遺伝的プログラミングによる形状可変トラス構造の 自律分散形姿勢制御則の獲得*

山崎 光悦^{*1}, Sourav KUNDU^{*1}, 浜野 道知^{*2}

Acquisition of Autonomous Distributed Rule for Posture Control of Variable Geometry Truss by Genetic Programming

Koetsu YAMAZAKI, Sourav KUNDU and Michitomo HAMANO

An acquisition technique of autonomous distributed rule for posture control of Variable Geometry Truss (VGT) is suggested by applying the Genetic Programming (GP) method. Each control rule produced by GP consists of hierarchical tree structures. The main tree structure contains condition trees and rule selection trees as function nodes, and control trees as terminal nodes. The acquisition system trains individual rule by the posture control simulation for the randomly given initial postures, and evaluates the rule by the simulation of given test postures. The validity of the acquisition system is confirmed numerically by applying to the stiffness maximization of ten-module VGT.

Key Words : Artificial Intelligence, Adaptive Control, Space Engineering, Posture Control, Genetic Programming, Variable Geometry Truss, Rule Acquisition, Autonomous Distributed Rule

1. 緒 言

将来、宇宙展開構造や宇宙作業用マニピュレータアームなどの宇宙柔軟構造物への応用が期待される形状可変トラス構造は構造の一部にセンサやアクチュエータを備えており、目的に応じて自らの形状を変えることができる⁽¹⁾⁽²⁾。特に可変長部材を含むトラスモジュールを直列に数段組上げたトラス形アームの軌跡制御や最終目的位置、負荷の角度に対する形状制御、姿勢決定あるいは振動抑制に関する研究が盛んに行われている⁽³⁾⁽⁴⁾。しかしこのような形状可変トラス構造の軌跡制御や姿勢決定を集中制御で行うには、その自由度の大きさなどから制御量の決定や最適化に時間を要し、制御処理速度の限界から実時間での制御が実現困難とされている。

一方、全体を構成するサブシステムが個々に自律性を保ちつつ変化し、全体として目的を遂行する自律分散形システムのもつ即応性と高効率性の特徴を利用して、高自由度をもつ形状可変トラス構造の姿勢適応への応用が研究されているが、自律分散形の適応則を直

感的に与えざるを得ないのが現状である⁽⁵⁾。

そこで本研究では形状可変トラス構造を各トラスモジュールがサブシステムとして機能する自律分散システムとして考え、外部からの負荷に対して最大剛性姿勢決定を目的とし、各モジュールごとの局所的な情報、規範を基に各モジュールを制御し、全体の目的を達成する自律分散形制御則を知識獲得機能をもつ遺伝的プログラミングを用いて進化的に獲得するシステムを開発し、獲得した制御則の有効性を有限要素法によるシミュレーションによって検討する。

2. 形状可変トラス構造

2.1 形状可変トラスモデル 本研究で対象とする形状可変トラス構造について述べる。図1(a)に示すような一辺の長さ L の3部材とアクチュエータを備えた可変長斜材の4部材から成る構造を単位モジュールとし、その単位モジュールを図1(b)のように可変長斜材が交互になるよう直列に n 段積み上げ、その最上段に外部からの荷重負荷を受けるための三角形の2部材からなる構造を設けた平面トラス構造である。各モジュールの可変長斜材の長さを制御することによりトラス全体の姿勢を制御する。荷重 P は先端に設けられた荷重作用点 A を介して水平方向と角度

* 原稿受付 1997年5月1日。

*¹ 正員、金沢大学工学部 (〒920-8667 金沢市小立野 2-40-20)。

*² 学生員、金沢大学大学院。

ϕ をなす方向に作用し、作用点が移動してもその作用方向と荷重値は変化しないものとする。また各モジュールの構成部材にはすべて応力センサ(#0~#3)が取り付けられており、これらから得られる情報をもとに自律分散的な制御を行うものとする。姿勢適応のシミュレーションでは、制御変数として水平部材と可変長斜材とのなす角度 θ_i を用い、 $10^\circ \leq \theta_i \leq 80^\circ$ の制限を設ける。また本研究で扱うトラス構造は、節点において各部材回転中心を共有して接続されると仮定した概念モデルとし、自重の影響は考慮しない。さらに、簡単のため可変長斜材の剛性は部材長さに反比例するものとし、ヤング率一定として扱うことにする。

