软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2011,22(5):929−937 [doi: 10.3724/SP.J.1001.2011.03814]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

基于等价类划分的配置求解与解释计算*

李宏博[1,2], 李占山[1,2+], 韩文成[1,2]

[1](吉林大学 计算机科学与技术学院,吉林 长春 130012)

[2](吉林大学 符号计算与知识工程教育部重点实验室,吉林 长春 130012)

Configuration Solving and Computing Explanations Based on Equivalence Class Partition

LI Hong-Bo[1,2], LI Zhan-Shan[1,2+], HAN Wen-Cheng[1,2]

[1](College of Computer Science and Technology, Jilin University, Changchun 130012,China)

[2](Symbol Computation and knowledge Engineer of Ministry of Education, Jilin University, Changchun 130012, China)

+ Corresponding author: E-mail: zslizsli@163.com

Li HB, Li ZS, Han WC. Configuration solving and computing explanations based on equivalence class partition. *Journal of Software*, 2011,22(5):929−937. http://www.jos.org.cn/1000-9825/3814.htm

Abstract: Some variables in a constraint-based configuration model may not have any immediate or indirect relationship between them; therefore, these variables will never intervene with each other in a constraint propagation. According to this characteristic of configuration, this paper proposes a preprocessing approach based on equivalent class partition, which can be used in the process of building the product model, so that the original configuration problem can be effectively divided into several sub-problems, and these sub-problems can be solved independently. The two backtracking-strategies are used in testing this method. The experimental results show that this method can effectively improve searching efficiency. Moreover, the method is integrated with the QUICKXPLAIN algorithm to compute conflict explanations, and the result show that it can also improve the efficiency of the algorithm.

Key words: constraint satisfaction; configuration; equivalence class; backtracking; conflict explanation

摘 要: 基于约束的配置模型中会有一些变量之间不存在任何直接或间接的约束关系,这样的变量之间进行约束传播不会互相影响取值.基于配置问题的这一特点,提出了一种等价类划分的思想,用于构造产品模型时的预处理技术,可以有效地将原问题划分为若干子问题,证明了这些子问题可以分别处理.分别采用两种回溯策略对求解效率进行了测试,结果表明能够有效地提高求解效率.最后,等价类划分方法与计算解释的 QUICKXPLAIN 算法集成计算冲突解释,测试结果表明,经过等价类划分后,同样可以有效地提高计算解释的效率.

关键词: 约束满足;配置;等价类;回溯;冲突解释

中图法分类号: TP181 **文献标识码**: A

约束满足问题(constraint satisfaction problem,简称 CSP)[1]是人工智能领域一个十分重要的分支,很多应用问题都可以用约束满足问题来建模,如 *N*-皇后、图着色问题、配置问题、作业调度问题等.约束满足问题

* 基金项目: 国家自然科学基金(60773097, 60873148, 60973089); 吉林省科技发展计划(20071106, 20080107)
收稿时间: 2009-04-29; 修改时间: 2009-11-26; 定稿时间: 2010-01-05

$CSP=(X,D,C)$,其中,X 是一个有限的变量集合$\{x_1,x_2,\ldots,x_n\}$;D 是所有变量对应的值域$\{D_1,D_2,\ldots,D_n\}$集合,其中,D_i 对应 x_i 的值域;C 是一个有限的约束集合.一个约束满足问题的解是对于每个变量 x_i 在其值域 D_i 中选取一个值构成一个集合 S,使得 S 中所有的变量取值满足 C 中的所有约束.约束满足问题在实际应用中有两个优点:表述方式更接近于实际问题;约束满足问题中的变量直接对应于问题实体,且约束也无须转换为线性不等式,公式表达更简练,解也更易理解.寻求约束满足问题的解,可通过对变量值域搜索来求解.经典回溯搜索算法(backtracking,简称 BT)[2]是一个完备的核心算法,但由于其搜索效率较低,目前已经很少被单独采用,后来通过引入回顾和展望两种策略显著地提高了求解效率.1993 年,Ginsberg 提出的动态回溯方法[3]是回顾策略的典型代表.该方法对变量值之间的删除解释作记录,搜索过程中一旦出现当前变量没有可用值时,通过查找删除记录能够直接修改当前变量的删除解释变量,并且将回溯时与当前变量无关的变量取值保留下来,避免了重复工作,提高了搜索效率.而展望策略则是将约束传播技术[4]嵌入到 BT 算法框架下.约束传播技术主要以弧相容技术为主,弧相容技术以 AC3[5],AC4[6]和 AC2001[7]为代表.1994 年,Freuder 提出了维持弧相容(maintaining arc consistency,简称 MAC)[8].MAC 算法在搜索过程中维持弧相容状态,每次变量赋值后利用 AC 算法进行约束传播,如果得到一个弧相容的状态,则赋值成功,否则赋值失败.MAC 是一个主流的搜索技术.由于弧相容维护过程中消耗大量时间,因此,2008 年孙吉贵和朱兴军等人提出了一种基于预处理的约束满足问题求解算法 BT+MPAC[9].该算法在求解前先对所有变量进行预处理,将不相容的值记录,在求解过程中将不相容的值直接删除,避免了弧相容维护所消耗的时间.

