原著論文

n -BDD のための節点の動的追加手法 - ロボットの行動則獲得への適用 -[†]

加納 政芳^{*1}・伊藤 英則^{*2}

本稿では、多出力二分決定グラフ (n -BDD) を利用したロボットの行動則の獲得手法を提案する。BDD は、論理関数の表現方法の 1 つであり、LSI の設計や、CAD システムのモデル検査などに利用されている。森脇らは、出力を複数持つ BDD を遺伝的プログラミングによって進化させる手法 n -BDD を提案し、これを食物連鎖の進化シミュレーションに用いることで、シミュレーション環境内で植物・草食動物・肉食動物の疑似生態系を実現している。しかし、この手法は、進化の過程では、BDD における変数順序の変更は認められておらず、疑似生態系の構築のために有用な変数順序を評価していた。そこで本稿では、この問題を回避するために変数順序を動的に決定できる新しい遺伝的操作を提案する。また、ロボットの行動則獲得のために交叉演算を導入する。実験では、ロボットの行動則獲得シミュレーション実験を行い、本手法の有効性を確認した。

キーワード：二分決定グラフ、遺伝的プログラミング、ロボット制御、行動則獲得、節点の動的追加

1. はじめに

近年、ロボット分野において、掃除や留守番を行うロボットや、さらには、人とコミュニケーションを行うロボットなど、家庭で使用するロボット(家庭用ロボット)への注目が高まってきている[1]～[4]。この中でも、人の作業を代行するためには身体性が類似しているほうが様々な作業を行えることから、人型ロボット(ヒューマノイドロボット)への関心は特に高い。しかしながら、複雑な機構を持つヒューマノイドロボットを制御することは難しく、複雑な力学計算が必要とされる。この問題を回避するために、本稿では、ロボットの行動則獲得のために多出力二分決定グラフを用いることを考える。

二分決定グラフ(Binary Decision Diagram; BDD) は、1 つの出発点(根)を持つ有向非巡回グラフであり、計算機上で論理関数をコンパクトに表現できるデータ構造である。ロボットの行動則を BDD で適切に表現できれば、動作を概括できると考える。概括的な動作を出力する行動則が得られれば、多少形状の異なるロボットでも同じグラフを用いて制御できると期待される。しかし、一般的な BDD では、論理関数と

同様に 0 か 1 しか出力できず、ロボットの行動出力の種類を表現するには到底足りない。さらに、動作に対応した BDD の最適な表現を見つけることすら難しいと考えられる。そこで本稿では、 n -BDD[5] をロボットの行動則獲得に用いる。

n -BDD とは、出力を複数持つ BDD (多出力二分決定グラフ: Multi-Terminal Binary Decision Diagram) を遺伝的プログラミングによって進化させる手法である。文献[5]では、 n -BDD を食物連鎖のシミュレーションに応用し、シミュレーション環境内で植物・草食動物・肉食動物の疑似生態系を実現している。この手法は、進化の過程では、BDD における変数順序の変更は認められておらず、そのため文献[5]では疑似生態系の構築のために有用な変数順序を評価していた。この手法をロボットの制御にそのまま利用しようとする、ロボットの状態・行動空間は高次元であることが問題となる。すなわち、変数順序が固定されている効率的な BDD の形状は複雑になるため、無駄な進化を必要とすることになり、さらには行動則そのものの獲得できない可能性がある。そこで本稿では、この問題を回避するために変数順序を動的に決定できる新しい遺伝的操作を提案する。また、ロボットの行動則獲得のために交叉演算を導入する。これらの手法により、柔軟な進化を行わせることができるだけでなく、行動空間の次元数の爆発を防ぐことができる。

ロボットの行動則を獲得する方法としては、強化学習を用いる方法[6]～[8]や、遺伝的プログラミングを

[†] A New Dynamic Insertion Operation for n -BDD - Applying to Obtaining Robot Controller -
Masayoshi KANO and Hidenori ITOH