2.2 姿勢適応シミュレーション 上記のように仮定した形状可変トラス構造の姿勢適応シミュレーションについて説明する。任意の姿勢にあるトラス構造

Fig.1 Model of Variable Geometry Truss (VGT)

Fig.2 Flowchart of adaptive simulation for VGT

を有限要素解析し、その解析値を前述の各モジュールの応力センサ(#0~#3)から得られる情報とし、後述の方法により与えられる制御則により各モジュールに対する制御量を計算し、トラスの姿勢を更新する。このサイクルを1ステップとし、収束条件を満たすまで繰返すことによりトラス構造の姿勢を徐々に変化させてゆく。姿勢適応シミュレーションの流れ図を図2に示す。

2.3 目的制御則 本研究で獲得対象とした制御則について説明する。荷重荷方向が規定されている形状可変トラス構造を対象に、初期姿勢に依存せず各モジュールが自分自身または上下のモジュールの局所情報のみから自分自身の制御量を決定しながら姿勢を変更し、最大剛性姿勢に到達させることのできる自律分散形制御則を獲得する。ここで最大剛性姿勢とは、トラス構造の全ひずみエネルギー U が最小となる姿勢のことであり次式で表せる。

$$\text{目的関数: } U = \sum_{i=0}^{n-1} u_i \rightarrow \min$$

ここに u_i はモジュール i のひずみエネルギーとする。

3. 制御則獲得法

3.1 遺伝的プログラミング 遺伝的プログラミング(GP)は遺伝的アルゴリズム(GA)を拡張させたもので、Koza⁽⁶⁾によって提唱された手法である。GAとの決定的な相違は個体(遺伝子)のもつデータ構造にある。通常GAの個体は一次元のバイナリコードであるのに対し、GPの個体は木構造(またはグラフ構造)で表現される。個体に木構造を用いることによりGAでは表現困難な複雑な関係式や概念を容易に表現でき、各種の操作ができる。

GPで用いる木構造について説明する。図3は木構造の一例で $(x-y)+z$ を表現している。ここで子をもつことのできる記号 $+$, $-$ といったノードを非終端ノード(FUNCTION node)といい、また両ノードとも二つの子をもつので二つの引数をもつという。また

Fig.3 Tree structure

記号 x, y, z といった子(引数)をもたないノードを終端ノード(TERMINAL node)といい、木構造の末端に配置される。図3(b)に示す表現は LISP で用いられる S 式(S-expression)とよばれ、左側の木構造と同一視される。

GP の基本的なアルゴリズムは GA とほぼ同様で、交差、再生産、突然変異などの遺伝的操作を行い、世代を重ねることにより解空間の探索を行う。例として交差の方法について説明する。図4の左側に示すような二つの木構造が親 A, B として選ばれたとする。まず、それぞれの木構造のノード数を数え、ランダムにそれぞれの交差ノード(Crossover point)を決定する。次にそれぞれの交差ノード以下の部分木(Subtree)を切り取り、互いの交差ノードの位置に置き換え(図4の右側)、子孫を作る。GP では GA とは異なり、交差により個体(木構造)の形状やサイズ、構成要素が大きく変化する。

3・2 木構造による制御則表現法 GP を用いて制御則を獲得するには制御則を GP の個体として扱えるよう木構造で表現する必要がある。そこで本論文で提案する木構造の制御則について説明する。

まず制御則は次の3種類の木構造から成るものとした。

- (i) Main-tree: 制御則の流れを記述(制御則本体)。
- (ii) Condition-tree/Select-rule-tree: 制御対象になっているモジュールの状態を上下のモジュールの関係から表現する。
- (iii) Control-tree: モジュール構成部材の応力感度からなる局所的な制御則(本研究では3種類を設定)。

図5に提案する制御則の全体の形態を示す。この制

御則は主プログラムである Main-tree によって制御量を決定する。そして Main-tree の各ノードによってサブプログラムである Condition-tree, Select-rule-tree, Control-tree が呼び出されて計算が行われ、返り値を Main-tree に返すという階層的な関係となっている。またこの制御則の機能は、制御対象となっているモジュール(図5の左下の黒い線で描かれているモジュール)の制御量を決定するとき、そのモジュールおよび上下のモジュールの各部材の応力センサから得られる局所的な情報を Condition-tree, Select-rule-tree, Control-tree に取り込み制御量を決定する。つまり局所的な応力情報のみから制御量を決定できることから、この制御則は自律分散形制御則であるといえる。

