几何因果定性推理的基本原理和算法*

葛建新 杨 莉

(浙江大学 CAD&CG 国家重点实验室 杭州 310027) (国家智能计算机研究开发中心 北京 100080)

摘要 因果定性推理是一种通过分析描述物理系统行为和关系的约束找出系统内部各个成分之间的因果结构的推理方法. 本文提出一种基于约束和变量分析的因果定性分析模型和算法. 该方法在产品设计中有广泛的应用,利用这个模型和算法可较好地解决参数化设计中的几何推理问题,还可用作概念设计的工具,用于完成复杂系统设计任务的划分及定序、设计变量之间相互依赖关系分析等工作. 算法具有应用性强、效率和稳定性好、支持欠约束和多解问题等优点.

关 键 词 定性推理,因果推理,几何推理,约束求解,参数化设计.

中图分类号 TP391

产品的设计过程本质上是一个约束满足问题(CSP),大部分的产品设计过程可归结为约束规定、约束变换和求解以及约束求解评估这样一个循环过程. 在这个过程中所涉及的约束种类千差万别,有外形美观方面的约束,又有成本方面的约束,还有产品功能和性能方面的约束等. 在产品设计过程中各种类型的约束不断地被加入到所设计产品的约束集中,而产品约束集中的约束也可能在设计过程中被删除或修改. 产品设计的设计工作从逻辑上说就是从满足约束集中所有约束的解空间中选取一个或多个解. 当约束集解空间为空时,约束集中的约束相互矛盾,这种情形被称为约束过载,而当解空间有无限多个解时,这种情形被称为约束不足.

设计过程中约束的管理是一项复杂的过程,这是由于大部分设计所涉及的约束种类多、数量大且关系复杂,且设计过程中出现约束集过载或不足的情形非常多. 因而有效地进行约束管理对于产品的设计来说是十分重要的.

目前有关产品设计约束求解和管理方面的研究工作已在国内外展开,但主要重点放在几何约束分析和求解方面,应用的背景主要在参数化设计方面. 根据约束求解方法,这些研究大致可分为以下 4 类:第 1,数值约束求解. 这种方法是将几个约束集转化为一个由一系列等式或不等式组成的非线性方程组,然后通过数值迭代方法求出解.[1~3]这种方法的优点是通用性好、程序实现简单,但是这种方法的数值稳定性严重依赖初始值的设定,而且这种方法无法有效分析和处理约束过载和约束不足的情形. 更为严重的是对于复杂产品的设计,

* 本文研究得到国家自然科学基金资助. 作者葛建新,1965 年生,博士后,副教授,主要研究领域为计算机辅助设计,几何造型. 杨莉,女,1965 年生,博士后,副研究员,主要研究领域为人工智能,专家系统,软件工程.

本文通讯联系人:杨莉,北京 100080,国家智能计算机研究开发中心

本文 1996-06-20 收到修改稿

该方法的求解速度无法满足实际需求. 第 2，基于规则的约束求解. 这种方法通过规则重写或匹配技术分析有关产品的约束集，确定产品的各个基本元素之间的关系，从而确定产品的求解过程.[4~8]这种方法提供了一个产品设计构造过程分析和求解的有效框架，但是由于基于规则技术本身的限制这种方法也存在一些问题，例如无法处理元素之间循环依赖问题，而且实现效率也较低. 第 3，基于符号代数的方法. 这种方法是将约束集转化一个代数方程组，然后通过符号代数方法，例如 Grobner 基或 Wu-Ritt 方法将方程组求解出来.[9,10]通过符号代数技术还可得到约束对象的相互依赖关系. 但是这种方法是基于代数簇分解技术，在通用性和实现效率方面还有待改进. 第 4，基于图的求解方法. 这种方法首先分析约束得到反映的约束对象之间的依赖关系和求解次序的依赖图，然后根据依赖图进行约束求解从而得到最终的设计结果.[11,12]这种方法效率高，而且能有效处理约束矛盾和约束不足的情况. 但以前的工作尚未解决约束对象之间存在依赖回路这个问题，使应用范围受到限制.