^{*1} 中京大学
Chukyo University

^{*2} 名古屋工業大学
Nagoya Institute of Technology

用いる方法[9, 10], 決定木学習を用いる方法[11]などがある。 n -BDDは, 離散状態行動空間を対象とするため, これらの手法と比べてシンプルな行動則を実現するところに特徴がある。また, n -BDDでは, 出力変数の数を任意に決定可能であり, 複雑な動作に対しても複数の行動出力によって, ある程度の対応ができると考える。BDDのような木構造を用いた関連研究として, 決定木学習C4.5を用いた手法が文献[11]で提案されている。この論文では, 決定木学習で得られた安定行動指標によって環境の変化に対して柔軟に対応できることが示されている。しかし, この手法を用いるためには, 正事例と負事例を事前に用意し, 特定の時刻ごとに成功指標を抽出する必要がある。事前に事例を準備することについては, 負事例は容易に集めることができると考えるが, 歩行動作などの複雑な動作の正事例を複数集めることは非常に困難である。また, 得られた成功指標を満たすような動作を生成する制御器を別で作る必要もあり, こちらの制御が動作の成功の可否を大きく左右するという問題もある。これに対して, n -BDDでは, 動作は事前に用意するのではなく, 適切な動作を出力できるように決定グラフの形状を進化させる。つまり, 事前の動作収集が不要であり, かつ, 動作のための行動則を動的に獲得できるという利点がある。さらに, BDDの変数の配置については, 1)出力を制御する力の強い入力(根に近い深さ)に配置される, 2)局所計算性のある入力変数グループは近い深さに配置される, という傾向がある[12, 13]。この特徴を利用すれば, 獲得すべき動作の暗黙知を表現する一助になると考える。

本手法の有効性を確認するために2種類のロボットを用いた進化シミュレーション実験を行う。1つは, LEGOマインドストームを用いたライントレースのシミュレーション実験で, n -BDDによって適切な行動則が獲得できるかどうかを評価する。もう1つは, ヒューマノイドロボットを用いたシミュレーション実験で, 椅子からの立ち上がり動作の行動則を獲得させる。このシミュレーションを通じて既存手法と提案手法を比較し, 複雑な問題に対して提案手法のほうがあることを確認する。さらに, 歩行の行動則獲得シミュレーションの結果についても報告する。

2. 二分決定グラフ

二分決定グラフ(BDD)は, 論理関数の表現方法の1つであり[14, 15], LSIの設計や, CADシステムのモデル検査などに利用されている。図1にBDDの例を示す。BDDは, 2種類の節点(変数節点: nonterminal vertex, 定数節点: terminal vertex)と枝で構成され

図1 BDDとMTBDDの例

る。枝には, 変数節点の左下から出る0枝と, 右下から出る1枝の2種類がある。図1では, 変数節点は丸で表されており, 各変数節点は割り当てられた変数を処理する。すなわち, 変数節点に書かれた変数の値が0のときは0枝をたどり, 1のときは1枝をたどる。変数の処理は深さ0の位置, すなわちグラフの根に配置された変数節点から順に行われ, 最終的に1つの四角で表される節点に到達する。この節点が定数節点であり, 論理関数の出力となる。

図1のBDDは3つの変数(x_1, x_2, x_3)を処理することができる。例えば, 同図(a)に $(x_1, x_2, x_3) = (0, 1, 1)$ を入力したときは, まず深さ0の位置に配置された x_1 の変数節点において0枝をたどり, つぎに深さ2に配置された x_3 の変数節点で1枝をたどり, 出力1を得ることができる。同様にして出力1を得る組合せをすべて考えれば, このBDDは $f = x_1x_2 + x_3$ を表現していることがわかる。

多出力二分決定グラフ(Multi-terminal BDD: MTBDD)は, BDDを拡張したものであり, 多出力を持つところに特徴がある。例えば, 図1(b)のMTBDDは n 個の出力を持つ。文献[5]では, このMTBDDを構築するために遺伝的プログラミングを導入している。文献[5]では, MTBDDのことを n -BDDと呼んでいるが, 本稿では, 一般的なMTBDDと, 進化プログラミングを導入したMTBDDを区別するために, 後者を n -BDDと呼ぶことにする。次節では, n -BDDについて概説する。

3. n -BDD

図2に n -BDDの遺伝的操作を示す。 n -BDDの遺伝的操作は3つからなる。

節点追加(insertion)は, 新たな変数節点をランダムに選んだ枝の上に追加する。追加した変数節点の0枝か1枝のどちらかは, 追加前に接続されていた節点に接続する。もう1つの枝は, 任意の定数節点または,

図2 n-BDDの遺伝的操作

追加された変数節点よりも下位の任意の変数節点に接続する。

節点削除(deletion)はランダムに選んだ変数節点を削除する。削除した変数節点に接続していた枝は、削除した変数節点のいずれかの枝が接続していた節点を指すように設定する。この処理は根に対しても適用される。

突然変異(mutation)は、ランダムに選ばれた変数節点の持つ0枝または1枝の指す向きを変更する。ただし、新たに指す節点は、任意の定数節点または、選択された変数節点よりも下位の任意の変数節点とする。

