数式処理 *J.JSSAC (2003)*
Vol. 9, No. 4, pp. 36 - 51

論文

多項式制約間の矛盾検出法

沢田 浩之*
Hiroyuki Sawada

産業技術総合研究所

(Received 2002/1/15 Revised 2003/1/27)

Abstract

There have been many attempts to formalize a practical problem, including a design problem, as a constraint satisfaction problem, and to solve it by applying generic constraint solving methods. In this approach, it is quite important issue to detect conflicts between the defined constraints. However, conventional methods such as the constraint propagation have a difficulty in finding out conflicts between non-linear constraints completely.

This paper proposes a new algebraic method of detecting conflicts between polynomial equations and inequalities. This new method does not fail to find out an existing conflict. This provides a user with useful information to identify the cause of the conflict and to resolve it.

1 はじめに

機械設計などの現実世界の問題を制約充足問題として定式化し、これをより汎用的な制約解消手法を用いることによって解決するという方法は、1990 年頃から数多く研究されてきている [2, 8, 14, 15]。これらの研究アプローチでは、まず、現実世界の問題を、論理式や数式を用いて、満足すべき制約条件の集合として記述する。そして、制約伝播 [1, 15]、制約論理プログラミング [7]、適応探索 [6, 10] などの手法による解析を行い、その後、適切な解を求めることが通常である。

ここで行われる解析において、制約条件相互の矛盾検出、すなわち、同時に成立させることができない条件を特定することは、重要な作業と位置付けられる。例えば、機械設計においてそのような矛盾を検出することは、設計者に対して異なる要求仕様間のトレードオフを認識させるとともに、問題に対する新たな知見を与え、より良い解を得るための出発点となるものとして [4, 9]、広く認められている。

*h.sawada@aist.go.jp

従来、そのような矛盾を検出する手段として、制約伝播が多く用いられている。しかしながら、制約伝播は健全ではあるが完全ではないことがすでによく知られており [1]、存在する矛盾を検出できない場合がある。特に、「個々の制約条件を単独で満足させることは可能でありながら、それらを同時に満足させることは不可能であるような状況」を検出することはできない。そのような状況を検出することを目的として、著者らは簡約木を用いた矛盾検出法を提案した [13]。この方法は、同時に満足させることのできない複数の制約条件を特定することはできるものの、それは矛盾が存在するための十分条件に過ぎず、やはり完全ではない。完全性の保証された矛盾検出方法としては、Quantifier Elimination (QE) [3] が知られている。だが、QE のみでは、矛盾が存在するか否かを確認することはできるものの、具体的にどの条件が互いに矛盾しているのかを特定するには至らない。

本論文では、与えられた多項式方程式および不等式集合の中から、互いに矛盾する条件、すなわち、同時に成立させることができない不等式およびそれら不等式間の矛盾に関わりを持つ方程式を検出する方法を提案する。本手法では、まず、スラック変数を導入することにより、すべての不等式を方程式に変換する。この変換によって、各不等式が示す条件は、スラック変数の値域として表現される。次に、グレブナ基底を計算することによって、スラック変数のみに関する連立方程式を導き、その連立方程式の実数解の有無を検証することによって、もとの不等式相互の矛盾を検出する。そして、そのスラック変数のみからなる連立方程式を導くのに必要な方程式を特定することにより、不等式間の矛盾に関わりを持つ方程式を検出する。

本論文の構成は以下の通りである。まず、第 2 章において問題の設定を行う。特に、制約条件集合における矛盾の検出が、同時に成立しない不等式を特定することであるとの主張を述べる。次に、第 3 章で、本論文で提案する矛盾検出法について説明する。第 4 章では、本手法を応用した例を、ロボットアーム設計問題を用いて示す。

なお、本論文では、制約条件はすべて多項式の方程式あるいは不等式であるとしている。

2 問題の設定

本章では、問題の設定を行うとともに、多項式制約条件集合における矛盾とは何を意味するのかについて述べる。

与えられた多項式制約集合を式 (1) とする。

$$\begin{aligned} &f_1(\boldsymbol{x}) = 0, \ldots, f_p(\boldsymbol{x}) = 0, \\ &g_1(\boldsymbol{x}) \neq 0, \ldots, g_q(\boldsymbol{x}) \neq 0, \\ &h_1(\boldsymbol{x}) \geq 0, \ldots, h_r(\boldsymbol{x}) \geq 0. \end{aligned} \tag{1}$$

ここで、$\boldsymbol{x} = (x_1, \ldots, x_n)$ は n 次元実数空間における変数、$f_i(\boldsymbol{x}), g_j(\boldsymbol{x}), h_k(\boldsymbol{x})$ は実数係数の多項式である。また、境界を含まない不等式制約 $e(\boldsymbol{x}) > 0$ ($e(\boldsymbol{x})$ は実数係数の多項式) が与えられた場合、それは $e(\boldsymbol{x}) \neq 0 \wedge e(\boldsymbol{x}) \geq 0$ として扱われるものとする。

本論文で扱う問題は、次のように定義される。

問題 1

1. 式 *(1)* に含まれる不等式の中から、同時に満足させることのできない不等式を検出する。

2. 矛盾が検出された場合、その矛盾の発生に関わりのある方程式を式 *(1)* から検出する。

この問題を解くために、まず、スラック変数 $\boldsymbol{s} = (s_1, \ldots, s_q)$ および $\boldsymbol{t} = (t_1, \ldots, t_r)$ を導入することにより、式 (1) を次式 (2) および (3) に置き換える。

$$\begin{aligned} &f_1(\boldsymbol{x}) = 0, \ldots, f_p(\boldsymbol{x}) = 0, \\ &g_1(\boldsymbol{x}) \cdot s_1 = 1, \ldots, g_q(\boldsymbol{x}) \cdot s_q = 1, \\ &h_1(\boldsymbol{x}) = t_1, \ldots, h_r(\boldsymbol{x}) = t_r. \end{aligned} \tag{2}$$

$$t_1 \geq 0, \ldots, t_r \geq 0. \tag{3}$$

このような変換により、式 (1) に含まれる不等式はすべて方程式に変換される。また、これらの不等式によって表現されていた条件は、式 (3) で示されるスラック変数 $t_1, \ldots, t_r$ の値域に置き換えられる。