以上这些方法虽然解决了参数化设计中几何约束的分析和求解问题，但或多或少还存在着一些问题，尤其对于产品设计，这只是解决了产品详细设计阶段的几何约束求解问题，而对于支持整个包括概念设计在内的产品设计来说，这些方法还有待于改进和推广. 针对以上问题，本文着重研究了产品设计过程中一般意义下约束分析和求解方法，主要工作包括提出了基于因果定性推理方法的约束分析方法及这种方法在产品设计中的一些具体应用.

1　约束的因果推理方法

1.1　基本概念

定性推理是通过对系统的结构、行为和功能描述以及它们之间的关系和因果性的研究，来探讨人类常识(定性)推理机制，以便有效地完成各项求解任务，它是一种跨领域的推理方法体系.[13]因果定性推理是定性推理的一个重要分支，它通过分析描述物理系统行为和关系的约束找出系统内部各个成分之间的因果结构，从而使人们更加有效地分析整个系统的内在结构并适当求解得出系统行为或状态的预测和仿真.[14]

约束是特定元素之间必须满足的一种关系，可以方便地表示为 $c=(type,x_1,x_2,\cdots,x_n)$，其中 $type$ 为约束类型，$x_i(i=1,\cdots,n)$ 为受约束的元素，称 x_i 为 c 的受约束元素，c 为 e_i 的相关约束.

在产品外形设计中，主要的引用元素包括几何元素、拓扑元素、与形状有关的形状特征、符号变量以及这些元素的内部属性，而在产品的功能设计中，引用元素范围更大，除上述元素外还包括产品的功能和性能等方面的属性元素. 这些种类繁多的元素将成为基于约束的产品外形表示方法中的主要引用元素.

每个被约束限制的元素具有各自的自由度，它是确定该元素所需不相关代数方程的个数，用 $FD(x)$ 表示元素 x 的自由度. 元素的自由度具有直观上的意义，例如无约束的自由二维点的自由度为 2，而在已知直线上的二维点的自由度为 1. 这是因为要确定这个点，前者需要两个代数方程，而后者仅需一个代数方程. 同理可知无约束的二维圆的自由度为 3.

由于约束本质上是一系列相关的数学方程，因而引入约束将导致它所限制的有关元素自由度的减少. 我们称这种由某约束 c 引起的相关几何元素自由度的减少量为该约束的自

由度亏损,记为 $LF(c)$.

约束的自由度亏损数值上与该约束相关的代数方程组个数相等.常见的约束及其亏损如下(在二维情形):

$(Radius\ C\ \ r)$	$LF=1$	// 圆 C 的半径为 r
$(Center\ C\ \ P)$	$LF=2$	// 圆 C 的圆心为 P
$(Dist\ \ P_1\ P_2\ \ d)$	$LF=1$	// 两点间距离为 d
$(Coord\ P\ \ x\ \ y)$	$LF=2$	// P 的坐标为(x, y)

设计人员通过约束描述了产品设计的要求,这种描述是一种形式化的面向符号的描述方法,因而需要一种机制将这些形式化的描述转化为具体的具有精确数值信息的产品几何外形信息.这个转换过程就是基于约束的推理和求解过程.

1.2 因果定性推理原始算法

因果定性推理的目标是通过分析约束以及相关的约束对象之间的关系得到约束对象之间的因果关系,即相互依赖关系.

在定义约束时实际上已隐含这样一个事实,即对于约束 $c=(type,x_0,x_1,\ldots,x_n)$,如果 $n+1$ 个元素 $x_0,x_1,\ldots,x_n$ 中任意 n 个元素被确定,则该约束可用于余下元素的求解,且该剩余元素被称为约束 c 的输出元素.因而每个约束都有一个输出元素,这实际上隐含了被约束对象之间的相互依赖关系,即约束的输出元素是依赖于该约束其它约束对象的,图1给出了约束 $c=(type,x_0,x_1,\ldots,x_n)$ 的情形,图中我们选择 x_0 作为约束 c 的输出元素,因而它依赖其它元素 $x_i(i=1,\ldots,n)$.

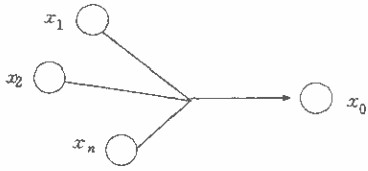

图1 约束的输出元素

进一步我们注意到在选定约束的输出元素之后,一旦这个约束生效,则其输出元素的自由度将下降,并且下降值正好为该约束的自由度亏损.在一开始每个对象均自由的、随着约束的引入和生效,约束对象的自由度逐渐降低,直至零为止,此时该对象无任何自由度,已被所加入的约束集完全确定.