これらの3つの遺伝的操作を繰返し実行することで、BDDの形状を変化させ、与えられた問題を解決するグラフ構造を獲得することができる。しかし、従来のn-BDDでは、変数順序は固定されているため、効率的なグラフ構造を得るためには、事前に変数順序を検討する必要がある。ロボットの制御では、少なくともロボットの有するモータの数だけ変数が必要と考えられるので、この問題は致命的である。そこで本稿では、変数順序を動的に変更できる変数節点追加アルゴリズムを提案し、これをロボット制御に応用する。さらに、ロボット制御のために交叉演算を導入できるn-BDDの配置方法を考える。

4. n-BDDによるロボット制御手法

4.1 新しい節点追加アルゴリズム

n-BDDの変数順序の問題を回避するために、本稿では、n-BDDの進化中に変数順序を動的に決定する変数節点追加アルゴリズムを提案する。図3に提案するアルゴリズムを示す。

本アルゴリズムは、変数順序を動的に決定しながら変数節点を追加するアルゴリズムである。まず、追加する変数節点の持つ変数の識別子 i をランダムに決定する(図3内2行目参照)。変数 x_i が根の変数節点として使用されている場合、根は1つの節点しか持つことができないので、再度追加する変数を選択し直す(同、4, 5行目参照)。つぎに、 x_i が、n-BDD内で

Insert vertex

Input S : a set of variables already used in n-BDD.

```

1. begin
2.   select a suffix  $i \in \{1, \dots, n\}$  of variable at random;
3.   create a new nonterminal vertex  $V(x_i)$ 
       which variable is  $x_i$ ;
4.   if  $x_i$  is used as the variable of root node
5.   then go to 2.;
6.   if  $x_i \in S$ 
7.   then begin
8.      $R = \{e | e \text{ is an edge on which } V(x_i) \text{ can insert}\}$ ;
9.     if  $R \neq \phi$ 
10.    then begin
11.      select an edge  $e \in R$  at random;
12.      insert  $V(x_i)$  on  $e$ ;
13.    end;
14.    else go to 2.;
15.  end;
16.  else insert  $V(x_i)$  on an edge selected at random;
17. end.
```

図3 動的節点追加アルゴリズム

で使われているかどうかを調べる。 x_i が使用されている場合、同じ深さに節点 $V(x_i)$ を追加しなければならないため、節点 $V(x_i)$ を追加可能な枝があるかどうかを調べ、それらの枝 e を要素とする集合 R を作る(同、8行目参照)。もし、 $R \neq \phi$ ならば、枝 $e \in R$ をランダムに選択し、その枝上に節点 $V(x_i)$ を追加する(同、11, 12行目参照)。 $R = \phi$ ならば、追加不可能なので、再度追加する節点の変数を選択し直す(同、14行目参照)。 x_i が使用されていない場合、節点 $V(x_i)$ をランダムに選択した枝上に追加する(同、16行目参照)。

本アルゴリズムでは、節点追加時にその節点が使用されていない場合は任意の枝上に追加できるため、変数順序にとらわれないBDDを構築することができる。これにより、膨大な変数の数を必要とする問題に対しても柔軟に対応できると考えられる。ただし、本アルゴリズムでは、ランダムに選択した節点が追加不可能な場合に繰返しして節点を選択するが、この繰返し処理に停止条件が存在しない。本アルゴリズムを使用する場合には、節点選択の部分に何らかの停止条件を設ける必要がある。

4.2 ロボットの行動則獲得のための交叉演算の導入

ロボットの行動則を少ない世代数で獲得させるために、 n -BDDの配置について考える。BDDは、論理関数をコンパクトな形式で表現できる。しかし、 n -BDDにおいては、表現形式がコンパクト化されるために、出力変数の数が膨大になると、節点間の接続が複雑化してしまい、1つの n -BDD内に適切な入出力関係を構築することが難しくなる。ロボットは一般的に、多くの制御機構を有するため、制御の状態空間は高次元となる。そのため、1体のロボットに1つの n -BDDを使用するだけで、適切な行動則を実現する n -BDDは非常に複雑な表現形式になることが予想され、これが進化学習の妨げになると考える。そこで本稿では、 n -BDDをロボットの制御機構ごとに配置することを提案する。図4に配置例を示す。この方法を用いれば2つの利点が得られる。1つは、交叉が可能となる点である。具体的には、2体のロボット間において、対応するモータに配置された n -BDDを交換するだけである。ここで、交叉による n -BDDの交換は、異なるモータ間では行われないことに注意されたい。この処理により、多様な個体(ロボット)が次世代に引き継がれるため、効率的な進化が行われると考える。もう1つは、出力変数の組合せ爆発を防ぐことができる。1つの n -BDDで1体のロボットの制御則を獲得したほうが、 n -BDDの上位に配置された変数節点がロボットの状態を適切に集約するため、モータ出力の協調が促され、目的の行動則をうまく獲得できる可能性がある。しかし、各制御機構に n -BDDを配置しても、交叉により多様な子孫が発生するので、それぞれの n -BDDが独立で良好な出力を学習できると考える。

