北京航空航天大學
BEIHANG UNIVERSITY

第二十一届“冯如杯”创意大赛论文

一类非平衡布尔方程组的摘叶算法研究

项 目 编 号 _____

院（系）名 称 数学与系统科学学院

作 者 姓 名 齐嘉悦（10091204）

杜 昊（10091201）

李 娜（10091203）

王居方（10091202）

指 导 教 师 郭炳晖

2010 年 4 月 2 日

一类非平衡布尔方程组的摘叶算法研究

摘要：

目前，对 NP 问题阈值的数值估计和相变机制的描述已经成为当前理论计算机科学和人工智能领域研究的热点问题之一，针对其中随机约束满足问题的可满足性方向，本文主要研究布尔方程组（布尔方程组是指定义在 Z_2={0,1}上的方程组）的相关可满足性阈值估计问题。针对线性与非线性方程的个数比为 4:1 的非等概率混合类型布尔方程组（MAS 模型），本文基于逐步消去对方程组是否存在解没有影响的变量从而缩减方程组规模、简化问题的摘叶算法思想，实现了相应的摘叶算法。通过对不同参数条件下大量随机实例进行数值实验及求解验证，得到了非等概率混合参数条件下 MAS 模型所对应问题可满足性阈值近似相变曲线，并给出了对应不同参数的阈值下限估计。所得结果不仅给出了 MAS 模型对应问题在非平衡条件下的可满足性阈值估计，而且在其他类型的约束满足方程组求解问题中有着潜在的应用前景。

关键词：

非平衡布尔方程组，摘叶算法，可满足性，阈值估计

Research on the leaf-removal algorithm of a class of unbalanced Boolean equations

Abstract:

Nowadays, numerical estimation to the NP questions and description to phase transition system has become one of the most popular questions in the field of current computer science and artificial intelligence. We chose one part from this which is named random limitation satisfiability to research and we focused on questions related to Boolean equations. Boolean equations referred to equation groups defined on Z_2 field. The paper focused on a certain type of unbalanced equations in which the linear part is three times larger than the nonlinear part. Via the method of leaf-removal algorithm, we obtained the floor estimation of the satisfiability threshold of the original problem by numerical experiments on large scale of instances with different parameter values. Our result not only gives out the threshold estimation of certain type equation groups, but may be widely used in searches for roots of other types of constraint satisfaction problems as well.

Key words:

unbalanced Boolean equations, leaf-removal algorithm, satisfiability, threshold estimation

目录

第一章 引言

近年来，随着对 NP 问题[1]算法实验和理论分析研究的逐渐深入，人们发现对于一大类 NP 完全问题都可以通过定义某个序参数将问题划分为可满足和不可满足两个区域: 可满足区域中几乎所有随机实例都是可满足的，而不可满足区域中几乎所有的随机实例都是不可满足的。在这两个区域之间可满足概率发生突变的现象称为可满足性的相变现象，发生突变的参数区间称为可满足性阈值[2]。

目前，对 NP 问题阈值的数值估计和相变机制的严格描述已经成为当前理论计算机科学和人工智能领域研究的热点问题之一。在数学、物理和计算机科学的交叉领域，对随机约束满足问题可满足性阈值的研究经历了一个逐步由相变现象向相变机制的过程。迄今为止，如何通过算法实验或者理论分析给出一个 NP 问题的可满足性阈值估计仍然是一个受到广泛重视的开放问题。其中，以布尔方程组的求解问题为代表的NP问题满足性阈值研究已经成为这个开放问题中最为活跃的研究领域。

布尔方程组求解属于约束满足问题。随机约束满足问题，是由一族给定取值范围的变量和变量间的随机约束条件组成，对约束满足问题的求解过程就是给每个变量指定一个赋值使得所有约束条件都得到满足的过程[3]。随机约束满足问题是人工智能研究领域的一个重要分支，有着很强的实际应用背景，而求解布尔方程组则是研究随机约束可满足性问题的基础。