図 1 に、この変換の持つ幾何学的意味を示す。与えられた制約条件集合である式 (1) は、$\boldsymbol{x}$-

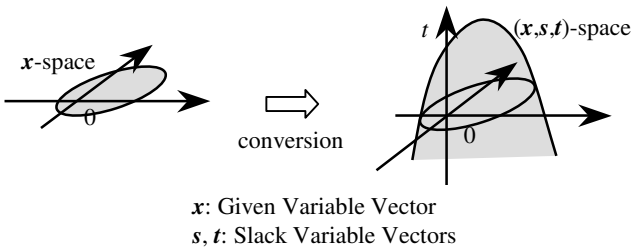

図 1: スラック変数導入による制約条件集合の変換が持つ幾何学的意味

空間内の領域を示す。スラック変数 $\boldsymbol{s}$ および $\boldsymbol{t}$ を導入することにより、この領域は $(\boldsymbol{x}, \boldsymbol{s}, \boldsymbol{t})$-空間内の代数曲面に変換される。一方、各不等式によって表されていた条件は、$(\boldsymbol{x}, \boldsymbol{s}, \boldsymbol{t})$-空間内の $t_1 \geq 0, \ldots, t_r \geq 0$ なる領域として表現される。つまり、式 (1) で定義される制約条件集合は、式 (2) で表される代数曲面のうち、式 (3) を満足する部分曲面に変換されることになる。

このことから、式 (1) に矛盾が存在するということ、すなわち式 (1) が解を持たないということは、式 (2) で表される代数曲面が、$(\boldsymbol{x}, \boldsymbol{s}, \boldsymbol{t})$-空間において、式 (3) なる領域を通らないことを意味するといえる。換言すると、式 (2) なる条件のもとで式 (3) を満足させることはできないということである。したがって、式 (1) の矛盾を検出することは、式 (2) なる条件のもとで、式 (3) の中から同時に満足させることのできない不等式を特定することに他ならないといえる。

次章では、本論文で提案する矛盾検出法について述べる。

3 矛盾検出法

3.1 不等式間の矛盾検出

スラック変数 $t_1, \ldots, t_r$ は、式 (3) を満足しなくてはならない。したがって、不等式間の矛盾は、$t_1, \ldots, t_r$ のみからなる連立方程式 (4) を導出することにより検出可能である。

$$\begin{aligned} c_1(t_{a_j(i,1)}, \ldots, t_{a_j(i,j)}) &= 0, \\ \vdots \quad \vdots \quad \vdots & \\ c_k(t_{a_j(i,1)}, \ldots, t_{a_j(i,j)}) &= 0. \end{aligned} \tag{4}$$

ただし、$\{a_j(i,1), \ldots, a_j(i,j)\}$ は 1 から r までの整数のうちの j 個の整数からなる組み合わせであり、全部で ${}_rC_j$ 通りの組み合わせが存在する。i はこれらの組み合わせを区別するために付けられる添字であり、1 から ${}_rC_j$ までの整数値をとる。

矛盾検出の規準

$\{c_1(t_{a_j(i,1)}, \ldots, t_{a_j(i,j)}) = 0, \ldots, c_k(t_{a_j(i,1)}, \ldots, t_{a_j(i,j)}) = 0, t_{a_j(i,1)} \geq 0, \ldots, t_{a_j(i,j)} \geq 0\}$ が解を持たないならば、式 (2) のもとで不等式 $t_{a_j(i,1)} \geq 0, \ldots, t_{a_j(i,j)} \geq 0$ のすべてを同時に満足させることはできない。スラック変数の定義式 (2) により、これは式 (1) の中の不等式 $h_{a_j(i,1)}(\boldsymbol{x}) \geq 0, \ldots, h_{a_j(i,j)}(\boldsymbol{x}) \geq 0$ を同時に満足させられないことを意味する。

本論文で提案する方法は、$\{a_j(i,1), \ldots, a_j(i,j)\}$ のすべての可能な組み合わせについて $t_{a_j(i,1)}, \ldots, t_{a_j(i,j)}$ からなる方程式集合を導出し、上述の規準に従って不等式間の矛盾を検出するものである。図 2 に、矛盾検出の作業手順を示す。

図 2 に示す作業手順では、まず、不等式全体に関する矛盾の有無を検査する。この時点で矛盾が検出されない場合、与えられた制約条件集合には矛盾が存在しないことになるので、空集合が返される。一方、この時点で矛盾が検出された場合には、同時に満足させることのできない不等式を特定するための検査ループが開始され、不等式の組み合わせそれぞれについて矛盾の有無が検査される。この検査ループは矛盾が検出された時点で終了し、スラック変数のみからなる矛盾を示す連立方程式 (4) が返される。

ここで返される連立方程式は最初に見つかった矛盾を示すものにすぎず、これ以外にも矛盾が存在する可能性は残されている。しかしながら、実際の問題解決における矛盾解消作業は、ステップ・バイ・ステップで行われることが通常である。したがって、最初に見つかった矛盾のみを示すことにも、十分な意義があるものと考えられる。

式 (4) は、式 (2) のグレブナ基底を計算することによって求められる。その際、計算に用いられる項順序は次の条件を満足する必要がある。

> 変数 $t_{a_j(i,1)}, \ldots, t_{a_j(i,j)}$ の順位は、辞書式順序のもとで、他のどの変数よりも下位である。

$\{c_1(t_{a_j(i,1)}, \ldots, t_{a_j(i,j)}) = 0, \ldots, c_k(t_{a_j(i,1)}, \ldots, t_{a_j(i,j)}) = 0, t_{a_j(i,1)} \geq 0, \ldots, t_{a_j(i,j)} \geq 0\}$ が解を持つか否かを判定する方法としては、QE のほか、この問題を制約条件付最小値問題

