行的时间复杂度,通过二分法减小枚举范围实现了这一目的。在解题过程中将方程分为 A, B 两部分,然后计算 A 的所有值,将它们存入一个数组中,然后再枚举 B 的值,进而在 A 产生的数组中找使 $A + B = 0$ 的数,并统计结果;同时为进一步降低时间复杂度,利用了哈希技术。另外,在程序设计时利用 VB 语言编程,实现了可视化和交互功能。

该文思想使笔者所在学校学生在首届全国计算机仿真大奖赛中荣获第二名。

参考文献:

[1] 王晓东. 计算机算法设计与分析[M]. 北京:电子工业出版社, 2001.

[2] 林尔桢. 方程的解数[EB/OL]. http://www.mydrs.org/program/list.asp? id=261 - NOI2001 - 2001-10-27.

[3] 刘　聃. 浅谈竞赛中哈希表的应用[EB/OL]. http://www.chinaschool.org /aosai/sf/hxb-04.htm-2003-07-31.

[4] 马子彦. 方程迭代求根加速收敛的算法研究[J]. 微机发展, 1996,6(6):28-30.

[5] 夏红霞,钟　珞. 哈希函数在优先队列中的应用[J]. 微机发展,1994,4(3):21-24.

[6] 陈宗海. 系统仿真技术及其应用[M]. 合肥:中国科学技术大学出版社,2001.

(上接第 3 页)

参考文献:

[1] Bier E A, Sloan K R. Two-part texture mappings[J]. IEEE Computer Graphics and Application,1986, 6(9): 40-53.

[2] 范　波,吴慧中. 多面体表面纹理映射方法的研究[J]. 计算机研究与发展, 1999, 36(4): 446-450.

[3] 周　昆,潘志庚. 调和映射的构造及其在图形学中的应用[J]. 中国图像图形学报,1998,3(7):578-582.

[4] 江巨浪,张佑生. 一种应用面积等比约束的半球面纹理映射算法[J]. 系统仿真学报,2004,16(9):1982-1984.

[5] Rogers D F. Procedural Elements for Computer Graphics [M]. Beijing:China Machine Press, 2002.531-534.