针对布尔方程组可满足性相变区域的参数范围确定问题即可满足性阈值问题主要有三种研究方法：传统的算法分析方法[4]；统计物理分析方法[5]；数学的严格上下界估计方法[2]。三种方法侧重点各不相同，一般地，针对问题可满足性阈值的数值估计进行不同方法间的对比，可以得到更加精确的参数区间估计。

本文主要侧重于数学严格下限方法与算法分析方法的综合，以一类非等概率混合线性和非线性布尔方程组的 MAS 模型[6]为研究对象，运用摘叶算法[7]，通过计算渗流核[8]，对不同参数条件下大量随机实例进行求解验证，将计算结果用于研究在非等概率混合参数条件下 MAS 模型所对应具体问题的可满足性阈值，得到了可满足性的近似相变曲线，并给出了满足性阈值的下限估计。

第二章 非线性参数 a=0.2 的 MAS 模型的定义及性质

一个 MAS 模型实例的定义为:考虑由 N 个布尔变量组成的集合 $\{x_1, x_2, ..., x_N\}$ 独立随机地从所有可能给定形式方程中选取 $M=aN$ 个方程组成一个方程组就得到一个随机 MAS 模型实例。其中给定方程的形式为: 类型Ⅰ，线性布尔方程形式 $x_i + x_j + x_k = J_i$，$j, k \in \{0, 1\}$; 或者类型Ⅱ，非线性布尔方程形式 $x_i + x_j \times x_k = 0$，在类型Ⅱ中 x_i 称为线性部分，x_j 和 x_k 称为方程的非线性部分。在实例的生成中，选择两种类型方程的概率分别是 4/5 和 1/5。这里需要特别指出，参数 a 是模型中约束方程与变量个数之间的比值，一般被称为约束方程密度，其决定了 MAS 模型的约束强度。

通过 MAS 模型的构造方式可以看出，一个随机生成的 MAS 模型实例可以被拆分为两个子问题。其中由类型Ⅰ的布尔方程组成的方程组可以看做是约束方程密度为 $4a/5$ 的随机 3-XORSAT 问题，由类型Ⅱ的布尔方程组成的方程组可以看做是约束方程密度为 $a/5$ 的随机 MAS-nonlinear 问题。在一般布尔运算的意义下，第一个子问题的所有解组成的集合对布尔加法构成群，第二个子问题的所有解组成的集合对布尔乘法构成半群。下面我们将在此模型的基础上展开对摘叶算法的讨论。

第三章 摘叶算法

3.1 理论分析

本文研究的是在一般模型定义和特定的参数条件下按照非线性方程概率为 0.2 生成的随机实例，实例中有可能出现满足某些特定消去条件的变量，这些变量的取值对方程组是否有解没有影响。所以，采取摘叶算法，将有效地缩减对象所含变量的规模。

通过将变量表示为节点，变量所满足的约束称为边，摘叶算法被广泛地应用于寻找大规模约束满足问题的渗流核。对于一般的线性布尔方程组求解的问题，摘叶算法的思想是将问题对应的随机图中的叶子节点（度为 1 的叶子节点）和与其连接的边全部删除，得到新图之后重复进行上面的操作，直到得到的图中不含

有叶子节点。由于对于每个含有叶子节点的约束方程，总可以通过调整叶子节点的取值使得该方程有解，去掉这个叶子节点对应变量所在的方程后，方程组是否有解的性质不会发生改变，即原问题的可满足性判定问题化简为一个等价但包含更少变量和约束问题的可满足性判定问题。