基于约束的配置问题[10]是一类特殊的约束满足问题,产品模型中零部件集合对应 CSP 中的 X,各个零部件参数的可配置值的集合对应 CSP 中的 D,零部件参数配置值之间的制约关系集合对应 CSP 中的 C.与普通约束满足问题相比,配置问题具有以下 3 个特点:

(1) 产品模型确定后短期内不会改变;

(2) 配置问题在配置过程中难免出现冲突,为了指导用户进行配置,配置器必须要给出冲突解释;

(3) 配置问题的产品模型通常都是约束密度较稀疏的问题,并且不同的部件之间可能不存在约束关系,即模型中可能存在一些不相关的变量.

约束满足问题求解算法完全适用于配置问题求解,能够很好地描述各种约束关系.无论一个 CSP 表示的配置是部分的还是完全的,都可以通过检验它的一致性来验证并对该配置进行修改.由于配置问题具有特点 1,因此预编译技术在配置问题中得到了很好的应用,如文献[11,12].

目前,计算解释方法主要分为侵入式和非侵入式两种[13]:侵入式策略需要记录较多的不相容信息,其典型代表是 ATMS 方法[14];非侵入式主要以 QuickXplain[15]为代表,该方法不需要记录相关的删除解释信息,在计算冲突时采用二分策略,并且可以根据用户的偏好给出合理解释,是计算解释的主流方法.

但无论是动态回溯方法还是 BT+MPAC 进行配置问题求解时,如果对问题中变量进行分类管理,则可以减少一些无用的回溯和遍历操作;利用 ATMS 方法计算解释需要做大量记录,占用空间较大,并且推理过程复杂;QuickXplain 计算解释方法虽然避免了 ATMS 方法的缺陷,但是由于没有利用参数之间的信息进行分类管理,因此发生配置冲突计算极小冲突过程中做了很多无意义的工作.

本文针对配置问题的特点 3,提出一种等价类划分的方法,将基于约束的配置问题进行分解来实现对配置问题的预处理.通过等价类划分方法可将原问题模型分解为几个子问题分别处理,提高求解和计算解释的效率.通过二元随机约束满足问题模拟用户配置进行测试,实验结果表明,对于可分解的随机问题,应用 BT+MPAC 算法进行求解时,效率有明显提高.我们选取的测试案例结果显示,回溯次数和搜索效率都可以提高 3 000 倍以上.应用动态回溯算法进行求解时,等价类划分预处理后的算法效率在问题规模较大时也有明显提高;与当前主流的非侵入式 QuickXplain 算法结合计算解释时,与原有的 QuickXplain 算法相比可明显减少一致性检查的次数.

本文第 1 节重点介绍等价类划分的方法.第 2 节主要介绍在经过等价类划分后在求解方面的改进.第 3 节分析基于等价类划分的 QuickXplain 算法的优化.第 4 节给出在求解和解释两个方面的实验结果.最后给出结论和展望.

1 等价类划分

为了便于描述问题,我们把所讨论的约束满足问题局限于二元约束满足问题.文献[16,17]指出,变量的传播次序对算法效率起着非常重要的作用.我们考虑这样一个 CSP 问题,$X=\{a,b,c,d,e,f\}$,$D=\{1,2,3,4\}$(表示每个变量的值域都一样),$C=\{a\neq d,b=e,c>e,f>d\}$.在这个约束满足问题中,$a$ 的取值无论怎样进行约束传播也不会影响到 b 的取值.同样,f 取值也不会影响到 b 的取值.进一步观察我们发现,a 能够通过约束传播影响到的只有 f,d 两个变量.同样,f 影响到 a,d;d 影响 a,f.这样,a,f,d 之间通过约束传播互相影响,b,c,e 互相影响,$\{a,d,f\}$,$\{b,c,e\}$这两个集合中的变量都无法影响到另外一个集合中的变量取值.此例中,求解时理论搜索空间是 $4^6=4096$.把原问题分成两个子问题之后搜索空间变成 $4^3+4^3=128$,按照 a-b-c-d-e-f 的赋值顺序利用回溯法进行求解,问题的搜索树如图 1 所示.

