

過小代数制約評価系とその機械設計への応用

機械技術研究所 沢田浩之

Abstract. 過小代数制約評価系とその設計ツールとしての有用性について述べる. 一般に, 設計問題は過小制約問題と考えられ, 解は一意に決定しない. そのような設計問題の解決を支援するために, 本評価系では, 1) 制約間の矛盾の検出, 2) 互いに矛盾する制約の提示, 3) 新しい制約の導出, 4) 数値解の例示, の4つの機能が提供される. これらの結果にもとづき, 設計者は制約条件の緩和や設計パラメータ値の決定を行う. 本評価系により, 存在する解を見落とす可能性や存在しない解を探すことに時間を費やす可能性を排除することが可能となる.

1. はじめに

一般に, 設計問題が制約充足問題として定式化可能であることはよく知られており, 多くの場合, それは過小制約問題となる. 過小制約問題とは, 決定すべきパラメータの数に比べて制約条件の数が不足しているために解が一意に決定しない問題のことを指す. 過小制約問題である設計問題は解決の見通しが悪く, 通常, 過去の設計事例や設計者個人の経験や勘にもとづいてパラメータの値を決定することにより解決されている. したがって, 存在する解を見落とす可能性や存在するはずのない解を探し求めることに労力を費やしてしまう可能性は免れない.

本稿では, 過小制約を適切に評価することによってそれらの可能性を排除し, 設計作業の支援を行う過小代数制約評価系について述べる. まず第2章で, 本評価系の機能, 構成およびアルゴリズムについて述べる. その後第3章において設計問題への適用例を示す. なお, 本稿では実数解のみを対象としている.

2. 過小代数制約評価系

2.1. 支援する設計段階

設計過程は大きく概念設計, 基本設計, 詳細設計の3段階に分類される. 概念設計では, 設計対象の概念的な構成が決定される. 基本設計では, 個々の概念設計案について, 機能を記述するために必要な基本的パラメータの値が決定される. 詳細設計では, 加工や製造のために必要なすべてのパラメータの値が決定される. 本評価系は, これら3つの設計段階のうち基本設計の支援を対象とし, 概念設計において定義された代数制約集合を評価するために用いられる.

2.2. 提供される機能

基本設計を支援する上で重要なことは、パラメータの値を決定するために必要な情報を設計者に対して提示することだと考えられる。そのために、本評価系では以下の機能が提供される。

1. 代数制約集合に矛盾が存在することを示す。

設計者は代数制約全体の見直しを行い、代数制約の一部を緩和するか、あるいは概念設計そのものを変更する。

2. 互いに矛盾する代数制約を示す。

設計者は矛盾する代数制約の見直しを行う。

3. 新しい代数制約を導出する。

設計者は、導出された代数制約を、概念設計において定義された代数制約集合に追加する。場合によっては、古い代数制約を新しい代数制約で置き換える。

4. すべての代数制約を満足する数値解を例示する。

設計者は、例示された数値解を参考にパラメータの値を決定する。あるいは、さらに代数制約を追加することにより、解の絞り込みを行う。

2.3. システム構成

本評価系は、数式処理部と数値計算部からなる。Fig. 1 にその概念的な構成を示す。

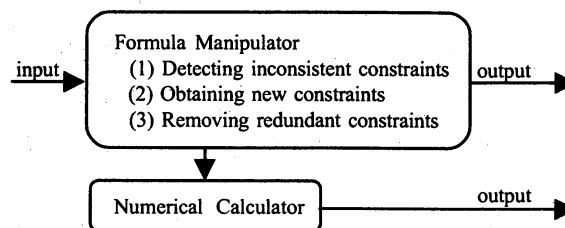


Fig. 1. Architecture of the under constraint solver

入力された代数制約集合は数式処理部に渡され、簡約木 [3] が構成される。これにより、

- (1) 互いに矛盾する代数制約の検出,
- (2) 新しい代数制約の導出,
- (3) 冗長な代数制約の検出,

が行われる。矛盾が検出されるかまたは新しい代数制約が導出された場合、それは直ちに設計者へ示される。矛盾が検出されず新しい代数制約も導出されなかった場合には、代数制約集合は、冗長な代数制約が取り除かれたのち、数値計算部へ送られる。数値計算部では、過小制約問題を最小値問題に帰着させることにより、数値解の計算が行われる [4]。解が存在しない場合には、空集合が返される。