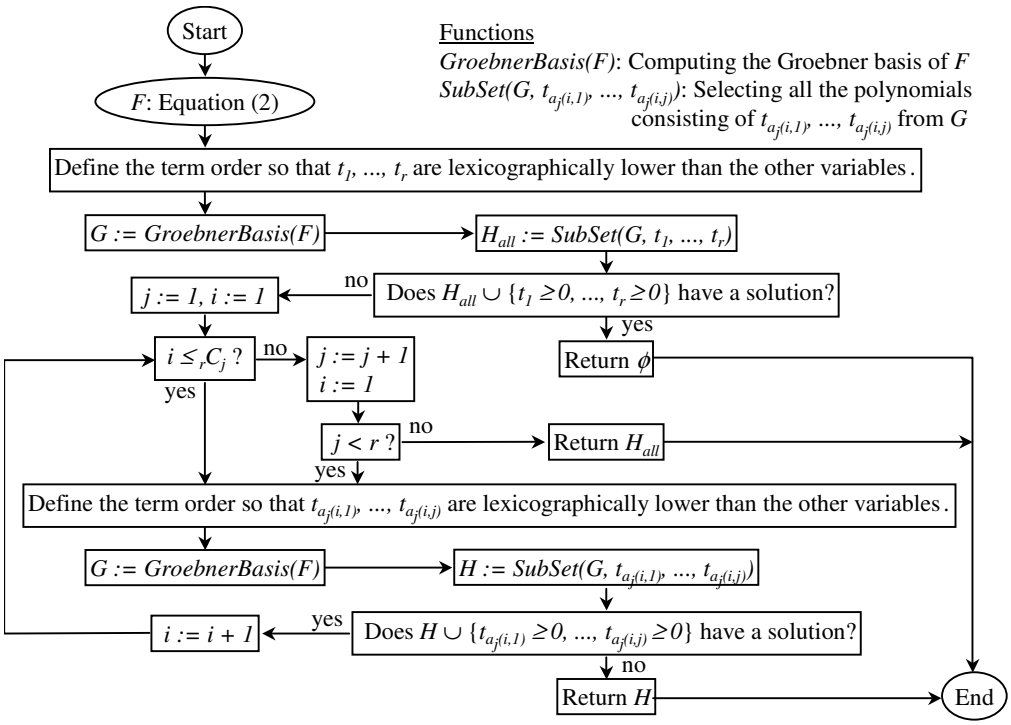

図 2: 矛盾検出の作業手順

に帰着させる方法 [12] が提案されている。現在のところ、変数の数が多い場合、一般に QE の計算効率は非常に悪い。実際、本論文の第 4 章で取り上げているアームロボット設計問題に QEPCAD[1)] [5] を適用してみたところ、メモリ不足により解くことができなかった。そのため、ここでは文献 [12] の方法を用いている (付録参照)。

3.2 矛盾と関わりのある方程式の検出

矛盾が検出された場合、その原因を特定することは実際の問題解決においてきわめて重要である。本節では、検出された不等式間の矛盾と関わりを持つ方程式を特定する方法について述べる。

式 (4) が $t_{a_j(i,1)} \geq 0, \ldots, t_{a_j(i,j)} \geq 0$ なる解を持たないとする。また、与えられた制約条件集合 (1) に含まれるある方程式を式 (5) とする。

$$f_k(\boldsymbol{x}) = 0. \tag{5}$$

式 (4) を導くために式 (5) が必要であるならば、式 (5) は式 (4) によって示される矛盾と関わりがあることになる。一方、式 (5) が式 (4) を導くために不要であるならば、式 (5) はこの矛

[1)] http://www.cs.usna.edu/~qepcad/B/QEPCAD.html

盾と関わりはない。

式 (5) が式 (4) を導くために必要か不要かは、以下の方法で調べることができる。式 (2) から式 (5) を除いた連立方程式によって定義されるイデアルを I とする。式 (4) に現れる多項式 $c_1(t_{a_j(i,1)},\ldots,t_{a_j(i,j)}),\ldots,c_k(t_{a_j(i,1)},\ldots,t_{a_j(i,j)})$ のうち少なくとも 1 つが I に属していないならば、式 (5) は式 (4) を導くために必要であると結論される。これに対して、$c_1(t_{a_j(i,1)},\ldots,t_{a_j(i,j)}),\ldots,c_k(t_{a_j(i,1)},\ldots,t_{a_j(i,j)})$ のすべてが I に属しているならば式 (4) は式 (5) なしでも導けることになり、式 (5) は不要であると結論される。したがって、I のグレブナ基底 G を用いることにより、式 (5) が式 (4) で示される矛盾と関係があるかないかを判定することができる。

式 (4) で示される矛盾と式 (5) との関係

式 (2) から式 (5) を取り除いた連立方程式によって定義されるグレブナ基底を G とする。

1. G が $c_1(t_{a_j(i,1)},\ldots,t_{a_j(i,j)}),\ldots,c_k(t_{a_j(i,1)},\ldots,t_{a_j(i,j)})$ のうち少なくとも 1 つの多項式を 0 に簡約化しない。
   $\Rightarrow$ 式 (5) は式 (4) で示される矛盾と関わりがある。

2. G が $c_1(t_{a_j(i,1)},\ldots,t_{a_j(i,j)}),\ldots,c_k(t_{a_j(i,1)},\ldots,t_{a_j(i,j)})$ のすべての多項式を 0 に簡約化する。
   $\Rightarrow$ 式 (5) は式 (4) で示される矛盾と関わりがない。