次に各木構造の詳細について説明する。

(1) Main-tree 各モジュールの制御量は Main-tree を評価することにより得られる。設定したノードとその機能を表1に示す。非終端ノードには3種類の関数、終端ノードには1種類のノードを設定する。このように設定した Main-tree の評価の方法を具体的に説明する。例として図6に示すような Main-tree を考える。まずルートの Prog 2 から評価を始める。Prog 2(引数を2つもつ関数)は引数0, 1を逐次実行するので、まず引数0のConditionをとる。ConditionではCondition-treeを評価してその返り値が1, -1なら引数0のActionを、返り値が0なら引数1のSelect-ruleをとる。Select-ruleではSelect-rule-treeを評価し、その返り値によって引数0のActionで評価するControl-treeを選ぶ。そしてそのActionではSelect-ruleにより選ばれているControl-treeを評価して制御量を得る。このようにしてMain-tree全体を評価し、1回の適応操作での制

Fig. 4 Crossover on genetic programming

Fig. 5 Form of Autonomous Distributed Control program

Table 1 Setting of function and terminal nodes for Main_tree

Main_tree	In every step, calculate control value of each module at next step by evaluating this tree once for each module.
FUNCTION_node	
Prog2(arg0,arg1)	Evaluate arg0 and arg1 in sequential order.
Condition(arg0,arg1)	If return value of evaluated Condition_tree is 1 or -1, carry out arg0 (but when return value is just -1, invert the control value), if return value is 0, we carry out arg1.
Select_rule(arg0)	By return value of evaluated Select_rule_tree , select Control_tree invokes the following Actions.
TERMINAL_node	
Action	Get control value by evaluating Control_tree . But when Control_tree is not selected by Select_rule_tree , use default Control_tree .

Table 2 Setting of function and terminal nodes for Condition_tree/Select_rule_tree

Condition_tree Select_rule_tree	When these trees are called by Condition or Select_rule node of Main_tree , they return a value of 1,0 or -1 to Main_tree by evaluating corresponding trees. (it is assumed that Control_tree has always three subtrees).
FUNCTION_node	
+(arg0,arg1)	arg0 plus arg1. But when the value is 2 or -2, substitute to 1, -1 respectively.
-(arg0,arg1)	arg0 minus arg1. But when the value is 2 or -2, substitute to 1, -1 respectively.
*(arg0,arg1)	Multiply arg0 by arg1.
Not(arg0)	Invert sign of arg0.
TERMINAL_node	
m_stress i	Stress value, 1(+), 0(0), -1(-) that are calculated by equation given below are substituted. $m_stress\ i = \sigma(s, m, i) - \{\sigma(s, m + l, i) + \sigma(s, m - l, i)\} / 2 \quad i = 0, 1, 2, 3$

Table 3 Setting of function and terminal nodes for Control_tree

Control_tree	When this tree is called by Action node of Main_tree , this tree return control value of Right_Move(1) , Stay(0) , Left_Move(-1) back to Main_tree by evaluating Control_tree that is selected by Select_rule node of Main_tree .
FUNCTION_node	
Same as Condition_tree / Select_rule_tree	
TERMINAL_node	
s_stress i	Stress sensitivity, 1(+), 0(0), -1(-) that are calculated by equation given below are substituted. $s_stress\ i = \{\sigma(s, m, i) - \sigma(s - l, m, i)\} / \Delta \alpha(m) \quad i = 0, 1, 2, 3$

Fig. 6 Example of Main_tree

御量を計算する。

(2) **Condition_tree/Select_rule_tree** **Condition_tree/Select_rule_tree**に設定するノードとその機能を表2に示す。非終端ノードには4種類の数学演算子、終端ノードに **m_stress** という制御対象となっているモジュールの部材応力とそれに対応する上下のモジュールの平均応力との差を設定する。制御対象となっているモジュールの各構成部材の **m_stress** を求

め、それらを非終端ノードで演算し、**Main_tree**に-1, 0, 1の三つの状態を返す。なお最上段での **m_stress** の計算には最下段のモジュールを一つ上のモジュールとして扱い、最下段では最上段のモジュールを一つ下のモジュールとして扱って計算する。