5. 進化シミュレーション実験

提案手法の有効性を確認するために、3つの進化シミュレーション実験を行った。まず5.1節で、LEGOマインドストームを用いたシミュレーション結果を報告する。ここでは、光センサ1つを有するライント

図4 n -BDDの配置例

レーサーを用いて、 n -BDDによって適切な行動則が獲得できるかどうかを評価する。つぎに5.2節で、ヒューマノイドロボットを用いて、椅子からの立ち上がり動作の行動則を獲得させるシミュレーション実験を行う。このシミュレーションを通じて既存手法と提案手法を比較し、複雑な問題に対して提案手法のほうがあることを確認する。さらに、5.3節で、ヒューマノイドロボットの歩行行動則の獲得シミュレーションの結果についても報告する。

5.1 ライントレーサーの行動則獲得シミュレーション

図5に示したコースを考える。この環境では、白地の背景に黒色の線でコースが描かれている。ロボットは光センサを1つ有しており、このセンサで床の色を判断し、行動を選択する。表1に、ロボットのセンサによって感知される床の色の種類を示す。同表において、Blackは黒線上、Grayは白い床と黒線との境界上、Whiteは白の床の上にセンサがきたときの反応値である。それぞれの反応値に対応した変数の組合せを n -BDDに入力し、出力された結果をもとにロボットは行動を行う。表2にロボットの動作を示す。本実験では、モータ出力値として3つのパターンを考えた。 n -BDDは2つのモータそれぞれに配置した。この問題では、ロボットはGrayの床をトレースすることで、適切にコースをトレースできるようになる。

図5 ライントレースのコース

表1 センサ反応値と対応する変数値

Sensed color	(x_0, x_1)
Black	(0, 1)
Gray	(0, 0)
White	(1, 0)

表2 行動の種類

Terminal vertex	Motor output
a_0	-2 (backward)
a_1	0
a_2	2 (forward)

行動則獲得シミュレーションのために、本実験では、LEGOマインドストームのシミュレータGURS[16]を用いた。適合度としては、ロボットが灰色の床上であり、かつ、少なくとも1つのモータが動作している場合に1を与えた。また、黒線上の場合には0.5を与えた。それ以外のときは、0とした。個体数は一世代30個体とした。遺伝的操作の割合は、交叉、節点追加、突然変異、節点削除をそれぞれ、0.5, 0.3, 0.15, 0.05とした。進化の戦略は、上位5個体をエリート戦略で次世代に残し、その他の個体については、適合度に比例したルーレット戦略で選択して遺伝的操作を施したものを次世代に残すものとした。以上のパラメータは経験的に決定した。また、第1世代の個体は、ランダムに生成された1つの変数節点のみ持つものとした。

図6に適合度の推移を、図7に第50世代の最適個体の動作を示す。適合度の最大値は約500[☆]である。図7より、ロボットが黒線と白い床の境界をトレースしていることが見てとれる。このときの、ロボットの行動則を図8に示す。また、この行動則によるロボットの挙動を表3にまとめる。これらの図表からわかるように、ロボットはセンサの反応値が黒のときは左に旋

(a) 左モータの n -BDD (b) 右モータの n -BDD

図8 獲得された行動則

表3 ロボットの挙動

Color	Left motor	Right motor	Movement
Black	0	2	Turn left
Gray	2	2	Go straight
White	2	-2	Spin right

回、灰のときは直進、白のときは右に回転するように行動する。この結果は、ロボットの右手側に黒線があるときにはそれをトレースできる動作であり、本実験のライントレースの設定では最適な行動則である。

5.2 ヒューマノイドロボットの椅子からの立ち上がり行動則獲得シミュレーション

本節では、高次元状態空間問題として、ヒューマノイドロボットの行動則獲得についてシミュレーションする。ロボットの動作としては、椅子からの立ち上がり動作を考える(図9)。ヒューマノイドロボットとしては、図10に示す富士通オートメーション社製ロボットHOAP-1 (Humanoid for Open Architecture Platform)を用いる。HOAP-1は、身長48[cm]、重量6 [kg]であり、20自由度、角度センサ、姿勢センサ(加速度センサ、ジャイロセンサ)、足底感圧センサを持つ。