图 1 和图 2 中,F 表示搜索失败,发生回溯;S 表示搜索成功,找到解.图 1 中的搜索树共发生了 9 次回溯才找到解,如果将约束传播互不影响的两个变量集合$\{a,d,f\}$,$\{b,c,e\}$分开赋值,将得到图 2 所示的搜索树,共发生了 6 次回溯,两种顺序所求到的解是一样的,但分解后回溯次数明显减少.为了有效地描述上述案例中的问题分解思想,我们给出一些定义.

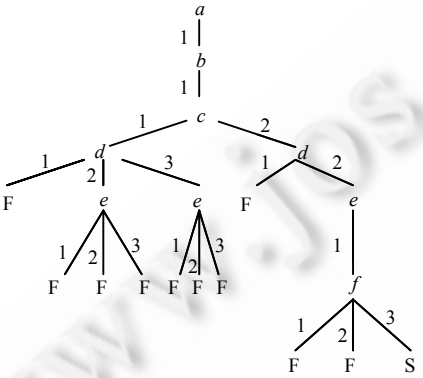

Fig.1 Searching tree with order *abcdef*

图 1 *abcdef* 顺序搜索树

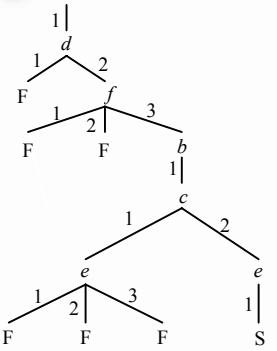

Fig.2 Searching tree with order *adfbce*

图 2 *adfbce* 顺序搜索树

定义 1(直接相关). 对于一个 CSP 问题(X,D,C),如果有 $x_1\in X,x_2\in X$,并且存在一条约束 $c\in C$,使得 $x_1\in X(c)$,$x_2\in X(c)$,则称 x_1,x_2 之间存在直接相关关系.其中,$X(c)$表示与 c 相关的变量集合.

定义 2(间接相关). 对于一个 CSP 问题(X,D,C),如果有 $x_1\in X,x_2\in X$,并且存在 $x_3\in X$,满足 x_1 与 x_3 之间存在直接相关关系,x_2 与 x_3 之间存在直接相关关系,则称这两个变量 x_1 与 x_2 之间存在间接相关关系.

定义 3(相关关系). 直接相关和间接相关统称为相关关系.

性质 1. 相关关系具有传递性质,即如果 x_1,x_2 之间存在相关关系,x_1,x_3 之间也存在相关关系,则 x_2,x_3 之间也存在相关关系.

定义 4(等价类). 对于一个 CSP 问题(X,D,C),我们称一个变量集合 X'是这个 CSP 问题的一个等价类,如果 X'满足如下条件:

(1) X'是 X 的子集;

(2) X'中的任意两个变量之间都存在相关关系;

(3) X'中的任意变量 x 与 $X-X'$中的任意变量 y 之间都不存在相关关系.

一个约束满足问题的一个等价类就是它的一个子问题.如果一个约束满足问题中的任意两个变量都存在相关关系,即这个约束满足问题无法划分出一个以上等价类,则称这个 CSP 问题为不变等价类.

对于一个 CSP 问题(X,D,C),它的一个等价类是通过如图 3 所示的算法 1 得到的.