(3) **Control_tree** **Control_tree**に設定するノードとその機能を表3に示す。非終端ノードは上述の **Condition_tree/Select_rule_tree**と同じである。終端ノードには **s_stress** という前ステップと現ステップの部材応力から計算された各構成部材の応力感度を設定し、これらを非終端ノードで演算することにより局所的な制御則を形成している。

結局、この制御則は **Main_tree**, **Condition_tree**, **Select_rule_tree** が一つずつ **Control_tree** が三つで計六つの木構造から構成されることになり、GPではこの六つの木構造の集まりを1つの個体とする。またこの制御則はある適応操作のステップでの状態や制御

Fig. 7 Flowchart of evaluation of individual rule

対象となっているモジュールの位置関係により制御量に変化を与えることができ、GPにより多様性に富んだ制御則の生成が可能となる。

3.3 個体の評価方法 上記のように考えた制御則である個体の評価の方法を述べる。全体の流れ図を図7に示す。まず各個体ごとにランダムな初期姿勢を与え、個体を用いて姿勢適応シミュレーションを行う。この過程を訓練と称する。この訓練における姿勢適応シミュレーションの結果より適合度を計算し、各個体の適合度より世代の最良個体を決定する。次に最良個体を用いて、あらかじめ設定しておいたテスト姿勢で姿勢適応シミュレーションを行い、その結果より計算される適合度を最良個体の適合度とする。この過程をテストとする。その後、他世代の最良個体との比較を行い世代を更新する。

このように訓練、テストという2段階の評価を行うことにより、初期姿勢に依存せず、目的の姿勢である最大剛性姿勢に達する制御則を得ることができる。

4. GPによる制御則獲得例

4.1 負荷方向規定モデル ($\phi=45^\circ$) 対象とする形状可変トラス構造は10段モジュールとし、部材長さ $L=1000$ mm, 断面積 $A=100$ mm², ヤング率 $E=205.8$ GPaとした。また負荷条件は負荷荷重 $P=294$ N, 負荷方向 $\phi=45^\circ$ とした。GP, 姿勢適応シミュレーションでの主なパラメータを表4に示す。姿勢適応シミュレーションの収束条件としては、すべてのモジュールの制御量が零となったとき(1・a), またはステップ数が最大ステップとなったとき(2・a)を考えた。GPの収束条件は、最良個体が10世代以上更新しない

Table 4 Parameters of GP/simulation

GP	
Population size	500
Max generation	50
Selection method	TOURNAMENT
Crossover probability at function node	70%
Crossover probability at any node	15%
Reproduction probability	10%
Mutation probability	5%
Number of training	3
Number of TEST postures	3
Truss simulation	
Max step	100
Unit control value (degree)	2.0

Fig. 8 TEST posture

Main_tree (Select_rule Action)	Control_tree #1 (s_stress3+s_stress1)
Condition_tree (m_stress1+m_stress3)	#0(default) ((s_stress1+s_stress1) +((s_stress1*s_stress0) +s_stress3))
Select_rule_tree (Not(m_stress2))	#1 (s_stress2+ s_stress2)

Fig. 9 Control program

とき(1 b), 世代が最大世代数となったとき(2 b)とした。またテスト姿勢としては図8に示す3種類の姿勢を用いた。

適合度の計算方法としては目的の姿勢が最大剛性姿勢であることから、評価の際に行う姿勢適応シミュレーションでの収束姿勢の全エネルギー値をそのまま個体(制御則)の適合度として用いた。

GPは13世代において収束条件1 bで収束した。最良個体は3世代で獲得され、その個体の適合度は0.0184であった。得られた制御則のS式表現を図9に示す。制御則のMain-treeは二つのノードから成る単純な木構造となっている。またMain-treeはConditionを含んでいないことから実際のトラス制御にはCondition-treeは使用されることがわかる。

得られた制御則を用いてテスト姿勢の一つである姿勢1とテスト姿勢には含まれない半円弧状の姿勢3の二つの初期姿勢で姿勢適応シミュレーションを行った。それぞれの初期姿勢と収束姿勢および比較のため

Fig. 10 Initial/converged posture and optimum posture

Table 5 Result of simulation

	Posture1		Postur3		Optimum
Converged step	initial	99	initial	99	—
Total energy(J)	2.7673	0.0184	0.8837	0.0237	0.01203
Converged condition	2a		2a		—