進化シミュレーションのために、Open Dynamics

図6 適合度 (20回の平均)

図7 進化結果 (第50世代)

図9 椅子からの立ち上がり動作

図10 HOAP-1

☆ ノイズを含むシミュレータのため、同一の n -BDD でも試行によって得られる適合度が変化する。

Engine[17]を用いた。ロボットは以下の情報をセンサにより感知する。

$$x(t) = (x_0^W, x_1^W, x_0^K, x_1^K, x_0^A, x_1^A, x_0^B, x_1^B). \quad (1)$$

ここで、 x_i^W , x_i^K , x_i^A はそれぞれ、腰、膝、足首の角度を表現する変数であり、 x_i^B は体の傾きを表す変数である(図9)。これらの変数値は、現在の関節の角度値から表4のように決定した。同表において、 $\theta_{\min}$, $\theta_{\max}$ はモータの可動範囲の限界値であり、それぞれ $\theta_{\min}^W = -80.0[\text{deg}]$, $\theta_{\max}^W = 70.0[\text{deg}]$, $\theta_{\min}^K = 0.0[\text{deg}]$, $\theta_{\max}^K = 120.0[\text{deg}]$, $\theta_{\min}^A = -60.0[\text{deg}]$, $\theta_{\max}^A = 60.0[\text{deg}]$, $\theta_{\min}^B = -60.0[\text{deg}]$, $\theta_{\max}^B = 60.0[\text{deg}]$ である。ロボットの行動出力は、表5とした。ロボットの制御時間間隔は、 $\Delta t = 0.01[\text{s}]$ とした。

1 試行は、ロボットが倒れるか10秒経過したときに

表4 角度情報と変数値の対応付け

Sensed degree	(x_0, x_1)
$\frac{3}{4}(\theta_{\max} - \theta_{\min}) \leq \theta < \theta_{\max}$	(1, 1)
$\frac{1}{2}(\theta_{\max} - \theta_{\min}) \leq \theta < \frac{3}{4}(\theta_{\max} - \theta_{\min})$	(1, 0)
$\frac{1}{4}(\theta_{\max} - \theta_{\min}) \leq \theta < \frac{1}{2}(\theta_{\max} - \theta_{\min})$	(0, 1)
$\theta_{\min} \leq \theta < \frac{1}{4}(\theta_{\max} - \theta_{\min})$	(0, 0)

表5 行動の種類

Terminal vertex	Motor output
a_0	-20.0 [deg/sec](backward)
a_1	0
a_2	20.0 [deg/sec](forward)

表6 比較実験に用いた手法。Method(a)が提案手法である。Method(c)ではロボット1体に対して1つの n -BDDしか準備しないので交差はできない。

手法	交差	n -BDD 配置
Method(a)	あり	複数 (モータ毎)
Method(b)	なし	複数 (モータ毎)
Method(c)	—	1つ

終了するものとした。適合度は、試行中の各時刻 t におけるロボットの胸の位置 $y(t)$ [m]を合算した値 ($\text{Fitness} = \sum_t y(t)$)とした。遺伝的操作のパラメータは5.1節と同様とした。

図11に第1000世代の最良個体による動作結果を示す。ロボットがバランスを取りながら立ち上がっている様子が見てとれる。この動作を出力する n -BDDを図12に示す。

つぎに、提案手法の進化の効率について調べる。表6に比較手法を示す。Method(a)はモータ毎に n -BDDを配置して交叉を行う手法(提案手法)、Method(b)はモータ毎に n -BDDを配置するが交叉はしない手法、Method(c)はロボット1体に対して1つの n -BDDを用いる手法である。これらの手法では、節点は動的に追加される。なお、Method(b)では、交叉以外の遺伝的操作の割合を等しくするために、節点追加、突然変異、節点削除の割合をそれぞれ、0.6, 0.3, 0.1とした。

Method(a)～(c)の適合度の推移を図13に示す。同図より、各関節に n -BDDを配置したMethod(a), (b)のほうがMethod(c)よりも高い適合度を獲得していることがわかる。また、Method(a)とMethod(b)の結果を見ると交叉を導入したMethod(a)のほうがより高い適合度を獲得できていることがわかる。これらの結果から、各関節に n -BDDを配置して、交叉演算を導入したほうが高い適合度を獲得できることが確認できた。