3.3 提示すべき情報–不等式ならびに方程式の持つ工学上の意味

本節では、設計問題等の現実世界の問題において不等式ならびに方程式が意味するところについて考察し、それに基づいて、矛盾解消のために有用な情報、すなわち問題解決に携わる作業者に対して提示すべき情報は何であるかについて述べることにする。

方程式は 2 つのグループに分けることができる。1 つは変数への数値の割り当てを示すものであり、1 つは明確に限定された変数間の関係を示すものである。変数への数値割り当ては、1 変数線形方程式として表される。例えば、「変数 x_k に数値 5 を割り当てる」ことは、「$x_k = 5$」という方程式として表現される。現実世界の問題では、ある変数に対して様々な値を割り当ててみることによって最適な解を探し出すという方法を採ることが多い。これに対して、多変数あるいは非線形の方程式は変数間の確固たる関係を示すものと考えられ、通常、変更や修正は不可能なものである。例えば、直流モータの特性を示す方程式「$V = RI + k\omega$」は、入力電圧 V、入力電流 I、内部抵抗 R、トルク定数 k、回転速度 ω の間の明確に定義された関係を示すものであり、変更は不可能である。

一方、不等式は、ある条件の許容範囲や許容誤差を示すことが通常である。例えば、w がある製品の重量を示す変数だとした場合、「$w < 10$ [kg]」という不等式は、その製品に許される重量の範囲を示すことになる。現実世界の問題解決では、この種の許容範囲や許容誤差は、他の制約条件との兼ね合いで緩められたり、あるいは厳しくされたりすることが往々にして起こる。

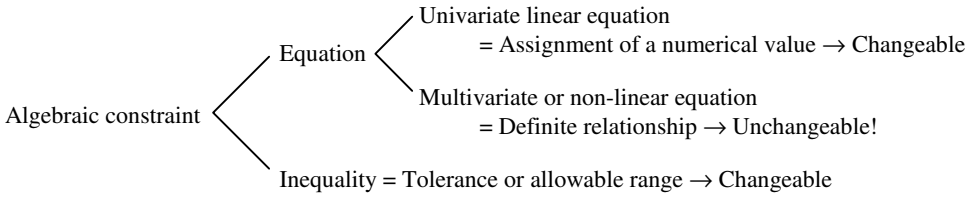

図 3: 代数制約条件の工学的意味に基づく分類

図 3 は、方程式および不等式のこのような分類を示したものである。問題解決にあたる作業者にとって必要なものは、現在の解を修正するための情報である。したがって、変更不可能な制約条件に関する情報は、変更可能なものに比べ、その価値はきわめて低いといえる。したがって、作業者に対して提示すべき情報は、同時に満足させることができない不等式の集合ならびにそれと関係する変数の値であると結論することができる。

4 例題

本章では、本論文で提案する矛盾検出法を応用した例を、図 4 のようなワイヤ駆動式 2 関節アームロボットの設計問題 [13] を用いて示す。実行されるオペレーション、ロボットの構成、仕様、オプションを以下に示す。

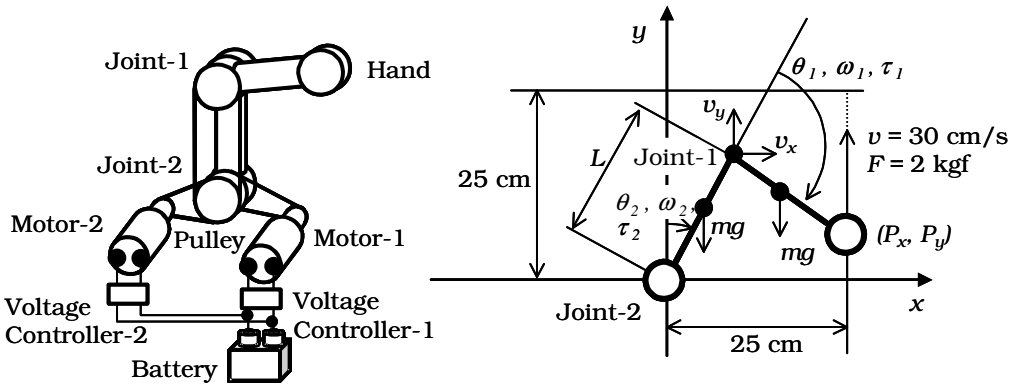

図 4: ワイヤ駆動式 2 関節アームロボット

オペレーション 質量 2[kg] のものを、速度 30 [cm/s] で持ち上げる。動作範囲は、$x = 25$ [cm]、0 [cm] $\leq y \leq 25$ [cm] である。動作は準静的であるとし、動力学は考慮しないものとする。

構成 ロボットは 2 つのリンク、2 つの直流モーター、2 つの電圧コントローラ、1 つの滑車、1 つの電源、および駆動ワイヤから構成される。Motor-1、Motor-2 はそれぞれ Joint-1、

Joint-2 を独立に回転させる。Motor-1 が回転すると、Joint-2 の位置に設置してある滑車が回転し、ワイヤを介して Joint-1 を回転させる。Motor-2 は Joint-2 と直接ワイヤで結び付けられている。これらのモーターには減速のための遊星歯車が取り付けられる。モーターの制御は、それぞれの電圧コントローラによって入力電圧をコントロールすることによって行われる。

簡単のため、2 つのモーターは同機種、取り付けられる遊星歯車の減速比も同じとする。

仕様 ロボットの仕様は以下の通りである。

リンク長さ: 20 [cm]

リンク質量: 1 [kg]

最大供給電流: 1.5 [A]

関節回転角: Joint-1 の回転角 θ_1 は 0 度以上 180 度以下であり、逆方向には曲がらない。

オプション 選択可能な減速比およびモーターのオプションは以下のとおりである。

減速比: 246:1、415:1。

選択可能なモーター:

モーター名	M2342	M3557
定格電圧 [V]	12	12
トルク定数 [mNm/A]	13.4	23.4
内部抵抗 [Ω]	1.9	2.4

ここでは、本手法を用いて、矛盾が生じるオプションの組み合わせを検出する例を示す。この問題で用いられる変数のリストを以下に記す。

L, m**:** リンクの長さおよび質量

P_x, P_y**:** ロボットハンドの x 座標および y 座標

F, v**:** ロボットハンドの y 方向の力および速度

v_x, v_y**:** Joint-1 の x 方向および y 方向速度

θ_1, θ_2**:** Joint-1 および Joint-2 の回転角

τ_1, τ_2**:** Joint-1 および Joint-2 に働くトルク

ω_1, ω_2**:** Joint-1 および Joint-2 の角速度

k, R**:** モーターのトルク定数および内部抵抗

ρ**:** 遊星歯車の減速比

V_1, V_2**:** Motor-1 および Motor-2 の入力電圧

I_1, I_2**:** Motor-1 および Motor-2 の入力電流

g**:** 重力加速度

制約条件集合は、以下の式 (6)〜(9) と、選択可能なオプションとの組み合わせとして与えられる。

幾何拘束条件

$$\begin{aligned} P_x &= L(\sin\theta_2 + \sin\theta_1\cos\theta_2 + \cos\theta_1\sin\theta_2), \\ P_y &= L(\cos\theta_2 + \cos\theta_1\cos\theta_2 - \sin\theta_1\sin\theta_2), \\ v &= v_y - L\omega_1(\sin\theta_1\cos\theta_2 + \cos\theta_1\sin\theta_2), \\ v_x &= -L\omega_1(\cos\theta_1\cos\theta_2 - \sin\theta_1\sin\theta_2), \\ v_y &= -L\omega_2\sin\theta_2, \\ v_x &= L\omega_2\cos\theta_2. \end{aligned} \tag{6}$$

力・トルクの釣合条件

$$\begin{aligned} \tau_1 &= -(F + \frac{1}{2}mg)L(\sin\theta_1\cos\theta_2 + \cos\theta_1\sin\theta_2), \\ \tau_2 &= -(F + \frac{3}{2}mg)L\sin\theta_2. \end{aligned} \tag{7}$$

直流モーターにおける入出力関係

$$\begin{aligned} &\tau_1 = \rho k I_1, V_1 = RI_1 + \rho k\omega_1, \\ &\tau_2 = \rho k I_2, V_2 = RI_2 + \rho k\omega_2. \end{aligned} \tag{8}$$

与えられた数値データ

$$\begin{aligned} &L = 0.2\ [\mathrm{m}], m = 1\ [\mathrm{kg}], F = 19.6\ [\mathrm{N}], \\ &v = 0.3\ [\mathrm{m/sec}], g = 9.8\ [\mathrm{m/sec^2}], P_x = 0.25\ [\mathrm{m}]. \end{aligned} \tag{9}$$

変数の許容範囲

$$\begin{aligned} \sin\theta_1 &\geq 0, \\ 0\ [\mathrm{m}] \leq P_y &\leq 0.25\ [\mathrm{m}], \\ -1.5\ [\mathrm{A}] \leq I_1 + I_2 &\leq 1.5\ [\mathrm{A}]. \end{aligned} \tag{10}$$

オプション

減速比 $\rho = 246$ または $\rho = 415$。

直流モーター 式 (11) (M2342) または式 (12) (M3557)。定格電圧に関する制約条件は共通で、式 (13) となる。

$$k = 13.4 \times 10^{-3}\ [\mathrm{Nm/A}], R = 1.9\ [\Omega]. \tag{11}$$

$$k = 23.4 \times 10^{-3}\ [\mathrm{Nm/A}], R = 2.4\ [\Omega]. \tag{12}$$

$$\begin{aligned} -12\ [\mathrm{V}] \leq V_1 &\leq 12\ [\mathrm{V}], \\ -12\ [\mathrm{V}] \leq V_2 &\leq 12\ [\mathrm{V}]. \end{aligned} \tag{13}$$

これらの制約条件の中には三角関数が含まれているが、本手法が対象としている制約条件は代数制約のみであり、三角関数を直接扱うことはできない。そこで、矛盾検出に先立ち、以下に示す変数変換および制約条件の追加を行うことによって、すべての制約条件を代数制約に置き換える必要がある。

変数変換

$$\begin{aligned} sin_1 &= \sin\theta_1,\ cos_1 = \cos\theta_1, \\ sin_2 &= \sin\theta_2,\ cos_2 = \cos\theta_2. \end{aligned} \tag{14}$$

追加される制約条件

$$\begin{aligned} sin_1^2 + cos_1^2 &= 1, \\ sin_2^2 + cos_2^2 &= 1. \end{aligned} \tag{15}$$

次に、スラック変数を導入することによって、不等式を方程式に変換する。

式 (10) の不等式の変換

$$\begin{aligned} sin_1 \geq 0 &\Rightarrow sin_1 = t_1,\ t_1 \geq 0. \\ P_y \geq 0 &\Rightarrow P_y = t_2,\ t_2 \geq 0. \\ P_y \leq 0.25 &\Rightarrow -P_y + 0.25 = t_3,\ t_3 \geq 0. \\ I_1 + I_2 \geq -1.5 &\Rightarrow I_1 + I_2 + 1.5 = t_4,\ t_4 \geq 0. \\ I_1 + I_2 \leq 1.5 &\Rightarrow -I_1 - I_2 + 1.5 = t_5,\ t_5 \geq 0. \end{aligned} \tag{16}$$

式 (13) の不等式の変換

$$\begin{aligned} V_1 \geq -12 &\Rightarrow V_1 + 12 = t_6,\ t_6 \geq 0. \\ V_1 \leq 12 &\Rightarrow -V_1 + 12 = t_7,\ t_7 \geq 0. \\ V_2 \geq -12 &\Rightarrow V_2 + 12 = t_8,\ t_8 \geq 0. \\ V_2 \leq 12 &\Rightarrow -V_2 + 12 = t_9,\ t_9 \geq 0. \end{aligned} \tag{17}$$