对于每个约束,其输出元素的选择并非是任意的,若对于某一约束选定输出元素并使之生效,结果使输出对象的自由度变为负数,则该约束输出对象的选择非法,这是由于这种情况下用于确定该约束的输出元素的约束代数方程总数已多于该输出对象自身的自由度.对于一个约束集,如果每个约束的输出元素的选择均为合法,则我们称由该约束集确定的因果依赖关系是相容的,否则为不相容.

因而原先的因果定性推理问题就变为约束集约束输出变量的布局问题,即对约束集中的每个约束作输出元素的选择使得该约束集所确定的因果依赖关系是相容的.由于对于约束 $c=(type,x_1,\ldots,x_n)$ 来说,其输出变量的选择可为 $x_i(i=1,\ldots,n)$,总共有 n 个选法,因此解决该问题最原始的方法为对每个约束的输出变量选择进行枚举,然后检测所确定的因果依赖关系是否相容.在算法中假定给定的约束集为 $C=(c_1,c_2,\ldots,c_m)$,其中 $c_i=(t_i,x_{i1},x_{i2},\ldots,x_{ini})$,$(i=1,\ldots,m)$.

该算法是个递归过程,总的计算工作以 $try(1)$ 启动,初始时每个元素的自由度均设置为最大.算法基本过程简单,而且还可确定所有可能的相容的元素因果关系.但是这些优点是

以牺牲效率为代价的，假定每个约束涉及的元素个数均为 n，则上述算法的复杂度为 $o(n^m)$，这是无法令人容忍的. 因此有必要对算法进行改进.

算法 1. 因果定性推理原始算法

```
try (i)
{
    for (j = 1 to n_i) {
        选择 x_ij作为 c_i 的输出元素;
        FD(x_ij) = FD(x_ij) - LF(c_i);      /* 因 c_i 生效而调整其输出元素 x_ij的自由度 */
        if (FD(x_ij)≥0) {
            if (i==m)                       /* 已处理完最后一个约束 */
                输出相容的因果关系;
            else
                try(i+1);                   /* 递归处理第 i+1 个约束 */
        }
        FD(x_ij)=FD(x_ij) + LF(c_i);        /* 恢复选 x_ij作为 c_i 输出元素前的自由度 */
    }
}
```

1.3 因果推理的改进算法

算法的改进从 2 方面着手：①通过一些启发式方法减少算法选择约束输出元素时的盲目性；②采用增量式算法，即始终保持一个相容因果关系（初始时该相容关系为空），在每次增加或删除一个约束时只需对已有的相容因果关系作相应调整，不必从无到有从新构造现有约束集所确定的相容因果关系，这就使算法的总开销分摊到用户每次交互输入约束时相容关系的调整过程中去，从而使算法的响应时间满足实际需求.

由于从约束集删除约束不会使原约束集所确定的相容的因果关系发生矛盾，因而算法的关键是要解决增加一个约束而产生的问题. 每次增加一个约束时，应适当选择该约束的输出元素，并在必要时调整已输入约束集中约束的输出元素，使得加入这个约束后约束集所确定的因果关系仍为相容的.

下面我们给出增加一个约束的算法 $AddCons(c)$，其中 $c=(type,x_1,x_2,\cdots,x_n)$ 为被加入的约束. 为了防止算法的死循环和减少算法选择的盲目性，引进时间戳变量 timeStamp，它是一个初始值为零的全局变量，其在算法每次循环过程中对当前处理的约束和约束对象进行标记. 算法首先从约束对象 $x_1,x_2,\cdots,x_n$ 中找出自由度最大的元素 x_p，此时若 x_p 的自由度大于约束 c 的的自由度亏损，则选 x_p 作为约束 c 的输出元素，同时约束 c 生效后 x_p 的自由度将降低 $LF(c)$，算法成功返回. 否则需要调整以 x_p 为输出元素的约束的输出元素，直至最终每个元素的自由度均大于等于零.