算法 2 可以将一个 CSP 问题划分成若干个子问题.由不变等价类的定义可知,算法 2 输出的所有子等价类都是不变等价类(如图 3 所示).

```
Algorithm 1. Computing an equivalence class.
Get_a_Subset(X,D,C)
Begin
    X′={};
    if X is not empty then
        choose a x_i from X;
        X′=X′∪{x_i};
    else return X is empty;
    while get a x_j that is not dealed from X′ do
        X′=X′∪{all the x_k that associate with x_j};
        x_j is dealed;
    return X′;
end
```

```
Algorithm 2. Computing all equivalence classes for a CSP.
Decompose(X,D,C)
begin
    while X is not empty do
        onesubset=Get_a_Subset(X,D,C);
        X=X−onesubset;
        subsets=subsets∪{onesubset};
    return subsets;
end
```

Fig.3 Equivalence class partition algorithm description

图 3 划分等价类算法描述

2 基于等价类划分方法的问题求解

基于约束的配置问题可以用约束满足问题的求解方法进行求解.主流的约束满足问题求解方法是 MAC 算法以及在 MAC 基础上的改进算法.某些特殊类型的问题可以通过基于等价类划分的方法将原问题划分为几个子问题分别求解,划分后问题的搜索空间将明显减小,并且求解时如果某个子问题无解则可以直接确定原问题无解.根据等价类划分方法将一个大问题划分成两个以上小问题可以减少许多不必要的操作,这一特点在回溯策略中表现得尤为明显.

定理 1. 一个约束满足问题如果可以通过算法 2 划分成 1 个以上子问题,那么这些子问题可以分别独立求解,并且所有子问题的解并在一起就是原问题的解.

证明:由等价类定义可知,一个 CSP 问题中的两个等价类的任意变量之间不存在相关关系,因此在进行约束传播时,这两个等价类中任意变量的取值都不会制约另一个等价类中的任意一个变量的取值.因此,这些子问题的解之间不会出现冲突.所以,这些子问题的解并在一起构成原问题的解. □

定理 2. 给定一个约束满足问题 $CSP=\{X,D,C\}$,$|X|=n$,$|D_i|=d$,则未划分时问题最坏搜索空间为 d^n,如果 CSP 可以应用算法 2 划分为 m 个变量个数相同的子问题分别求解,则问题最坏搜索空间变为 $md^{n/m}$.

证明:因为经过划分后,每个子问题的最坏搜索空间都是 $d^{n/m}$,所以 m 个子问题搜索空间就是 $md^{n/m}$. □

性质 2. 对于一个可以应用算法 2 划分为两个以上子问题的 CSP,应用深度优先搜索策略进行求解,在不记录回溯位置的情况下,等价类划分后的回溯次数小于等于未划分时的回溯次数.

性质 3. 对于一个可以应用算法 2 划分为两个以上子问题的 CSP,应用动态回溯算法求解,划分前后回溯次数不变.

3 配置问题解释计算

配置解释的计算目前主要有侵入式和非侵入式两种.侵入式解释计算方法在求解过程中要对变量值删除原因进行记录,当问题规模较大时会占用很大空间.我们这里主要讨论非侵入式方法的典型代表——QuickXplain 算法.该算法是基于二分策略的思想在发生冲突后计算冲突,其算法描述[15]如图 4 所示.

Junker 指出,QuickXplain 算法把大部分时间都耗费在一致性检查过程中,因此,如果调用 *isConsistent* 的次数减少的话,一致性检查的耗时也将减少.Junker 给出了表 1 中 3 种不同的二分策略进行一致性检查的次数.

由表 1 可以看出,一致性检查的次数和初始冲突集的约束个数 n 成正比.Junker 还指出,一个问题如果可以被分解并且一个优先冲突集完全在某一个子问题中,那么 n 的大小就受子问题大小的限制了.等价类划分方法恰好是可以将一个问题分解成若干个子问题,并且一个优先冲突集一定存在于某一个子问题中,如果用户配置

到某个变量 i 时发生冲突,可以将与 i 在同一等价类中的用户配置变量作为求最小冲突的初始冲突集.而不在同一等价类中的用户配置变量将不会影响 i 的取值,所以不必考虑.这样就减小了初始 n 的个数,达到提高效率的目的.

Algorithm. *QUICKXPLAIN*(B,C,<).
1. if *isConsistent*($B\cup C$) return 'no conflict'
2. else if $C=\varnothing$; then return *null*;
3. else return *QUICKXPLAIN′*(B,B,C,<);

Algorithm. *QUICKXPLAIN′*(B,E,C,<).
1. if $E\neq\varnothing$ and not *isConsistent*(B) then return $\varnothing$;
2. if $C=\{a\}$ then return $\{a\}$;
3. let $a_1,a_2,\ldots,a_n$ be an enumeration of C that respects <;
4. let k be *split*(n) where $1\leqslant k<n$;
5. $C_1:=\{a_1,a_2,\ldots,a_k\}$ and $C_2:=\{a_{k+1},a_{k+2},\ldots,a_n\}$;
6. $E_2:=$*QUICKXPLAIN′*($B\cup C_1;C_1;C_2;<$);
7. $E_1:=$*QUICKXPLAIN′*($B\cup E_2;E_2;C_1;<$);
8. return $E_1\cup E_2$;