つぎに、変数節点の動的追加の効果について調べる。5.1節のライントレーサーの実験では、入力が2変数なので、動的追加による効果は期待できない。しかし、本実験については、追加された変数の順位が変化することによって、適切な行動則を獲得していることが予想される。変数の順位づけについては、文献[13]において、BDDのサイズと計算時間に対して影響があることが示されており、また、文献[5]において、 n -BDDの進化の効率に影響があることが示されている。ここでは、目的の動作を表現するための適切な指標(変数順序)が、変数節点の動的追加によって抽出できるかどうかについて考察する。

図11 進化結果 (第1000世代)

図12 獲得された n -BDD (椅子からの立ち上がりを実現する行動則)

図13 比較結果 (20回平均)

まず, Method(a)の条件で100回の実験を行った。つぎに, 適合度318以上の動作を出力する n -BDD を立ち上がりの行動則と定義し, 行動則を獲得した個体ごとの n -BDD の形状を比較した。

100回の実験のうち, 行動則は, 平均 340 ± 256 世代で41回獲得されていた。行動則獲得世代にばらつきがあったため, 200世代までに行動則を獲得した実験, すなわち, 動作を概括する特徴を抽出していると考えられる n -BDD に注目した。

200世代までに行動則を獲得できた実験は17回であった。これら実験の n -BDD に限ると, 行動則獲得時の変数節点の総数の平均は, 腰, 膝, 足首の順に,

6.1 ± 3.0 , 5.9 ± 3.9 , 4.4 ± 3.5 であり, また, 平均 108 ± 45.9 世代で立ち上がりの行動則を獲得していた。ここで, 全100実験における第108世代の変数節点の総数の平均は, それぞれ 5.3 ± 2.7 , 5.2 ± 2.7 , 6.1 ± 2.5 であった。この結果を見ると, 利用された変数節点の総数については, 第200世代までに行動則を獲得した実験の n -BDD と, その他の実験との間に差は見られない。しかし, 全100実験における第108世代の獲得適合度の平均は279であり, 適合度については明らかな差がある。これは, 出力に対して影響力の強い変数が上位に配置されたためだと考える。

この17回の実験における各変数の使用状況を表7～9に示す。ここで, 変数の使用回数と, 変数の使用された深さの平均について, ケンドールの順位相関係数を求めたところ, 負の相関があることがわかった(腰 $\tau = -.67$, $p < .05$, 膝 $\tau = -.76$, $p < .01$, 足首 $\tau = -.59$, $p < .05$)。この結果は, よく利用される変数ほど, 根に近い位置に配置されることを示している。そこで, 変数順序を表10のように固定してBDDを進化させる実験を行った。Method(d)は使用頻度の低かった変数を上位に, Method(e)は使用頻度の高かった変数を上位に設定した手法である。Method(e)のほうがMethod(d)よりも高い適合度を獲得することが予想される。実験では, 変数順序を固定にした以外は, Method(a)の実験と同条件とした。実験結果を図14に示す。同図には比較のためにMethod(a)のグラフも示

表7 17実験における腰の n -BDDにおける変数配置状況（使用された変数の種類：4.6 ± 1.7種類）

変数	使用回数	深さの平均
x_0^W	5	4.0 ± 1.4
x_1^W	7	2.7 ± 1.9
x_0^K	6	2.3 ± 1.2
x_1^K	15	1.3 ± 1.8
x_0^A	12	2.2 ± 1.7
x_1^A	12	2.1 ± 1.9
x_0^B	10	2.4 ± 1.8
x_1^B	10	1.4 ± 1.5

表8 17実験における膝の n -BDDにおける変数配置状況（使用された変数の種類：4.5 ± 1.9種類）

変数	使用回数	深さの平均
x_0^W	11	1.6 ± 1.4
x_1^W	14	1.0 ± 1.1
x_0^K	13	1.2 ± 0.8
x_1^K	9	2.9 ± 2.4
x_0^A	7	4.1 ± 1.8
x_1^A	6	3.2 ± 2.8
x_0^B	8	2.6 ± 1.7
x_1^B	8	2.4 ± 1.9

表9 17実験における足首の n -BDDにおける変数配置状況（使用された変数の種類：3.4 ± 1.8種類）

変数	使用回数	深さの平均
x_0^W	4	2.0 ± 1.4
x_1^W	6	1.5 ± 1.4
x_0^K	6	1.8 ± 1.7
x_1^K	4	3.3 ± 1.7
x_0^A	13	0.8 ± 1.1
x_1^A	12	1.3 ± 1.4
x_0^B	7	2.4 ± 1.9
x_1^B	5	1.6 ± 1.1