这是一个简洁而有效的递归算法，而且时戳的引进保证了在存在相容的因果关系的条件下算法能在有限时间内将正确的因果关系找出来. 算法的基本过程直观上类似于半导性中电子填充空穴的过程，在考虑增加一个约束时若该约束所选择的输出元素的自由度降为负数，则我们可想象在这个元素上产生了一个某个阶的电子，而其它元素若存在自由度，则可将其想象为阶数为该自由度的空穴，因而算法执行过程就是一个电子在网中漫游，一旦电子碰到一个空穴，则电子部分或全部变吸收，这个过程一直持续到电子被空穴全部吸收. 由于引入了时间戳，算法的漫游不可局限在网络的某个局部，因而上述的结果可归纳为以下一个定理.

定理. 若约束集 C 能产生一个相容的因果关系，则上述算法在有限步内终止.

根据上述思路，可以得到上述定理的严格证明，由于篇幅的限制这里不再赘述. 事实上进一步分析表明在约束上附加时间戳无助于算法的正确性，但它作为启发式信息可以改进算法的效率.

算法 2. 因果定性推理改进算法

```
AddCons (c)          /* 增加一个约束 c，其中 c=(type,x1,x2,…,xn) */
{
    timeStamp=timeStamp+1;
    设置 c 的时间戳;
    xp=元素 x1,x2,…,xn 中自由度最大的元素;
    if (FD(xp)≥LF(c)) {
        选择 xp 为 c 的输出元素;
        FD(xp)=FD(xp)-LF(c);
        设置 xp 的时间戳;
        return;
    }

    xp=元素 x1,x2,…,xn 中时间戳最小的元素;
    设置 xp 的时间戳;
    选择 xp 为 c 的输出元素;
    FD(xp)=FD(xp)-LF(c);
    T=以 xp 为输出元素的所有约束并按时间戳大小从小到大排列;

    R=∅;                          /* R 为临时失效的约束集 */
    while (FD(xp)<0) {
        从 T 中取出一个约束 c;     /* 使 c 临时失效 */
        R=R+{c};
        FD(xp)=FD(xp)+LF(c);
    }

    while (R≠∅) {
        从 R 中取出一个约束 c;
        AddCons (c);              /* 重新使 c 生效 */
    }
}
```

下面我们给出 2 个具体的例子.

例 1：如图 2 对一个三角形有以下约束：

$c_1=(dist\ A\ C)$ /* A 与 C 有距离约束 */

$c_2=(dist\ A\ B)$ /* A 与 B 有距离约束 */

$c_3=(dist\ B\ C)$ /* B 与 C 有距离约束 */

$c_4=(hor\ A\ B)$ /* A 与 B 有水平约束 */

以上每个约束的自由度亏损均为 1，且受约束元素为 3 个点 A、B 和 C，其自由度初始时为均为 2. 图 3 为算法执行的一个过程，其中括号中的数字为几何元素当前自由度，斜杠后的数字为时间戳. 需要指出的是满足例 1 中的约束集的相容关系有多个，不只图 3 中的一个结果.

例 2：图 4(a)中的矩形可用以下约束描述：

$c_1=(HorDist\ B\ C)$ $c_2=(VerDist\ A\ B)$ $c_3=(Hor\ A\ D)$ $c_4=(Ver\ D\ C)$

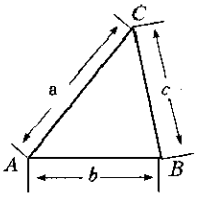

图2　一个简单的三角形

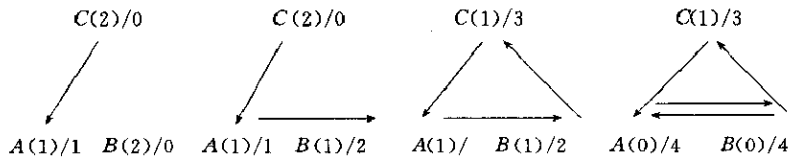

图3　例1的算法执行过程

执行本算法后最终结果如图4(b).