次に、数式処理部および数値計算部で用いられているアルゴリズムについて述べる。

2.3.1. 数式処理部

概念設計において定義された代数制約集合を式(1)とする.

$$\begin{aligned} f_1(x_1, \dots, x_n) = 0, \dots, f_p(x_1, \dots, x_n) = 0, \\ g_1(x_1, \dots, x_n) \neq 0, \dots, g_q(x_1, \dots, x_n) \neq 0, \\ h_1(x_1, \dots, x_n) \geq 0, \dots, h_r(x_1, \dots, x_n) \geq 0. \end{aligned} \quad (1)$$

式(1)は, スラック変数 s_j および t_k を導入することにより, 次式(2) $\cap$ (3)に置き換えられる.

$$\begin{aligned} f_1(x_1, \dots, x_n) = 0, \dots, f_p(x_1, \dots, x_n) = 0, \\ g_1(x_1, \dots, x_n) \cdot s_1 = 1, \dots, g_q(x_1, \dots, x_n) \cdot s_q = 1, \\ h_1(x_1, \dots, x_n) = t_1, \dots, h_r(x_1, \dots, x_n) = t_r. \end{aligned} \quad (2)$$

$$t_1 \geq 0, \dots, t_r \geq 0. \quad (3)$$

数式処理部では, 式(2)から $t = (t_1, \dots, t_r)$ のみで構成される等式が導かれ, 以下の規準にしたがって, 式(3)との矛盾の検出と新しい制約の導出が行われる.

- t_k の正係数多項式が負の定数に等しいとき, 矛盾.
例えば, " $t_1 + 2t_2t_3 = -1$ " が導かれたとすると, これは矛盾である.
- t_k の正係数多項式が0に等しいとき, 新しい制約が導出される.
例えば, " $t_1 + 2t_2t_3 = 0$ " が導かれたとすると, 新しい制約 " $h_1(x) = 0 \wedge \{h_2(x) = 0 \vee h_3(x) = 0\}$ " が得られる. ただし, $x = (x_1, \dots, x_n)$ である.

本矛盾検出法の幾何学的意味は, Fig. 2を用いて以下のように説明される. スラック変数 t および $s = (s_1, \dots, s_q)$ の導入により, 式(1)で表される x 空間における曲面は, 式(2) $\cap$ (3)で表される (x, s, t) 空間における曲面に変換される. 式(2) $\cap$ (3)に矛盾が存在することは, 式(2)で表現される曲面が式(3)で表現される領域を通過しないことを意味する. したがって, 式(2)で表現される (x, s, t) 空間の曲面を t 空間へ射影することにより, 式(2) $\cap$ (3)の矛盾を検査することができる.

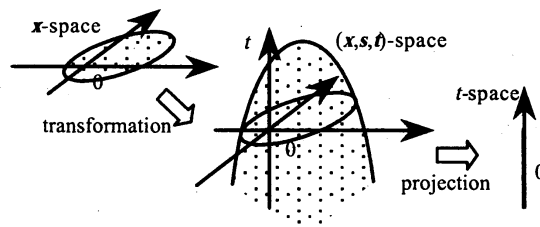


Fig. 2. Geometric meaning of the inconsistency checking

t のみからなる等式の導出は, 簡約木を構成することによって行われる. 簡約木は, 式(2)の各式を節点とする木を作成し, 個々の式を簡約規則と見なすことによって相互に簡約化[1]を行うことにより構成される. その手順を以下に, また, 簡約木の構成例を Fig. 3に示す.