表10 変数の固定順序

手法	モータ	変数順序（左ほど根に近い）
Method(d)	腰	$x_0^W - x_0^K - x_1^W - x_0^B - x_1^A - x_0^A - x_1^K$
	膝	$x_1^A - x_0^A - x_0^B - x_1^B - x_1^K - x_0^W - x_0^K - x_1^W$
	足首	$x_1^K - x_0^W - x_1^B - x_0^K - x_1^W - x_0^B - x_1^A - x_0^A$
Method(e)	腰	$x_1^K - x_0^A - x_1^A - x_1^B - x_0^B - x_1^W - x_0^K - x_0^W$
	膝	$x_1^W - x_0^K - x_0^W - x_1^K - x_1^B - x_0^B - x_0^A - x_1^A$
	足首	$x_0^A - x_1^A - x_0^B - x_1^W - x_0^K - x_1^B - x_0^W - x_1^K$

した。同図より、提案手法のMethod(a)と比較し、Method(e)は高い適合度を、Method(d)は低い適合度を獲得している。これは、Method(e)では、ロボットの行動を適切に集約する変数節点が n -BDDの上位に配置されるため、目的の行動則を獲得できる可能性が高まったためだと考えられる。

図14 変数順序を固定したときの適合度の推移（20回平均）

5.3 ヒューマノイドロボットの歩行行動則の獲得シミュレーション

5.2節では、ロボットの動作のうちでも過渡的な動作の行動則獲得についてシミュレーションした。本節では、周期的な動作として、ロボットの歩行動作の行動則獲得についてシミュレーションする。

1試行は、ロボットが倒れるか10秒経過したときに終了するものとした。適合度は前進した距離[m]とした。遺伝的操作のパラメータは、一世代の個体数を100個体、エリート戦略で残される個体数を上位10個体、その他を5.1節と同様に設定した。モータは、図15に示す9つのモータを可動させた。ここで、 θ^{W1} は腰を前後方向、 θ^{W2} は腰を左右方向、 θ^{A1} は足首を前後方向、 θ^{A2} は足首を左右方向にそれぞれ可動させるモータである。腰の左右方向については、同じ行動を出力させるため、1つのモータとして取り扱った。各モータには5.2節と同様に2つの変数を割り当てたので、体の傾きの変数 θ^B とあせて、変数の数は20となる。ただし、それぞれのモータの可動範囲の限

図15 n -BDDの配置

界値は, $\theta_{\min}^{W1} = -35.0[\text{deg}]$, $\theta_{\max}^{W1} = 0.0[\text{deg}]$, $\theta_{\min}^{W2} = -15.0[\text{deg}]$, $\theta_{\max}^{W2} = 15.0[\text{deg}]$, $\theta_{\min}^K = 35.0[\text{deg}]$, $\theta_{\max}^K = 60.0[\text{deg}]$, $\theta_{\min}^{A1} = -45.0[\text{deg}]$, $\theta_{\max}^{A1} = -10.0[\text{deg}]$, $\theta_{\min}^{A2} = -15.0[\text{deg}]$, $\theta_{\max}^{A2} = 15.0[\text{deg}]$, $\theta_{\min}^B = -5.0[\text{deg}]$, $\theta_{\max}^B = 5.0[\text{deg}]$ とした. ロボットの行動出力は, 表11とした. 本実験では, 行動出力に速度変化量を用いたが, これは, n -BDDは離散的な行動出力しかできないため, 角度変化量による制御では, なめらかな周期的なモータ出力ができないと考えたためである. また, 今回はシンプルな n -BDDでどの程

度の歩行の行動則が獲得できるか確認するために, 出力(変数節点)の数を3とした. ロボットの制御時間間隔は, $\Delta t = 0.01[\text{s}]$ とした.

獲得された行動則による動作の一例を図16に示す. また, 適合度の推移を図17に示す. 第1世代の最良個体は, 前に転倒するだけの動作であり, このときの適合度は0.29[m]であった. 第10000世代では, 左右に足を踏み出す行動則を獲得し, 1.08[m]まで前進することができた. このときの各関節の角度変化を図18に示す. 同図より, 腰の前後方向(θ^{W1})については, 左右のモータで対称の出力が得られていることがわかる. 歩行動作においては, θ^{W1} の周期性がもっとも重要と考えられる. 今回の実験では, θ^{W1} が周期的な出力を行う行動則を獲得できたことが, 歩行動作につながっていると考えられる. θ^{W1} の周期的な動作については, 図16からも, 交互に足が踏み出されていること

表 11 行動の種類

Terminal vertex	Motor output
a_0	-1.0 [deg/sec ²](backward)
a_1	0
a_2	1.0 [deg/sec ²](forward)