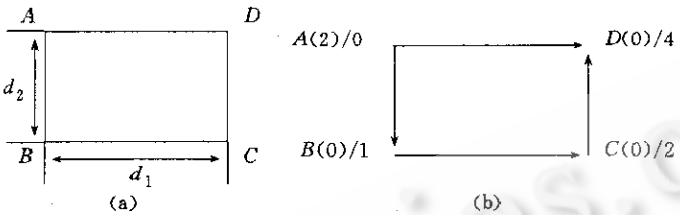

图4　例2及其计算结果

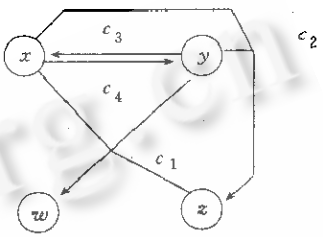

图5　例3及其计算结果

例3：设有下列通过方程表示的约束：

$c_1=(F_1,w,x,y,z)$；　　/* $F_1(w,x,y,z)=0$ */

$c_2=(F_2,x,y,z)$；　　/* $F_2(x,y,z)=0$ */

$c_3=(F_3,x,y)$；　　/* $F_3(x,y)=0$ */

$c_4=(F_4,x,y)$；　　/* $F_4(x,y)=0$ */

上述约束中每个变量的初始自由度均为1，而每个约束的自由度亏损均为1，应用算法2可得到图5的结果.

1.4　因果关系的抽取

上节的算法通过增量方式为一个约束集中的每个约束确定了约束的输出元素，根据约束输出元素的定义，这实际上已确定了被约束对象之间的相互依赖关系，即约束对象之间的因果关系. 但是由于存在2个对象之间相互依赖关系，即对象 A 依赖于对象 B，而 B 反过来又依赖于 A，因而上述关系并非半序关系. 这种现象使我们无法直接计算约束对象之间的依赖关系，为此首先为约束变量集为 $X=\{x_1,x_2,\cdots,x_m\}$ 构造一个 $m\times m$ 的邻接矩阵 A，定义如下

$$a_{ij}=\begin{cases}1 & \text{当且仅当存在约束 } c \text{ 使得 } x_i \text{ 和 } x_j \text{ 同时受 } c \text{ 作用且 } x_j \text{ 为 } c \text{ 的输出元素；}\\ 0 & \text{否则}\end{cases}$$

进一步可定义 A 的传递闭包 $R=A+A^2+\cdots+A^m$，R 反映了变量集 X 中元素之间的相互依赖关系，即若 R 中元素 $r_{ij}=1$，则说明 x_j 直接或通过一个间接的依赖链依赖于 x_i，否则 x_j 不依赖于 x_i. R 的计算可用经典的 Warshall 算法[15]，具体见算法3.

利用 R 定义 X 上的关系≈，对 $\forall x_i,x_j\in X$，$x_i\approx x_j$ 当且仅当 $r_{ij}=r_{ji}=1$，易证≈为等价关系，因而利用该等价关系对 X 作商空间 X'，进一步由 R 诱导在 X' 的关系上构成半序关系. 这个半序关系就是我们所需的因果关系.

利用上述方法可以方便地构造上一节几个例子中约束对象之间的因果关系，图6给出了例3的因果关系图，其中变量 x 和 y 已合并在一个等价类中.

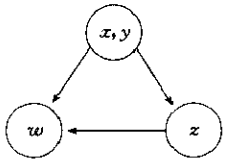

图6 例3的因果关系图

算法3. 构造邻接矩阵 A 的传递闭包 R

```
Closure (A)
{
    R=A;
    for (k=1 to n)
        for (i=1 to n)
            if(r_ik==1)
                R_i=R_i∨R_k          /* i行与k行逻辑和 */
    return R;
```

2 因果推理在产品造型中的应用

因果推理方法在产品造型中有许多重要的应用，最直接的应用就是参数化设计. 其基本方法是根据约束进行几何元素上的因果推理，推理的结果得到几何元素集等价类上的半序关系，根据半序关系可对每个等价类进行求解，求解次序的构造算法见文献[12]，而对每个等价类的求解则视情况而定，若等价类为单个元素，则该等价类中的唯一元素可用解析法求解，否则可将该等价类有关的约束转化成数值方程组，通过叠代方法计算得到结果.

这种方法与原有的一些方法相比有以下一些优点：(1)算法适用范围有所提高，解决了元素依赖网中存在闭环的复杂情形；(2)算法可靠性和效率比直接的数值方法或符号代数方法有所改进，而且在大部分绘图参数化设计中不存在依赖回路，故求解过程基本为解析的，因而效率和稳定性大幅度提高. (3)利用因果关系还可进一步分析多解的情形，而且在约束网无回路时可通过枚举列举出所有的解空间. (4)允许设计存在欠约束情形，这对于支持初步设计来说是相当重要的.