1. $C := \{f_i(\mathbf{x}) = 0, g_j(\mathbf{x}) \cdot s_j = 1, h_k(\mathbf{x}) = t_k \mid 1 \leq i \leq p, 1 \leq j \leq q, 1 \leq k \leq r\}$, $C' := \emptyset$ とする. C の各要素を端点とする木を構成し, これを簡約木の初期状態とする.
2. 簡約木の各節点において, その節点の制約式を C の各要素について 1 回だけ簡約化し, その結果生成された式をその節点の子節点として簡約木に追加する. C の 2 つ以上の要素によって簡約化が可能な場合, そのすべてについて子節点を生成する. ただし, 簡約化の結果生成された式がその節点の子節点としてすでに存在している場合には, 子節点は生成されない. C のどの要素によっても簡約化が不可能だった場合, その節点における制約式を C' に追加する.
3. 生成された制約式が t のみからなる式であれば, 先述の規準による検査を行う. 矛盾が検出されるか, または, 新しい制約が導出された場合, 終了.
4. 新しく追加された節点があれば, ステップ 2 へもどる.
5. $C' = \emptyset$ であれば, 終了. そうでなければ $C := C \cup C'$ としてステップ 2 へもどる.

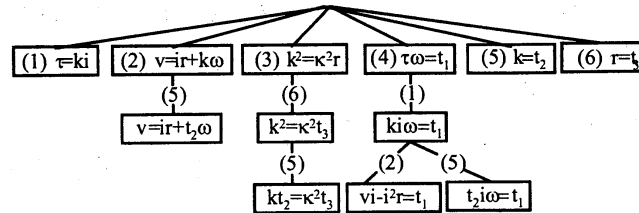


Fig. 3. Reduction tree

簡約木の構成による矛盾検出が有効に行われるように, 本評価系では以下の手法を用いている.

1. 節点生成の抑制

ある節点における制約式が C の複数の要素によって簡約化可能であり, かつ, それらの頭項が互いに素であるとき, それら簡約化可能な要素のうちの 1 つについてのみ子節点が生成される. これは, 頭項が互いに素な要素を用いた簡約化によって最終的に得られる多項式は, その要素の適用順序に依存しないという事実によるものである.

2. 変数間順序の設定

t のみからなる等式を導くため, t の順位は, 辞書的に最も低く設定される. さらに, 変数間の依存関係を示す制約グラフ [2] において, 不等式制約と直接結びついている変数の順位を t に次いで低く設定する. 制約グラフの例を Fig. 4 に示す.

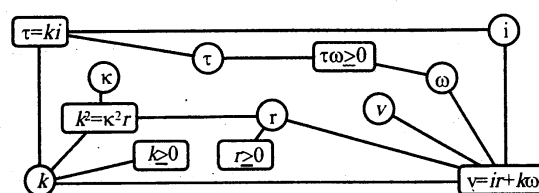


Fig. 4. Constraint graph

3. 等式制約の正規化

制約同士は、制約グラフを介して互いに影響し合う。すなわち、制約相互の簡約化は、制約グラフの枝にそって進むものと考えられる。簡約化を滞りなく行い、最終的に t のみからなる等式を導くために、前項で設定した項順序における等式制約のグレブナ基底 [1] を計算し、等式制約の正規化を行う。

2.3.2. 数値計算部

数値計算部は、式 (2) $\cap$ (3) を入力として受け取り、その数値解を計算する。式 (2) $\cap$ (3) で表される領域を R とし、 $(\mathbf{x}, \mathbf{s}, t)$ 空間において有限個の最小点を持つ関数 $u(\mathbf{x}, \mathbf{s}, t)$ を導入する。

$$R = \{(\mathbf{x}, \mathbf{s}, t) | f_1(\mathbf{x}) = 0, \dots, f_p(\mathbf{x}) = 0, g_1(\mathbf{x}) \cdot s_1 = 1, \dots, g_q(\mathbf{x}) \cdot s_q = 1, \\ h_1(\mathbf{x}) = t_1, \dots, h_r(\mathbf{x}) = t_r, t_1 \geq 0, \dots, t_r \geq 0\}.$$