通过分析保持该问题可满足性不变的摘叶方式，总结出以下的摘叶算法执行条件[8]：

a）如果一个变量只在唯一一个类型Ⅰ方程中出现，那么删去该方程原问题可满足性不变；

b）如果一个变量只在唯一一个类型Ⅱ方程中的线性部分出现，那么删去该方程原问题可满足性不变；

c）如果某个类型Ⅱ方程中非线性部分的两个变量只在该方程中出现而不在其他方程中出现，那么删去该方程原问题可满足性不变。

对于一个给定的 MAS 模型实例 T，进行两步操作：a）对实例中的变量在方程中出现的位置和频率进行统计；b）对符合上述条件的变量和方程全部删除，这样就得到了在可满足性意义下等价实例 T（1）。接下来再对实例 T（1）进行同样的两步操作，得到在可满足意义下的等价实例 T（2）。如此下去直到某个实例 T（n）中无法找出符合上述条件的方程则算法停止。

每个方程组所剩下来的方程的个数即为该方程组的渗流核，如果渗流核为 0，即所有的方程对该方程组是否有解都无影响，那么这个方程组一定有解。如果渗流核不为 0，方程是否有解还需要更进一步的检验。也就是说，**渗流核为零是方程组实例有解的充分条件**[8]。本文以渗流核为零的方程组在随机实例中所占的比例为统计目标，计算随机实例中方程组有解概率的下限。

3.2 算法实现

该算法思路描述如下[8]：

a）按照模型要求生成 MAS 模型随机实例 $T_i(0)$。

b）循环运行摘叶算法，在第 m 轮中循环检查 $T_i(m)$ 是否存在满足 2.1 中摘叶算法执行条件的方程：如果存在，执行处理程序 LF procedure，得到等价的 $T_i(m+1)$；否则跳出循环体。

c）如果此时等价的$T_i(m)$为空，则记为 1，实例有解；若此时等价的实例$T_i(m)$不为空，记为 0。返回最终$T_i(m)$为空的个数 n。

具体的算法描述：

```
LF procedure
Llag=0; i=0;
while (Flag==0)
   switch T(i) and mark each leaf variable;
   if (the set of equations satisfying the leaf-removal condition A(i) is not empty)
      delete equations in A(i)；
      i++;
   else
   Flag=1;
end
end
CB procedure
   i=0;n=0;
   if(Ti(m)  is empty)
        ++n;
        i++;
if(Ti(m)  is not empty)
;
i++;
   return i;
end
```

第四章 不等概率 MAS 模型解的情况估计

本文采用摘叶算法对不等概率 MAS 模型实例（线性与非线性方程概率比为 4:1）的解进行分析估计，在变量数为 1000、约束方程密度为 a 的条件下，我们

计算了随机实例中存在解的实例（渗流核为 0 是方程组存在解的充分条件）所占的比例，并研究了不等概率 MAS 模型中存在解的平均概率随着约束方程密度 a 增加的变化曲线。在对 10000 个不同规模的随机 MAS 模型实例运行摘叶算法，并在确定其是否有解的基础上，得到的实例有解的平均概率随着约束方程密度变化的图像，针对三类不同变量规模实例求解的数值结果（如图 1 所示）。