除了在参数设计方面的应用，因果定性推理还可用作概念设计方面的工具，例如复杂系统设计任务的划分以及定序，设计变量之间依赖性的分析等.

3 结论和未来工作

本文提出了一种基于约束和变量分析的因果定性分析模型和算法，利用这个模型和算法较好地解决了参数化设计等产品设计的问题，这种算法具有应用性强、效率和稳定性好，支持欠约束和多解问题等优点，并在产品设计中有广泛的应用.

值得进一步研究的问题包括算法复杂性分析、全部相容关系的快速求解以及欠约束情形下求解方法的确定等.

致谢 感谢浙江大学计算机系CAD课题组董金祥教授、陈德人副教授、李海龙博士、谭孟恩博士、何清法硕士、唐敏硕士、王勇硕士等同仁的帮助和支持.

参考文献

1 Light L, Gossard D. Modification of geometric models through variational geometry. Computer-Aided Design, 1982, **14(4)**: 209～214.

2 Lee K, Andrews G. Inference of the positions of components in an assembly: part 2. Computer Aided Design, 1985, **17(1)**: 20～24.

3 Rocheleau D N, Lee K. System for interactive assembly modeling. Computer-Aided Design, 1982,**14(4)**:65～72.
4 Aldefeld B. Variation of geometries based on a geometric-reasoning method. Computer-Aided Design, 1988,**20(3)**:117～126.
5 Sunde G. Specification of shape by dimensions and other geometric constraints. In: Wozny M J, Encarnacao J L eds. Geometric Modeling for CAD Applications, North Holland, 1988. 199～213.
6 Suzuki H, Ando H, Kimura F. Geometric constraints and reasoning for geometrical CAD systems. Computer & Graphics, 1990,**14(2)**:211～224.
7 Verroust A, Schonek F, Roller D. Rule-oriented method for parameterized computer-aided design. Computer-Aided Design, 1992,**24(10)**:531～540.
8 Yamaguchi Y, Kimura F. A constraint modeling system for variational geometry. In: Wozny M J, Turner J U, Preiss K eds. Geometric Modeling for Product Engineering, North Holland, 1990. 221～223.
9 Kondo K. Algebraic method for manipulation of dimensional relationships in geometric models. Computer-Aided Design, 1992,**24(3)**:141～147.
10 Buchanan S A, Pennington A de. Constraint definition system: a computer-algebra based approach to solving geometric-constraint problems. Computer-Aided Design, 1993,**25(12)**:741～750.
11 Dechter R, Peal J. Network-based heuristics for constraint-satisfaction problems. Artificial Intelligence, 1987,**34(1)**:1～38.
12 葛建新,彭群生,董金祥等.基于约束的形状自动求解新算法.计算机学报,1995,**18(2)**:114～126.
13 石纯一,陈见,赵永等.定性推理进展.第二届中国人工智能联合学术会议,1992. 128～133.
14 蔡勇,石纯一.定性推理中一种因果分析方法.模式识别与人工智能,1995,**8(3)**:33～42.
15 Base S. Computer algorithms: introduction to design and analysis. New York: Addison-Wesley Company, 1978.

PRINCIPLES AND ALGORITHMS FOR CAUSAL QUALITATIVE REASONING OF GEOMETRIC OBJECTS

GE Jianxin

(State Key Laboratory of CAD&CG Zhejiang University Hangzhou 310027)

YANG Li

(National Research Center for Intelligent Computing Systems Beijing 100080)

Abstract Causal qualitative reasoning is aimed to explore the causalities between components in a system by analyzing the descriptions for physical behaviors and relationships of the system. This paper presents a model and algorithms of causal qualitative reasoning based on constraint and variable analysis. The result can be widely applied to many aspects of product design systems, such as parametric design by geometric reasoning, decomposition of complex design tasks and dependency analysis of design parameters. The algorithms have advantages in applicability, efficiency, robustness and ability to deal with under-constrained conditions in a constraint system.

Key words Qualitative reasoning, causal reasoning, geometric reasoning, constraint satisfaction, parametric design.

Class number TP391