領域 R の境界は閉じているので、 $R \neq \emptyset$ であれば、関数 $u(\mathbf{x}, \mathbf{s}, t)$ は領域 R において最小値を持つ。したがって、領域 R における関数 $u(\mathbf{x}, \mathbf{s}, t)$ の最小値を計算することにより、式 (2) $\cap$ (3) 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、過小制約問題 (2) $\cap$ (3) は、最小値問題に帰着される。関数 $u(\mathbf{x}, \mathbf{s}, t)$ が領域 R において最小値を持たなければ、それは $R = \emptyset$ を意味する。

関数 $u(\mathbf{x}, \mathbf{s}, t)$ の最小点は、その極値点を計算することによって求められる。極値点の計算は、領域 R の境界と内部に分けて行う必要がある。したがって、極値点の計算に先立ち、領域 R を次式 (4) のように分割する。

$$R = \bigcup R_{ij}, \\ R_{ij} = R \cap \{(\mathbf{x}, \mathbf{s}, t) | t_{i_1} = \dots = t_{i_j} = 0, t_{i_{j+1}} < 0, \dots, t_{i_r} < 0\}. \quad (4)$$

部分領域 R_{ij} における関数 $u(\mathbf{x}, \mathbf{s}, t)$ の極値点は Lagrange 乗数法 [6] により計算されるが、ここで生じてくる問題点とその解決方法について述べ、その後極値点の計算手順を示す。

問題点は大きく2種類に分けられる。1つは極値点を計算すること自体にともなう問題点であり、1つは Lagrange 乗数法を用いることにともなう問題点である。

(1) 極値点計算にともなう問題点

極値点が無限個存在する場合、それら無限個の解の中から実数解を特定する必要がある。本評価系では以下の方法を用いている。

1. 極値点の軌跡を表す等式の集合を T とする。また、 $U := \{u(\mathbf{x}, \mathbf{s}, t)\}$ とする。
2. 等式集合 T の解が有限個であれば、解を計算して終了。
3. $(\mathbf{x}, \mathbf{s}, t)$ 空間において有限個の最小点を持つ関数 $u'(\mathbf{x}, \mathbf{s}, t)$ を導入する。ただし $u'(\mathbf{x}, \mathbf{s}, t)$ は、 $\forall u(\mathbf{x}, \mathbf{s}, t) \in U$ について、 $u(\mathbf{x}, \mathbf{s}, t) = \text{const.}$ なる条件の下で恒等的には一定値としない関数である。
4. 等式集合 T によって与えられる条件の下で、 $u'(\mathbf{x}, \mathbf{s}, t)$ の極値点を計算する。
5. 極値点の軌跡を表現する等式集合を T とする。また、 $U := U \cup \{u'(\mathbf{x}, \mathbf{s}, t)\}$ として、ステップ2へもどる。

(2) Lagrange 乗数法にともなう問題点

等式制約 $w_1(z) = \dots = w_l(z) = 0$ ($z = (z_1, \dots, z_m)$) のもとにおける関数 $v(z)$ の極値点は、次式で与えられる Lagrange 関数 $L(z, \lambda)$ の極値点として求められる。

$$L(z, \lambda) = v(z) + \sum_{i=1}^l \lambda_i w_i(z), \quad \lambda = (\lambda_1, \dots, \lambda_l).$$

これが Lagrange 乗数法であり、条件付き最小値問題の解法として広く用いられている方法であるが、その適用可能な問題および計算される極値点に関して以下の条件がある。

- 適用可能な問題に関する条件
制約条件の数 l が独立変数の数 m よりも少ない問題にしか適用できない。
- 計算される極値点に関する条件
特異点、すなわちヤコビ行列 $J(z)$ の階数が l よりも小さくなる点は計算されない。

$$J(z) = \begin{pmatrix} \partial w_1 / \partial z_1 & \cdots & \partial w_1 / \partial z_m \\ \vdots & & \vdots \\ \partial w_l / \partial z_1 & \cdots & \partial w_l / \partial z_m \end{pmatrix}.$$