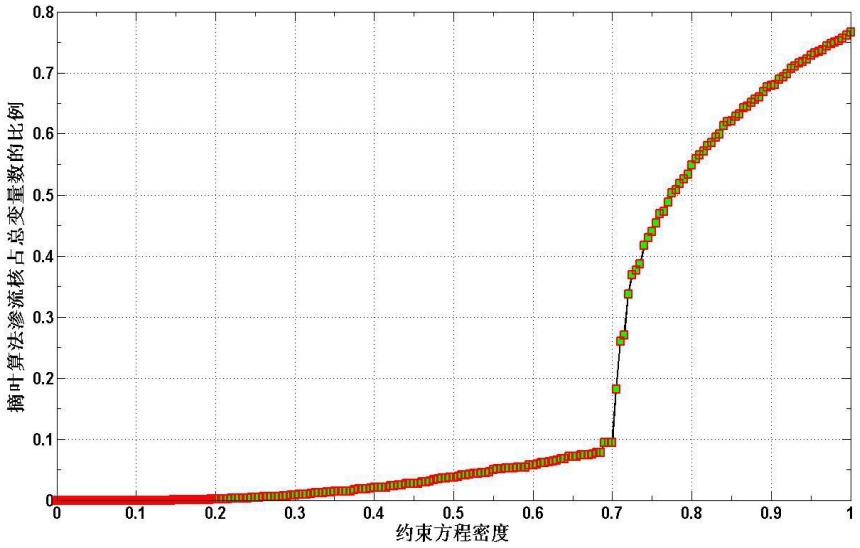

图 1　不同规模随机实例有解之平均概率图像

结束语

本文将算法实验与分析相结合，即通过对作用在给定的参数条件下大量的随机实例进行数据实验，研究了不等概率的混合线性和非线性布尔方程条件下 MAS 模型有解的平均概率随着约束方程密度增加而变化的大体规律，采用摘叶算法，估计了在不同约束方程密度参数条件下大量随机实例中有解的实例所占的比例。在总结了前人的经验基础上，大胆创新，实践了不等概率的 MAS 模型的有解概率的估计，但由于时间及所学的知识有限，利用摘叶算法只能得到 MAS 模型的有解概率的下限，并不能完全计算出有解概率，我们将在以后的学习中更多地了解相关知识，争取有更进一步的研究和创新。

参考文献：

［1］ HARTMANN A K，WEIGT M． Phase transitions in combinatorial optimization problems: basics， algorithms and statistical mechanics［M］.［S．1.］: Wiley-VCH，2005．

［2］ FRIEDGUT E． Sharp thresholds of graph properties， and the k-SAT problem［J］. Journal of the American Mathematical Society，1999，12(4) : 1017-1054．

［3］ 郭炳晖. 随机系统相变的复杂性及动力学研究. 北京：北京航空航天大学, 2005.

［4］ KIRKPATRICK S，SELMAN B． Critical behavior in the satisfiability of random Boolean expressions［J］. Science，1994，264(5163) : 1297-1301．

［5］ MEZARD M，PARISI G，ZECCHINA R． Analytic and algorithmic solutions of random satisfiability problems［J］. Science，2002，297(5582) : 812-815．

［6］ WEI Wei，GUO Bing-hui，ZHENG Zhi-ming． Statistical and algebraic analysis of a family of random Boolean equations［J］. Journal of Statistical Mechanics: Theory and Experiment，2009(2) : 2010．

［7］ MEZARD M，RICCI-TERSENGHI F，ZECCHINA R． Two solutions to diluted p-spin models and XORSAT problems［J］. Journal of Statistical Physics，2003，111(3-4) : 505-533．

［8］ 郭炳晖，韦卫，郑志明. 一类布尔方程组的可满足性阈值研究 [J]. 计算机应用研究，2010, 1001-3695(2010) 10-3666-04.