Fig.4 Description of QuickXplain algorithm

图 4 QuickXplain 算法描述

Table 1 Consistency check times

表 1 一致性检查次数

Split(n)	*BestCase*	*WorstCase*
$n/2$	$\log_2(n/k)+2k$	$2k\log_2(n/k)+2k$
$n-1$	$2k$	$2n$
1	k	$n+k$

如果可以将原始问题按照等价类划分方法分为 m 个变量个数相同的子问题,那么平均情况下表 1 中的 n 将变成 n/m,QuickXplain 的 3 种二分策略进行一致性检查的次数将分别变为表 2 的结果.

Table 2 Consistency check times with equivalence class partition

表 2 等价类划分后一致性检查次数

Split(n)	*BestCase*	*WorstCase*
$n/2$	$\log_2(n/k)+2k-\log_2 m$	$2k\log_2(n/k)+2k-2k\log_2 m$
$n-1$	$2k$	$2n/m$
1	k	$n/m+k$

经过划分后,假设平均情况下每个子问题中的变量个数相同,得到表 2 中的检查次数.从表 2 中可以看出,经过等价类划分后的一致性检查次数小于等于表 1 中原问题的一致性检查次数.减少的次数主要受 m 的影响,m 越大,初始 n 被缩小的倍数越多,一致性检查次数就越少.实际应用中,划分后的子问题中变量个数往往不同,我们有如下定理:

定理 3. 基于等价类划分的 QuickXplain 算法计算配置冲突解释,一致性检查次数小于等于未划分时的一致性检查次数.

4 实验结果

我们选择二元随机约束满足问题经典模型 Model B[18]进行测试.由于问题具有随机性,所以大多数约束满足问题求解算法都可以用这种方式进行测试.Model B 问题模型由 4 个参数控制,$\langle n,m,p_1,p_2\rangle$.其中,$n$ 是问题中变量个数;m 是每个变量的值域大小,其中每个变量的值域大小都是相同的;p_1 表示约束图中约束的密度,即在约束图 G 中均匀选择 $p_1\times n\times(n-1)/2$ 条约束;p_2 表示约束的松紧度,对于每条约束均匀地选 $p_2\times m\times m$ 个值对作为满足约束的值对.测试平台为 Windows XP sp3/Pentium(R) D CPU 3.40GHz/2GDDR2 SDRAM/JDK 1.5.0.由于随机问题不容易生成可以划分等价类的问题,我们先生成子问题,然后将子问题随机合并成一个大问题进行模拟测试.

4.1 BT+MPAC

该算法是由孙吉贵、朱兴军等人于 2008 年提出的[9],其主要思想是在求解前进行预处理,避免了搜索过程

中的弧相容维护操作,进而达到提高求解效率的目的.该算法的求解效率是 MAC 算法的 2 倍~50 倍[9].下面采用含有 30 个变量的二元随机问题进行测试,测试中采用可以划分出两个(15,20,0.4,p_2)的子问题,p_2 为图 5 中的 x 轴,每组参数分别测试 50 次后计算平均值,统计后求解效率和回溯次数对比如下:

BT+MPAC 可以有效减少不必要的搜索分支,并且在搜索过程中避免了弧相容维护的代价,因此效率比深度优先回溯算法提高很多.但由于 BT+MPAC 也是基于深度优先回溯策略,回溯过程中是按顺序回溯的,因此无法避免深度优先策略的弊端,与普通深度优先回溯存在相同的问题,和第 1 节提到的例子是一样的.为避免深度优先回溯策略存在的问题,接下来我们采用一种记录式回溯搜索策略进行测试.