以降、2 通りの減速比と 2 通りのモーターからなる合計 4 つの組み合わせについて、図 2 の手順に従って矛盾検出作業を行う。その際に用いた計算機環境は以下の通りである。

矛盾検出に用いた計算機環境

CPU: Xeon 2.2 GHz
RAM: 4 GB
OS: Red Hat Linux 7.2
数式処理ライブラリおよびプログラム言語: Risa/Asir [11]

表 1 に、矛盾検出の結果を示す。表中の計算時間は CPU 時間と GC 時間を合計したものであり、矛盾と関係する変数値を特定するための時間も含まれている。

矛盾が検出された 2 つの組み合わせについて、その詳細を述べる。なお、すべての計算は、浮動小数ではなく、有理数で行われている。

1. モーター M2342 と減速比 $\rho = 246$ の組み合わせ

   制約条件集合は、式 (6)、(7)、(8)、(9)、(11)、(14)、(15)、(16)、(17)、そして $\rho = 246$ によって与えられる。図 2 の手順に従うと、まず、不等式全体に関する矛盾が検出さ

表 1: 矛盾検出の結果

	モーター M2342	モーター M3557
減速比 $\rho = 246$	矛盾あり（計算時間: 51 秒）	矛盾なし（計算時間: 42 秒）
減速比 $\rho = 415$	矛盾なし（計算時間: 40 秒）	矛盾あり（計算時間: 286 秒）

れる。詳細な検査を進めると、スラック変数 t_1, t_4 に関する 0 以上の解が存在しない方程式 (18) が得られる。

$$\begin{aligned}&2305920400000000t_1^4 + (46123223980745 61t_4^4 + 269250826266922 68t_4^3 + \\ &47323966257847044t_4^2 + 23435864640798912t_4 - \\ &6017562585485184)t_1^2 + 10191356780062500t_4^2 + \\ &29746741413375000t_4 + 21706349895562500 = 0.\end{aligned} \tag{18}$$

これにより、$t_1 \geq 0$ と $t_4 \geq 0$ は同時に成立しないと結論される。

さらに第 3.2 節の方法を適用することにより、このオプションの組み合わせには、全部で以下の制約条件の間に矛盾が存在することが分かる。

$$\begin{aligned}&\sin\theta_1 \geq 0, I_1 + I_2 \geq -1.5, P_x = 0.25, F = 19.6, L = 0.2, \\ &m = 1, g = 9.8, k = 13.4 \times 10^{-3}, \rho = 246.\end{aligned} \tag{19}$$

2. モーター M3557 と減速比 $\rho = 415$ の組み合わせ

制約条件集合は、式 (6)、(7)、(8)、(9)、(12)、(14)、(15)、(16)、(17)、そして $\rho = 415$ によって与えられる。図 2 の手順に従うと、まず、不等式全体に関する矛盾が検出される。詳細な検査を進めると、スラック変数 t_1, t_2, t_6 に関する 0 以上の解が存在しない連立方程式 (20) が得られる。

$$\begin{aligned}&250880000t_2t_1^2 + ((1035840000t_6 - 11646080000)t_2^2 + 64740000t_6 + \\ &26548168)t_1 + 37721408400t_2^3 + 2357588025t_2 = 0, \\ &80281600t_1^3 + (331468800t_6 - 3726745600)t_2t_1^2 + (12854850688t_2^2 - \\ &76440000)t_1 - 37721408400t_2^3 + 3677837319t_2 = 0, \\ &-1024t_1^2 - 160000t_2^4 + 5600t_2^2 + 975 = 0, \\ &(-331468800t_6 + 3726745600)t_1^2 + (12544000000t_2^3 - 13293890688t_2)t_1 + \\ &37721408400t_2^2 - 3677837319 = 0, \\ &(-9834496000000t_2^2 - 1072964505600t_6^2 + 24126951014400t_6 - \\ &126167632427008)t_1^2 + (-45570132157056t_6 + 482777169513472)t_2t_1 + \\ &(122104198990800t_6 - 1517004160214400)t_2^2 - 11905159401603t_6 +\end{aligned} \tag{20}$$

$$
\begin{aligned}
&124840509956136 = 0,\\
&-3855122432000000000t_2t_1^3 + (1736593052313600000t_6^3 -\\
&5857420532520960000t_6^2 + 626928869234743296000t_6 -\\
&212407219338690355200)t_1^2 + (737552588961951360000t_6^2 -\\
&1610614543719502464000t_6 + 813457477017220888448)t_2t_1 +\\
&(-1976256460666098000000t_6^2 + 467720134234259400000t_6 -\\
&258618665051139383640000)t_2^2 + 19268500491494455500t_6^2 -\\
&4186925509949759070000t_6 + 210177458533682569449 = 0.
\end{aligned}
$$

さらに第 3.2 節の方法を適用することにより、このオプションの組み合わせには、全部で以下の制約条件の間に矛盾が存在することが分かる。

$$
\begin{aligned}
&\sin\theta_1 \geq 0, P_y \geq 0, V_1 \geq -12, P_x = 0.25, F = 19.6, v = 0.3,\\
&L = 0.2, m = 1, g = 9.8, k = 23.4 \times 10^{-3}, R = 2.4, \rho = 415.
\end{aligned}
\tag{21}
$$

5 おわりに

設計問題等の現実世界の問題には、多くの場合、潜在的に矛盾する条件が含まれている。例えば、「軽くて丈夫」という要求が満足すべき条件として与えられた場合、これは「重量が小さく、強度および剛性が高い」と解釈されることが多いが、重量を小さくすることは強度・剛性の低下を伴うことが通常である。そのため、問題解決に当たる作業者には、このような潜在的な因果関係を把握し、直接あるいは間接的に相反する条件を特定し、妥協点を探し当てることが求められる。