ここで、

- $l \geq m$ の場合の極値点の計算方法、
- 特異点の計算方法、

について述べるが、これに先立ち等式制約 (5) の中に含まれる冗長な等式の検出方法について触れておく。

$$w_1(z) = \dots = w_l(z) = 0. \quad (5)$$

本稿では、冗長な等式を以下のように定義する。

冗長な等式: その等式を除いても、生成されるイデアルがもとのものと同一であるような等式。

冗長な等式は以下の手順により検出される。まず、式 (5) のグレブナ基底 G_0 を計算する。以降、各 $j = 1, \dots, l$ について式 (5) から $w_j(z) = 0$ を除いた連立方程式のグレブナ基底 G_j を計算する。 $G_0 = G_j$ であれば $w_j(z) = 0$ は冗長である。

以降、式 (5) に冗長な等式は含まれていないとの前提の下で、上述の (a), (b) の計算方法について議論する。

(a) $l \geq m$ の場合の極値点の計算法

$J(z)$ の階数を $\text{rank}(J(z))$ と表す。 $l > m$ であれば任意の点 z において $\text{rank}(J(z)) < l$ であるので、特異点の計算方法にしたがう。 $l = m$ の場合、 $\text{rank}(J(z)) = l$ なる点、すなわち正則点と特異点の混在が考えられる。以下の手順でこれらの点が求められる。

1. 正則点の計算

$J(z)$ の行列式を $|J(z)|$ と表すと、正則点は $|J(z)| \neq 0$ を満足する。したがって、新しく変数 ζ を導入し、式 (5) と $|J(z)| \cdot \zeta = 1$ を連立させることにより正則点が計算される。なお、正則点の個数は有限個である [6]。

2. 特異点の計算

特異点は $|J(z)| = 0$ を満足する。したがって、式 (5) と $|J(z)| = 0$ を連立させることにより特異点の軌跡が得られる。これを式 (5) に代えて新たな等式制約とし、計算を進める。

(b) 特異点の計算方法

特異点では $J(z)$ の行ベクトルは 1 次従属となり、次式 (6) が成立する。

$$\exists c_1, \dots, c_l, \neg(c_1 = \dots = c_l = 0) \\ c_1 \begin{Bmatrix} \partial w_1 / \partial z_1 \\ \vdots \\ \partial w_l / \partial z_m \end{Bmatrix} + \dots + c_l \begin{Bmatrix} \partial w_l / \partial z_1 \\ \vdots \\ \partial w_l / \partial z_m \end{Bmatrix} = \begin{Bmatrix} 0 \\ \vdots \\ 0 \end{Bmatrix}. \quad (6)$$

式 (5) $\cap$ (6) に $c_k = 1$ ($1 \leq k \leq l$) を連立させれば、特異点のうち $c_k \neq 0$ に対応する点の軌跡が求められる。軌跡が有限個の点からなれば、それを計算する。軌跡が無限個の点からなれば、これを式 (5) に代えて新たな等式制約として計算を進める。

以上に述べた手順にしたがい、等式制約を次々と更新することによって、極値点が求められる。新しく得られた等式制約がもとのものと同一となるときには、無限ループに陥ることを防ぐために計算を打ち切る必要がある。その場合には、極値点を求めることができない。そのような例を以下に示す。

無限ループに陥る等式制約の例: $z_3 - z_1 z_2 = 0, z_1^2 - z_2 = 0, z_2^2 - z_1 z_3 = 0$.