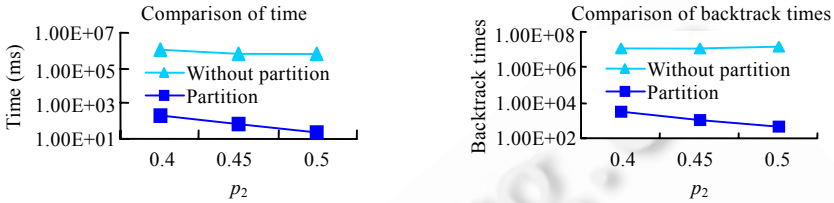

Fig.5 Experimental results of BT+MPAC

图 5 BT+MPAC 测试结果

4.2 动态回溯

动态回溯算法在每次给变量赋值前都要先进行约束传播,将不相容的值删除,同时记录排除解释用于回溯时查找回溯位置.因为动态回溯算法在执行过程中记录了大量的排除解释信息,因此可以避免第 2 节例子中的多余回溯操作.所以,我们以动态回溯策略为基准来测试等价类划分与动态回溯相结合方法在求解效率上的提高.我们选择动态回溯算法思想中较优的一种实现策略[19]进行测试.我们选取如下一组问题参数作为子问题:(100,30,0.04,0.35),子问题个数分别选择 2~5 个,每种情况分别测试了 50 组数据然后求平均值后统计结果如下:

经过测试我们发现,4 组数据的耗时平均情况分别提高了 13.7%~49.5%;并且随着子问题个数增加,提高的效果越来越明显(如图 6 所示).

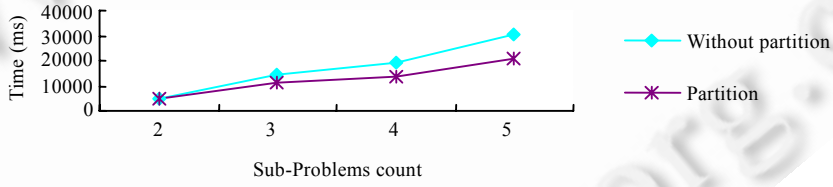

Fig.6 Experimental results of dynamic backtracking

图 6 动态回溯测试结果

4.3 QuickXplain

我们采用与前面相同的策略生成一些可以划分等价类的二元随机约束满足问题,随机问题参数为(30,5,0.07,0.5).模拟用户配置过程,随机选取变量配置为值域中的一个随机取值,我们采用 $split(n)=n/2$ 的情况进行 50 次测试,测试统计数据平均值比较如下:

由图 7 中的数据可以看出,经过等价类划分后的冲突计算减少了一致性检查的次数;并且随着可划分出的子问题个数的增加,两条线在 Y 轴方向上的距离也在逐渐增大.这说明在平均情况下,随着子问题个数的增加,划分后的计算解释效率比未划分的计算解释效率提高得越来越明显.最坏情况下,是和没有经过等价类划分的一致性检查次数相同;平均情况下,当划分出 2~5 个等价类时,一致性检查次数可以减少 25%~42%;当问题规模增大时,提高效果将更明显.因此,等价类划分方法用在配置计算解释中可以有效地提高效率.

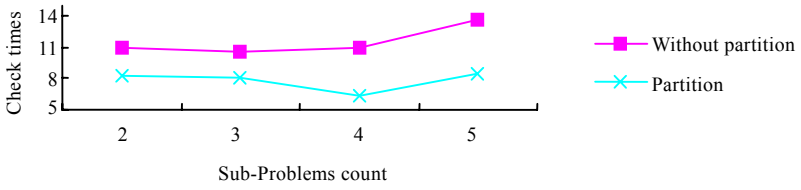

Fig.7 Experimental results of QuickXplain

图 7 QuickXplain 测试结果

5 结论与展望

配置问题与其他约束满足问题有一个不同点,就是配置问题模型中可能有一些变量之间不存在相关关系,这样的变量之间无论怎样进行约束传播都不会互相影响取值.基于配置问题的这一特点,本文提出了一种等价类划分的思想,用于分解产品模型.由于配置模型确定后在一段时间内相对稳定,因此可以在构造产品模型时通过预处理技术将原问题划分为若干子问题,并证明了这些子问题可以分别处理.接着,分别采用两种求解策略对求解效率进行了测试.测试结果显示,经过等价类划分后的问题求解效率显著提高.在计算解释方面,本文分析了主流的非侵入式计算解释算法 QUICKXPLAIN.根据 QUICKXPLAIN 计算解释的思想,经过等价类划分后可以减少一致性检查的次数.由于 QUICKXPLAIN 算法的大部分时间都消耗在一致性检查过程中,因此,经过等价类划分后可以有效提高 QUICKXPLAIN 计算解释的效率.配置问题是一个交互式的过程,一个大型的配置系统中如果用户要配置出一个满意的产品,可能要经过多次的交互,修改配置变量值.这时,计算解释的效率就显得尤为重要.等价类划分方法可以有效地将原始大问题划分为若干个子问题进行分别处理,最坏情况下,即原始问题中的所有变量之间都存在直接或间接的约束关系,原始问题就是一个不变等价类,此时的求解效率和计算解释效率将退化为未经过等价类划分时的效率.本文测试数据表明,通常情况下,无论是求解过程还是计算解释的过程,效率都有大幅度提高.因此,等价类划分思想可以在配置问题中得到很好的应用.等价类划分思想是可以应用于所有约束满足问题的.由于配置问题比较容易划分出等价类,因此本文主要针对配置问题进行讨论.下一步工作将考虑将等价类划分思想进一步扩展,将其应用在其他可以用约束满足问题进行建模的问题中.