本論文では、現実世界の問題を代数制約からなる制約充足問題として定式化した場合、これらの問題解決過程における矛盾検出が、同時に成立しない不等式を特定することに他ならないことを述べ、そのための代数的手法を新しく提案した。そして、矛盾解消作業を支援するために、検出された矛盾と関わりを持つ方程式を特定する方法について述べた。さらに、不等式と方程式の持つ工学的意味について考察し、作業者に対して提示すべき有用な条件は、同時に成立しない不等式の集合とそれと関係する変数の値であることを示した。

従来手法と比較した場合の本手法の利点、そして現状と課題について以下に論じる。

本手法の利点

- 矛盾の検出を厳密に行うことが可能である。
  従来多く用いられている制約伝播法では、個々の不等式を満足させられるかどうかを検査することしかできない。しかしながら、現実には、不等式 A、不等式 B をそれぞれ満足させることはできるが、それらを同時に満足させることはできない、といった状況が多々ある。この類の矛盾を制約伝播法で検出することは不可能である。また、著者らが過去に提案した簡約木を用いる方法 [13] では、この種の矛盾を検出することはできるものの、それは矛盾が存在するための十分条件に過ぎず、存在している矛盾を

検出できない場合がある。だが、本手法では矛盾の検出を厳密に行うことが可能である。その具体的な方法および現時点における課題については、後述の「現状と課題」における「実数解を厳密に扱うための方法」の項で説明する。

- デッドロックが生じない。
  制約伝播法は、構成される制約ネットワークに沿って、変数の値を順次伝播させていく方法である。したがって、制約ネットワークに閉路が存在する場合、デッドロックが生じて解析を進められなくなることがある。一方、本手法ではそのようなデッドロックは生じない。

現状と課題

本手法の大きな課題として計算効率が挙げられる。計算効率に大きく関わると考えられる各項目について述べる。

- 消去イデアルの計算
  図 2 の手順に従えば、スラック変数 $t_1, \ldots, t_r$ のすべての組み合わせについて消去イデアルを計算することになるため、最悪の場合 2^r 回の消去イデアル計算が必要となる。これらの各消去イデアル計算には、一般に、グレブナ基底計算を用いることになり、その計算効率は決してよくない。

- 消去イデアルの無矛盾性判定
  消去イデアルの無矛盾性の判定、すなわち式 $(4) \cup \{t_{a_j(i,1)} \geq 0, \ldots, t_{a_j(i,j)} \geq 0\}$ が解を持つか否かの判定には、この問題を制約条件付最小値問題に帰着させる方法 [12] を用いている (付録参照)。文献 [12] で提案されている方法では、その制約条件付最小値問題を厳密に解くために、グレブナ基底計算を繰り返し行うことが述べられている。これもまた、計算効率上きわめて不利といえる。

- 矛盾と関わりのある 1 変数線形方程式の検出
  第 3.2 節および第 3.3 節の記述に従えば、矛盾と関わりのある 1 変数線形方程式を検出するためには、その数だけグレブナ基底計算を行なう必要がある。これもまた、多くの計算コストを要すると考えられる。

- 実数解を厳密に扱うための方法
  本論文で述べた手続のみでは、変数 $\boldsymbol{x}$ が実数であることを保証できない。すなわち、本論文で述べた手順に従って「矛盾なし」と結論されても、厳密には、それは $\boldsymbol{x}$ が複素数の場合には無矛盾であることを言うに過ぎず、$\boldsymbol{x}$ が実数の場合に無矛盾であることは保証しない。例として、式 (22) が与えられた場合を考える。

$$x^2 + y^2 + 1 \leq 0. \tag{22}$$

  スラック変数 t の導入により、式 (22) は式 (23) に変換される。

$$x^2 + y^2 + t + 1 = 0,\ t \geq 0. \tag{23}$$

式 (22) には実数解が存在しないにもかかわらず、スラック変数 t のみからなる方程式集合を導くことはできないため、本論文で述べた手順に従った場合、「矛盾なし」と結論される。

変数 $\boldsymbol{x}$ が実数であることを保証するためには、与えられた制約条件集合に、変数 $\boldsymbol{x}$ が実数であるための必要十分条件、すなわち各変数の自乗が 0 以上であるという条件を付け加える必要がある。式 (22) の場合、$x^2 \geq 0$、$y^2 \geq 0$ の 2 つの条件を追加し、それに対応する 2 つのスラック変数 t_x、t_y を導入することによって、式 (24) へ変換する。

$$x^2 + y^2 + t + 1 = 0,\ x^2 = t_x,\ y^2 = t_y,\ t \geq 0,\ t_x \geq 0,\ t_y \geq 0. \tag{24}$$

これにより、スラック変数のみからなる方程式 $t_x + t_y + t + 1 = 0$ が導かれ、矛盾が検出される。

このようにして、原理的には変数 $\boldsymbol{x}$ を実数として厳密に扱うことが可能ではあるものの、これは計算効率を著しく低下させる。実際、第 4 章で取り上げたアームロボット設計問題について上述の方法を試みたところ、1 日かけても答えが返ってこなかった。

このように、本手法を現実の大規模な問題に適用するために解決しなくてはならない課題はきわめて多い。そして、ここに挙げた計算効率の問題は、本手法に限らず、数式処理の応用全般に関わる問題だと考えられる。数式処理の特長は、その厳密性にある。一方、現実世界で扱われる問題には必ず誤差が含まれており、その誤差の範囲内で正しいことが言えれば、厳密な正しさを保証する必要はないという状況があり得ることも事実である。

本論文では、入力される多項式集合には誤差が含まれていないとの前提で、すべての計算を数式処理的に扱った。しかしながら、現実世界の問題を扱う場合に求められる機能や性能を考えた場合、厳密性を多少犠牲にしてでも計算効率の向上を図る必要がある。その具体的な手段として考えられるのは、数値計算の部分的な導入である。例えば、上に挙げた計算効率に関わる項目のうち、「消去イデアルの無矛盾性判定」で行なわれる制約条件付最小値問題の解法として数値計算を用いれば、かなりの効率向上が期待できる。その場合、行われた数値計算の正しさをどのようにして保証するのかが問題となる。数値計算の部分的導入による計算効率の向上とその正しさの保証方法、これらが今後の課題の本質と言えるだろう。

謝 辞

本研究にあたり、適切な助言を頂いた産業技術総合研究所知能システム研究部門副部門長の小鍛治繁博士に深く感謝します。

参 考 文 献

[1] Bowen, J.: Aspect of Constraint Processing, *AI in Engineering*, **6**(2), 1991, 100-102.