これは変数の数と等式制約の数が等しいにも関わらず、1 次元となる代数多様体である [7]。

(3) 部分領域 R_{ij} における関数 $u(x, s, t)$ の極値点の計算手順

領域 R の部分領域 R_{ij} における関数 $u(x, s, t)$ の極値点は Fig. 5 に示す手順で計算される。部分領域 R_{ij} は、まず、その解が有限個か無限個かが調べられる。無限個の場合、変数の数と等式の数と比較され、その結果により前項で述べた手順にしたがって極値点の計算が行われる。

なお本評価系では、極値点の計算に際して、数値誤差による計算の誤りを排除するために、グレブナ算法を基本とした外接球法 [5] を採用している。

3. 適用例

本章では、2 つの簡単な問題を用いて、本評価系の適用例を示す。

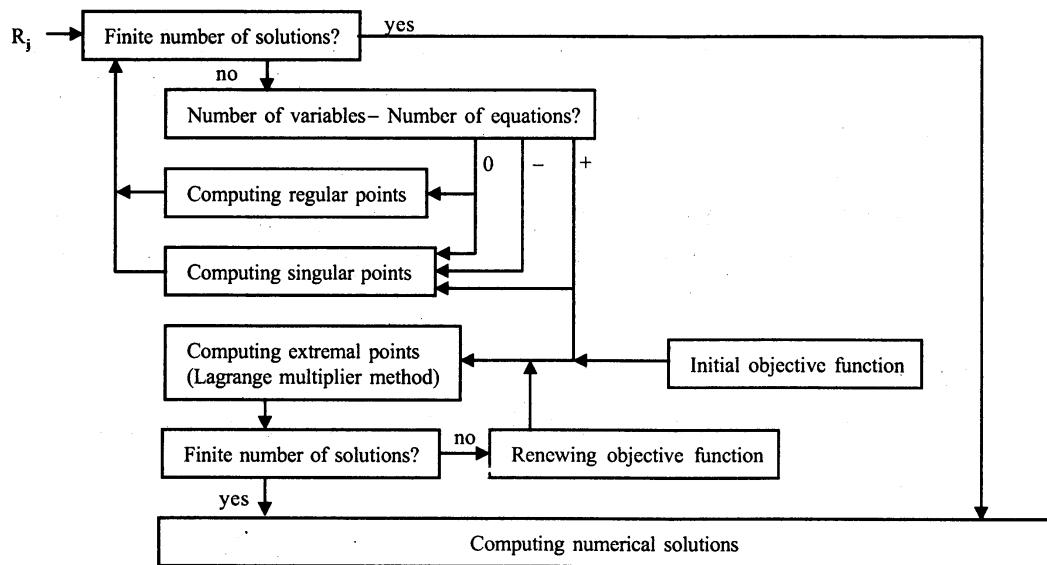


Fig. 5. Computation of extremal points

3.1. 動力伝達機構

Fig. 6は、電源、モーター、歯車からなる動力伝達機構であり、各機械要素に関する制約は次のように与えられる。

電源: $0 \leq v_b \leq v_0$, $0 \leq i_b \leq i_0$.

モーター: $\tau = ki$, $v = ir + k\omega$, $\tau\omega \geq 0$, $k^2 = \kappa^2 r$, $k \geq 0$, $r \geq 0$.

歯車: $\tau_i = \rho\tau_o$, $\rho\omega_i = \omega_o$.

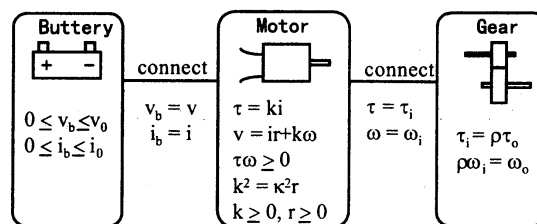


Fig. 6. Transmission

また、これらの機械要素を接続するための制約は以下のように与えられる。

電源とモーターの接続: $v_b = v$, $i_b = i$.

モーターと歯車の接続: $\tau = \tau_i$, $\omega = \omega_i$.

このような動力伝達機構に対して、

$$(1) \tau_o \omega_o \geq v_0 i_0,$$

(2) $v_0 = 15[\text{V}]$, $i_0 = 0.1[\text{A}]$, $1/4 \leq \rho \leq 4$, $\kappa = 0.001[\text{Nm}/\text{V}^{1/2}\text{A}^{1/2}]$, $\tau_o \geq 0.005[\text{Nm}]$,

なる要求仕様が与えられた場合についてそれぞれ考える.