References:

[1] Freuder EC, Mackworth AK. Constraint satisfaction: An emerging paradigm. In: Rossi F, van Beek P, Walsh T, eds. Handbook of Constraint Programming. Amsterdam: Elservier, 2006. 13−23. [doi: 10.1016/S1574-6526(06)80006-4]

[2] van Beek P. Backtracking search algorithms. In: Rossi F, van Beek P, Walsh T, eds. Handbook of Constraint Programming. Amsterdam: Elservier, 2006. 86−118. [doi: 10.1016/S1574-6526(06)80008-8]

[3] Ginsberg ML. Dynamic backtracking. Journal of Artificial Intelligence Research, 1993,1(1):25−46.

[4] Bessiere C. Constraint propagation. In: Rossi F, van Beek P, Walsh T, eds. Handbook of Constraint Programming. Amsterdam: Elservier, 2006. 29−83.

[5] Mackworth AK. Consistency in networks of relations. Artificial Intelligence, 1977,8(1):99−118. [doi: 10.1016/0004-3702(77)90007-8]

[6] Mohr R, Henderson TC. Arc and path consistency revised. Artificial Intelligence, 1986,28(2):225−233. [doi: 10.1016/0004-3702(86)90083-4]

[7] Bessiere C, Régin JC. Refining the basic constraint propagation algorithm. In: Proc. of the IJCAI 2001. 2001. 309−315.

[8] Sabin D, Treuder EC. Contradicting conventional wisdom in constraint satisfaction. In: Cohn AG, ed. Proc. of the 11th European Conf. on Artificial Intelligence. Amsterdam: John Wiley and Sons, 1994. 125−129. [doi: 10.1007/3-540-58601-6_86]

[9] Sun JG, Zhu XJ, Zhang YG, Li Y. An approach of solving constraint satisfaction problem based on preprocessing. Chinese Journal of Computers, 2008,31(6):919−926 (in Chinese with English abstract).

[10] Li ZS, Wang T, Sun JG, Zhang CH. Research on principle of product configurator. Application Research of Computers, 2004, 24(10):24−26 (in Chinese with English abstract).

[11] Vempaty NR. Solving constraint satisfaction problems using finite state automata. In: Swartout WR, ed. Proc. of the AAAI. San Jose: AAAI Press, 1992. 453−458.

[12] Subbarayan S, Jensen RM, Hadzic T, Andersen HR, Møller J, Hulgaard H. Comparing two implementations of a complete and backtrack-free interactive configurator. In: Barták R, Junker U, Silaghi MC, Zanker M, eds. Proc. of the Workshop on CSP Techniques with Immediate Application (CP-2004). 2004. 97−111. http://itu.dk/people/sathi/papers/cspia2004.pdf

[13] Haag A, Junker U, O'sullivan B. A survey of explanation techniques for configurators. In: Sinz C, Haag A, eds. Proc. of the Workshop on Configuration (ECAI 2006). 2006. 8−13. http://fmv.jku.at/ecai-config-ws-2006/W16.pdf

[14] de Kleer J. An assumption-based truth maintenance system. Artificial Intelligence,1986,28(1):127−162. [doi: 10.1016/0004-3702(86)90080-9]

[15] Junker U. QuickXplain: Preferred explanations and relaxations for over-constrained problems. In: Ferguson G, McGuinness D, eds. Proc. of the AAAI. San Jose: AAAI Press, 2004. 167−172.