[2] Bowen, J. and Bahler, D.: Frames, Quantification, Perspectives, and Negotiation in Constraint Networks for Life-Cycle Engineering, *AI in Engineering*, **7**(4), 1992, 199–226.

[3] Caviness, B. F. and Johnson, J. R.: Quantifier Elimination and Cylindrical Algebraic Decomposition, Springer–Verlag, Wien, 1996.

[4] Candy, L., Edmonds, E. A. and Patrick, D. J.: Interactive Knowledge Support to Conceptual Design, *AI System Support for Conceptual Design* (Sharpe, J. E. E., ed.), Springer–Verlag, London, 1996, 260–278.

[5] Collins, G. E. and Hong, H.: Partial Cylindrical Algebraic Decomposition for Quantifier Elimination, *J. Symb. Comput.*, **12**(3), 1991, 299–328.

[6] Goldberg, D. E.: Genetic Algorithms in Search, Optimization, and Machine Learning, Addison–Wesley, 1989.

[7] Jaffar, J., Michaylov, S., Stuckey, P. J. and Yap, R. H. C.: The CLP(R) Language and System, *ACM Transactions on Programming Languages and Systems*, **14**(3), 1992, 339–395.

[8] Kuokka, D. and Livezey, B.: A Collaborative Parametric Design Agent, *Proc. AAAI '94*, **1**, 1994, 387–393.

[9] Oh, V. and Sharpe, J. E. E.: Conflict Management in an Interdisciplinary Design Environment, *AI System Support for Conceptual Design* (Sharpe, J. E. E., ed.), Springer–Verlag, London, 1996, 298–318.

[10] Pham, D. T. and Karaboga, D.: Intelligent Optimization Techniques, Springer–Verlag, 1999.

[11] 齋藤友克, 竹島卓, 平野照比古: 日本で生まれた数式処理ソフト　リサアジールガイドブック, SEG 出版, 東京, 1998.

[12] 沢田浩之: 数式処理的手法に基づく過少代数制約問題の数値解法, 情報処理学会論文誌, **40**(5), 1999, 2314–2324.

[13] 沢田浩之, Yan, X.-T.: 制約ベース型初期設計支援システム, 数式処理, **8**(2), 2001, 19–35.

[14] Thornton, A. C. and Johnson, A. L.: CADET: A Software Support Tool for Constraint Processes in Embodiment Design, *Research in Engineering Design*, **8**(1), 1996, 1–13.

[15] Young, R. E., Greef, A. and O'Grady, P.: SPARC: An Artificial Intelligence Constraint Network System for Concurrent Engineering, *AI in Design '91* (Gero, J. S. ed.), Butterworth–Heinemann Ltd., 1991, 79–94.

付録

与えられた多項式制約集合 (25) が解を持つか否かを判定するために、この問題を制約条件付最小値問題に帰着させる方法 [12] について、その概略を述べる。

$$\phi_1(z_1,\ldots,z_\eta)=0,\ldots,\phi_\zeta(z_1,\ldots,z_\eta)=0, z_1\geq 0,\ldots,z_\eta\geq 0. \tag{25}$$

式 (25) で表される領域を A とする。

$$A = \{(z_1, \ldots, z_\eta) | \phi_1(z_1, \ldots, z_\eta) = 0, \ldots, \phi_\zeta(z_1, \ldots, z_\eta) = 0, z_1 \geq 0, \ldots, z_\eta \geq 0\}.$$

ここで、以下の性質を持つ目的関数 $u(z_1, \ldots, z_\eta)$ を導入する。

目的関数 $u(z_1, \ldots, z_\eta)$

1. 関数 $u(z_1, \ldots, z_\eta)$ は、$(z_1, \ldots, z_\eta)$ 空間において連続である。
2. 関数 $u(z_1, \ldots, z_\eta)$ は、$(z_1, \ldots, z_\eta)$ 空間において最小値を持つ。
3. $z_1, \ldots, z_\eta$ のうちの少なくとも 1 つがプラス無限大もしくはマイナス無限大となるとき、関数 $u(z_1, \ldots, z_\eta)$ はプラス無限大に発散する。

領域 A の境界はすべて A に含まれているので、A が空集合でなければ、関数 $u(z_1, \ldots, z_\eta)$ は領域 A において最小値を持つ。したがって、領域 A における関数 $u(z_1, \ldots, z_\eta)$ の最小値を計算することにより、式 (25) の解がその最小点の座標値として求められる。このようにして、式 (25) の解の有無を判定する問題は、制約条件付最小値問題に帰着される。関数 $u(z_1, \ldots, z_\eta)$ が領域 A において最小値を持たなければ、それは A が空集合であることを意味し、式 (25) は解を持たない。

文献 [12] では、目的関数 $u(z_1, \ldots, z_\eta)$ として以下のような 2 次関数を使っている。

$$u(z_1, \ldots, z_\eta) = z_1^2 + \ldots + z_\eta^2.$$

また、関数 $u(z_1, \ldots, z_\eta)$ の最小値の計算には Lagrange 乗数法を用いている。