(1) $\tau_o \omega_o \geq v_0 i_0$ のとき

定義された不等式制約は数式処理部へ送られ, 第 2.3.1. 項で述べた手順にしたがって等式制約に置き換えられ, 次式 (7) の制約集合をもとに簡約木が構成される.

$$\begin{aligned} \{v_b = t_1, v_0 - v_b = t_2, i_b = t_3, i_0 - i_b = t_4, \tau = ki, v = ir + k\omega, \tau\omega = t_5, k^2 = \kappa^2 r, \\ k = t_6, r = t_7, \tau_i = \rho\tau_o, \rho\omega_i = \omega_o, v_b = v, i_b = i, \tau = \tau_i, \omega = \omega_i, \tau_o\omega_o - v_0i_0 = t_8\}. \end{aligned} \quad (7)$$

簡約木を構成することにより式 (8) が得られ, 新しい制約 (9) が導かれる.

$$t_1t_4 + t_2t_3 + t_2t_4 + t_3t_7 + t_8 = 0 \quad (8)$$

$$\begin{aligned} (v_b = 0 \vee i_b = i_0) \wedge (v_b = v_0 \vee i_b = 0) \wedge (v_b = v_0 \vee i_b = i_0) \\ \wedge (r = 0 \vee i_b = i_0) \wedge (\tau_o\omega_o = v_0i_0). \end{aligned} \quad (9)$$

(2) $v_0 = 15[\text{V}]$, $i_0 = 0.1[\text{A}]$, $1/4 \leq \rho \leq 4$, $\kappa = 0.001[\text{Nm}/\text{V}^{1/2}\text{A}^{1/2}]$, $\tau_o \geq 0.005[\text{Nm}]$ のとき

(1) と同様の手順により, 数式処理部において式 (10) の制約集合から簡約木が構成される.

$$\begin{aligned} \{v_b = t_1, v_0 - v_b = t_2, i_b = t_3, i_0 - i_b = t_4, \tau = ki, v = ir + k\omega, \tau\omega = t_5, k^2 = \kappa^2 r, \\ k = t_6, r = t_7, \tau_i = \rho\tau_o, \rho\omega_i = \omega_o, v_b = v, i_b = i, \tau = \tau_i, \omega = \omega_i, v_0 = 15, i_0 = 0.1, \\ \rho - 1/4 = t_8, 4 - \rho = t_9, \kappa = 0.001, \tau_o - 0.005 = t_{10}\}. \end{aligned} \quad (10)$$

式 (11) が導かれ, 式 (12) の不等式制約の間に矛盾が生じることがわかる.

$$\begin{aligned} 80t_1t_4 + 8t_2 + 80t_5 + 125t_8^2t_{10}^2 + 250t_8t_{10}^2 + 125t_{10}^2 \\ + 250t_8^2t_{10} + 500t_8t_{10} + 250t_{10} + 125t_8^2 + 250t_8 + 5 = 0. \end{aligned} \quad (11)$$

$$0 \leq v_b \leq v_0, i_b \leq i_0, \tau\omega \geq 0, \tau_o \geq 0.005, 1/4 \leq \rho. \quad (12)$$