[16] van Dongen MRC, Mehta D. Queue representation for arc consistency algorithms. In: McGinty L, Crean B, eds. Proc. of the 15th Irish Conf. on Artificial Intelligence and Cognitive Science. 2004. 334−343. http://csweb.ucc.ie/~dongen/papers/AICS/04/aics04

[17] Wallace RJ, Freuder EC. Ordering heuristics for arc consistency algorithms. In: Mellish CS, ed. Proc. of the AI/GI/VI'92. 1992. 163−169. http://4c.ucc.ie/web/upload/publications/misc/wall

[18] Smith BM, Dyer ME. Locating the phase transition in binary constraint satisfaction problems. Artificial Intelligence, 1996,81: 155−181. [doi: 10.1016/0004-3702(95)00052-6]

[19] Li HB, Li ZS, Ai Y, Du HY. On research of optimization strategy for dynamic backtracking. In: Wang XZ, Yeung DS, eds. Proc. of the Int'l Conf. on Machine Learning and Cybernetics, Vol.1. New York: IEEE Computer Society, 2009. 266−271.

附中文参考文献:

[9] 孙吉贵,朱兴军,张永刚,李莹.一种基于预处理技术的约束满足问题求解算法.计算机学报,2008,31(6):919−926.

[10] 李占山,王涛,孙吉贵,张朝辉.产品配置器的工作机理研究.计算机应用研究,2004,24(10):24−26.

附录

性质 2. 对于一个可以应用算法 2 划分为两个以上子问题的 CSP,应用深度优先搜索策略进行求解,在不记录回溯位置的情况下,等价类划分后的回溯次数小于等于未划分时的回溯次数.

证明:对于两个在赋值路径上不相邻的在同一等价类中的变量 i 和 j,它们在赋值路径上间隔了 m 个变量,这 m 个变量和 i,j 不在同一等价类中,这 m 个变量经过了 k 次回溯后达到局部相容状态.假设 j 当前赋值失败是由 i 导致的,那么如果想让 i 和 j 的赋值相容,则必须回溯到 i.但它们之间相隔了 m 个变量,则还要经过最坏 md(其中 d 为变量值域大小)次搜索和 m 次回溯才能回溯到 i.当 i 选择了可以与 j 赋值相容的值之后,这 m 个变量又要经过 k 次回溯才能重新赋值相容.因此,这个回溯过程中多执行了 $k+m$ 次回溯和 md 次搜索.如果第 1 次回溯到 i 选择的新值不能与 j 相容,还要经过更多的 $k+m$ 次回溯和 md 次搜索才能赋值相容.因此,如果我们应用等价类划分方法将同一等价类中的变量单独求解,就可以避免上述的 $k+m$ 次多余回溯操作.当原问题是一个不变等价类时,等价类划分后的回溯次数等于未划分时的回溯次数. □

性质 3. 对于一个可以应用算法 2 划分为两个以上子问题的 CSP,应用动态回溯算法求解,划分前后回溯次数不变.

证明:由于动态回溯在回溯过程中记录了排除解释,发生回溯时直接回溯排除解释记录中的变量,因此当集成等价类划分办法时,没有增加多余的回溯操作. □

定理 3. 基于等价类划分的 QuickXplain 算法计算配置冲突解释,一致性检查次数小于等于未划分时的一致性检查次数.

证明:我们以 v_2 表示当前所有已配置的变量个数,v_1 表示当前变量 i 所在的等价类中已经配置的变量个数.

由 v_1,v_2 所表示的意义可知,$v_1 \leqslant v_2, v_2/v_1$ 是初始 n 实际缩小的倍数.在实际应用中,有如下 3 种情况:

(1) $v_2/v_1>m$.此时,经过等价类划分后的一致性检查次数将比表 2 中的平均情况要少.

(2) $1<v_2/v_1<m$.此时,经过等价类划分后的一致性检查次数将比表 2 中的平均情况要多.

(3) $v_2/v_1=1$.此时,恰好用户当前配置的变量都是和 i 在同一个等价类中,这也是最坏的情况.这时,经过等价类划分的一致性检查次数将和没有划分时完全一样.

由以上 3 种情况分析可知,经过等价类划分后,一致性检查次数小于等于未划分时的一致性检查次数. □

李宏博(1985－),男,吉林公主岭人,硕士,主要研究领域为约束满足问题,配置问题.

李占山(1966－),男,博士,教授,主要研究领域为基于模型的诊断,约束满足问题,配置建模与求解,智能规划.

韩文成(1983－),男,硕士,主要研究领域为约束满足问题求解,产品配置器.