Fig. 11 Total strain energy variations

ランダムサーチによって求めた最適姿勢を図10に示す。また姿勢適応シミュレーションの収束値と最適姿勢のエネルギー値を表5に示す。どちらの収束姿勢も荷重方向に向いた姿勢となっており、最適姿勢に十分に近い姿勢を得ている。エネルギー値も最適値には及ばないものの、初期値と比較すると十分に低い値となっていることがわかる。また図11は適応過程における全エネルギー値の変化を示すが、初期のステップにおいてエネルギー値が着実に減少し、20ステップ前後で収束姿勢に到達している。なおその後のエネルギー値の上下は、数段のモジュールが一定の間隔で振動しているために生じている。このように、獲得した制御則はテスト姿勢および任意の初期姿勢に対して剛性の高い姿勢を得ることができていると判断できる。

4.2 荷重方向規定モデル2 ($\phi=90^\circ$) 次に荷重方向を $\phi=90^\circ$ に変えたモデルの制御則の獲得を行った。荷重方向以外のパラメータは先のモデルと同じである。

GPは22世代において収束条件1bで収束した。最良個体は12世代で獲得され、その個体の適合度は0.0125であった。得られた制御則のS式表現を図12

```

Main_tree
0 (Condition
1  Action
2  (Condition
3    (Condition
4      Action
5      Action)
6    (Select_rule
7      Action)))

Select_rule_tree
((m_stress2* m_stress1)
- ((m_stress2- m_stress2)
- m_stress3))

Control_tree
#-1
(Not(s_stress1))

#0(default)
(s_stress2+s_stress2)

#1
(s_stress1*s_stress3)

Condition_tree
(Not(m_stress2))

```

Fig. 12 Control program

Fig. 13 Initial/converged posture

に示す。図9の制御則と比べMain-treeがやや複雑な構造となっている。しかし実際には図12に示すMain-treeの3~5行めは評価されることはなく、単純な構造といえる。

得られた制御則を用いて先と同様の初期姿勢で姿勢適応シミュレーションを行った。図13にそれぞれの初期姿勢および収束姿勢を示す。またそのときのエネルギー値を表6に示す。どちらの収束姿勢もまっすぐ立った姿勢となっており、荷重方向に向いた姿勢となっている。エネルギー値も低い値となっており、十分に剛性の高い姿勢といえる。図14は適応過程における全ひずみエネルギー値の変化を示したものであるが、 $\phi=45^\circ$ のときと同様で初期のステップでエネルギー

Table 6 Result of simulation

	Posture1		Posture3	
Converged step	initial	99	initial	99
Total energy(J)	1.0585	0.0121	0.2918	0.0135
Converged condition	2a		2a

Fig. 14 Total strain energy variations

ギー値は急激に減少し、20 ステップまでに収束姿勢に達している。その後、数個のモジュールが一定の間隔で振動しているためにエネルギー値が上下し続け、収束に至っている。

5. 結 言

本報では、形状可変トラス構造を対象にした自律分散形制御則を遺伝的プログラミングにより獲得するための方法を提案した。また負荷方向規定モデルで $\phi = 45^\circ, 90^\circ$ の二つの荷重条件で制御則の獲得を行い、その制御則を用いて二つの初期姿勢に対して姿勢適応シミュレーションを行った結果を示し、その有効性を確認した。またその他数種類の初期姿勢に対しても姿勢適応シミュレーションを行っており、どちらの制御則も初期姿勢に依存せず剛性の高い姿勢が得られることを確認している。

文 献

- (1) Wada, B. K., Fanson, J. I. and Crawley, E. F., *Mech. Eng.*, **112**-11(1990), 41-46.
- (2) 名取道弘・中谷一郎, 機誌, **94**-871(1991), 483-488.
- (3) 室津義定・ほか3名, 機論, **57**-543, C(1991), 3518-3525.
- (4) Tanaka, M. and Hanahara, K., *J. Intell. Mater. Syst. Struct.*, **3**-4(1992), 617-630.
- (5) 山崎光悦・平野薫, 機論, **62**-601, A(1996), 2173-2179.
- (6) Koza, J., *Genetic Programming*, (1992), MIT Press.