3.2. 歯車箱

Fig. 7 に示す 3 軸歯車箱の, 歯車軸の配置問題を考える. この問題は, 歯車径と箱の大

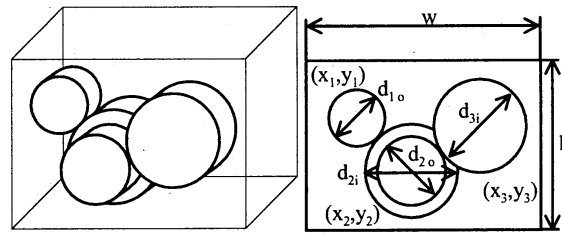


Fig. 7. Three-axes gearbox

きさが与えられたときに歯車軸の位置を求めるものである。制約条件は次式(13)で与えられる。

$$\begin{aligned}
 (x_2 - x_1)^2 + (y_2 - y_1)^2 &= ((d_{1o} + d_{2i})/2)^2, \\
 (x_3 - x_2)^2 + (y_3 - y_2)^2 &= ((d_{2o} + d_{3i})/2)^2, \\
 (x_1 - x_3)^2 + (y_1 - y_3)^2 &\geq ((d_{1o} + d_{3i})/2)^2, \\
 d_{1o}/2 \leq x_1 \leq w - d_{1o}/2, \quad d_{1o}/2 \leq y_1 \leq h - d_{1o}/2, \\
 d_{2i}/2 \leq x_2 \leq w - d_{2i}/2, \quad d_{2o}/2 \leq x_2 \leq w - d_{2o}/2, \\
 d_{2i}/2 \leq y_2 \leq h - d_{2i}/2, \quad d_{2o}/2 \leq y_2 \leq h - d_{2o}/2, \\
 d_{3i}/2 \leq x_3 \leq w - d_{3i}/2, \quad d_{3i}/2 \leq y_3 \leq h - d_{3i}/2.
 \end{aligned} \tag{13}$$

また、歯車径は式(14)として与えられるものとし、

$$(1) \quad w = 325, h = 300,$$

$$(2) \quad w = 160, h = 160,$$

の場合について、それぞれ考える。

$$d_{1o} = 42, d_{2i} = 86, d_{2o} = 70, d_{3i} = 110. \tag{14}$$

(1) $w = 325, h = 300$ のとき

制約集合は数式処理部へ送られるが、矛盾は検出されない。冗長な制約が取り除かれた後、数値計算部において数値解の計算が行われる。得られた解のうちの3つを示す。

$$\begin{aligned}
 (x_1, y_1, x_2, y_2, x_3, y_3) &= (121, 279, 181, 257, 270, 245), \\
 &= (204, 279, 144, 257, 55, 245), \\
 &= (121, 21, 181, 43, 270, 55).
 \end{aligned}$$

(2) $w = 160, h = 160$ のとき

制約集合は数式処理部へ送られるが、矛盾は検出されない。冗長な制約が取り除かれた後、数値計算部において数値解の計算が行われる。しかし、数値解は求められず、この制約を満足する解は存在しないことがわかる。

4. おわりに

数式処理部と数値計算部からなる過小制約評価システムについて述べ、その基本設計への応用例を示した。例題にも見られるように、数式処理部における矛盾検出機能は完全なものではないが、互いに矛盾する制約を特定することが可能である。一方、数値計算部では、互いに矛盾する制約を特定することはできないものの、その数値計算の過程において矛盾の検査が完全に行われる。これらを相補的に用いることにより、基本設計を有効に支援することが可能であると考えられる。今後、より多くの実際的な設計事例に適用することにより問題点の洗い出しを行い、計算効率の向上をはじめとするシステムの改良を進めていく必要がある。

参 考 文 献

- [1] Buchberger, B.: Gröbner Basis: An Algorithmic Method in Polynomial Ideal Theory, Technical Report, RISC-LINZ (1983).
- [2] 永井保夫, 生駒憲治: 制約グラフのスパース構造に基づいた整合性解析と制約処理の効率化についての検討, ICOT 研究速報, TM-0928(1990).
- [3] 沢田浩之: 過小代数制約問題における矛盾の簡易検出法, 情報処理学会第 55 回全国大会, 1-295 (1997).
- [4] 沢田浩之: 多変数連立代数不等式の解法, 情報処理学会第 49 回全国大会, 1-101 (1994).
- [5] 沢田浩之: 新たな条件式の導入による多変数連立代数方程式の解法, 情報処理学会論文誌, Vol.36, No.12, pp.2761-2770 (1995).
- [6] 高木貞治: 解析概論, 岩波書店 (1961).
- [7] 上野健爾: 代数幾何入門, 岩波書店 (1995).