附录 1：（关于摘叶算法的 MATLAB 源程序）

```
% the program to compute the satisfiability of MAS
pnum=1;
for p=0.2:0.1:0.2
% the ratio of the number of equations part2 non-linear equations
N=10000; % the number of variables
qmaxnum=10; % the number of instances
alphanum=1;
for M=5780:10:5790
   ALPHA(pnum,alphanum)=M/N;
   GM=M;
   interqnum=0;
for qnum=1:1:qmaxnum
   M=GM;
   M2=round(p*M); % judge the number of part2
   M1=M-M2; % the number of part1
   MAT=zeros(M,3);
   % generate the matrix of equations
   for i=1:1:M
        for j=1:1:3
             pos=unidrnd(N);
             if (j>=2)
                  for t=1:1:100
                       if (pos==MAT(i,j-1))
                            pos=unidrnd(N);
                       else
                            break;
                       end
                  end
             end
             MAT(i,j)=pos;
        end
   end
   % SO=MAT;
   % the process of leaf removal algorithm
   markleaf=1;
   while (markleaf>0)
        MARKLIN=zeros(1,N);
        MARKNLIN=zeros(1,N);
        DYNS=zeros(1,N);
        DYNN=zeros(1,N);
        % identify the leaf variable in linear case
        for i=1:1:M1
```

```
    for j=1:1:3
        intermark=MAT(i,j);
        MARKLIN(1,intermark)=MARKLIN(1,intermark)+1;
    end
end
% identify the leaf variables in the linear posotion of part2
for i=(M1+1):1:M
    intermark=MAT(i,1);
    MARKLIN(1,intermark)=MARKLIN(1,intermark)+1;
end
% identify the variables in the non-linear posotion of part2
for i=(M1+1):1:M
    intermark=MAT(i,2);
    MARKNLIN(1,intermark)=MARKNLIN(1,intermark)+1;
    intermark=MAT(i,3);
    MARKNLIN(1,intermark)=MARKNLIN(1,intermark)+1;
end
% verify the real linear leaf
dnums=1;
for t=1:1:N
    if (MARKLIN(t)==1)
        if (MARKNLIN(t)==0)
            DYNS(dnums)=t;
            dnums=dnums+1;
        end
    end
end
% verify the real non-linear leaf
dnumn=1;
for t=1:1:N
    if (MARKNLIN(t)==1)
        if (MARKLIN(t)==0)
            DYNN(dnumn)=t;
            dnumn=dnumn+1;
        end
    end
end
% leaf removal
marksen=0;
for i=1:1:M1
    for j=1:1:3
        interver=MAT(i,j);
        for k=1:1:(dnums-1)
            if (DYNS(k)==interver)
```

```
                    interjud=1;
                    break
                else
                    interjud=0;
                end
            end
            if (interjud==1)
                break
            end
        end
        if (interjud==1)
            marksen=marksen+1;
        else
            if (i==1)
                if (marksen==1)
                    % MAT(1,:) is null
                end
            else
                for j=1:1:3
                    % reconstruct the matrix
                    MAT(i-marksen,j)=MAT(i,j);
                end
            end
        end
    end
    funM1=M1;
    M1=M1-marksen;
    for i=(funM1+1):1:M
        interver=MAT(i,1);
        for k=1:1:(dnums-1)
            if (DYNS(k)==interver)
                itu=1;
                break
            else
                itu=0;
            end
        end
        if (itu==1)
            marksen=marksen+1;
        else
            internp1=MAT(i,2);
            internp2=MAT(i,3);
            for j=1:1:(dnumn-1)
                if (DYNN(j)==internp1)
```

```
                        for k=1:1:(dnumn-1)
                            if (DYNN(k)==internp2)
                                itunp=1;
                                break
                            else
                                itunp=0;
                            end
                        end
                        if (itunp==1)
                            break
                        end
                    else
                        itunp=0;
                    end
                end
                if (itunp==1)
                    merksen=marksen+1;
                else
                    for j=1:1:3
                        % reconstruct the matrix
                        MAT(i-marksen,j)=MAT(i,j);
                    end
                end
            end
        end
        M=M-marksen;
        M2=M-M1;
        if (marksen==0)
            markleaf=0;
        end
    end
    %------ end the process of leaf removal algorithm
    if (M==0)
        interqnum=interqnum+0;
        qnum;
    else
        procenum=0;
        MARKSUM=MARKLIN+MARKNLIN;
        for i=1:1:N
            if (MARKSUM(i)>0)
                procenum=procenum+1;
            end
        end
        qnum;
```

```
        interqnum=interqnum+(procenum/N);
   end
end
   PROCE(pnum,alphanum)=interqnum/qmaxnum;
   alphanum=alphanum+1;
end
pnum=pnum+1
end
```

附录 2：（关于制图的 MATLAB 源程序）

```
% The program to graph the numerical calculation result of Leaf-Removal
for i=1:1:261
      Xlable(i)=LFMAT(1,i);
      Ylable(i)=LFMAT(4,i);
end
plot(Xlable,Ylable,'-ks','LineWidth',1.5,'MarkerEdgeColor','r','MarkerFaceColor','g','
MarkerSize',10)
```