図 16 進化結果 (第 10000 世代)

図 17 最良個体の適合度の推移

図 18 各関節の角度の時間変化

から見てとれる．この他，進化の結果，図19のように人とは異なる動作で前進する行動則も獲得することができた．これらの動作では，十分に歩行できておらず最終的に転倒してしまう．今回の進化シミュレーションでは，シンプルな行動出力でどの程度複雑な動作が行えるか確認するために，出力数を3として進化シミュレーションを行ったが，ある程度の周期的な歩行を実現することができた． n -BDDの入出力変数の数を増すなどすれば，歩行動作の安定性を高められると考える．

6. おわりに

本稿では，変数順序を動的に決定できる新しい遺伝的操作を提案した．また，ロボットの行動則獲得のための交叉演算についても提案した．これらの手法により，柔軟な進化を行わせることができるだけでなく，行動空間の次元数の爆発を防ぐことができる．進化シミュレーション実験では，ライントレーサーおよびヒューマノイドロボットの行動則の獲得実験を行い，提案手法の有効性を確認した．

今後の課題としては，まず， n -BDDの入出力変数の数を増すなどして，より，柔軟な行動則が獲得できるか調査することがあげられる．また，行動数を増やす方法として，複数の n -BDDの出力をアンサンブルしてモータ出力を得ることも考えている．つぎに，本稿では， n -BDDの変数節点は2つの枝しか持つことができないため，1つの関節情報を複数の変数に割り当てたが，同一の関節を制御する変数を独立に取り扱うことは，問題を複雑化しているともいえ，これを解決する必要がある．解決方法としては，1つの変数節点から複数の枝を出力できるMulti-valued Decision Diagram (MDD) [18]を導入することがあげられる．つぎに，遺伝的操作の確率を変化させた際の性能の変化を見ることがあげられる．すなわち，進化初期では変数節点の数が少ないので，節点追加の使用確率を高めたほうがよいと考えられ，進化後期では， n -BDDは十分に大きくなっているため，交叉や突然変移などの使用確率を高くしたほうがより多様性のある子孫が残せるのではないかと考える．

謝辞

本研究の一部は，文部科学省科学研究費補助金若手研究(A) (課題番号20680014)，私学ハイテク・リサーチ・センター補助金，および，人工知能研究振興財団研究助成によった．

図19 人の歩行と異なる進化結果 (第10000世代)

参考文献

- [1] 沢井邦仁: “QRIO”におけるデザインプロセス, 日本ロボット学会誌, Vol.22, No.8 (2004).
- [2] <http://www.ifbot.net/>.
- [3] 神田真司, 村瀬有一, 藤岡和夫: インターネット対応型ロボット: MARON-1, *Fujitsu*, Vol.54, No.4, pp.285-292 (2003).
- [4] 藤田善弘: パーソナルロボット PaPeRo の開発, 計測と制御, Vol.42, No.6 (2003).
- [5] 森脇康介, 横井大祐, 犬塚信博, 伊藤英則: 遺伝的プログラミング技法を用いた多出力二分決定グラフの進化—食物連鎖におけるマルチエージェントの進化シミュレーション, 人工知能学会誌, Vol.14, No.3, pp.477-484 (1999).
- [6] Asada, M., Katoh, Y., Ogino, M. and Hosoda, K.: A humanoid approaches to the goal-reinforcement learning based on rhythmic walking parameters-, *7th RoboCup International Symposium* (2003).
- [7] 中村泰, 佐藤雅昭, 石井信: 神経振動子ネットワークを用いたリズム運動に対する強化学習法, 電子情報通信学会論文誌 D-II, Vol.J87-D-II, No.3, pp.893-902 (2004).
- [8] 森本淳, 銅谷賢治: 強化学習を用いた高次元連続状態空間における系列運動学習—起き上がり運動の獲得—, 電子情報通信学会論文誌 D-II, Vol.J82-D-II, No.11, pp.2118-2131 (1999).
- [9] 玉秀列, 宮下和雄, 長谷和徳, 荒川佳樹: GPを用いた2足歩行モデルとその3次元ビジュアルイゼーション, 電子情報通信学会技術研究報告 (MVE, マルチメディア・仮想環境基礎), Vol.101, No.390(20011019), pp.13-19 (2001).
- [10] Inoue, Y., Tohge, T. and Iba, H.: Learning to Acquire Autonomous Behavior - Cooperation by Humanoid Robots, *Genetic and Evolutionary Computation Conference (GECCO2004)*, pp.590-602 (2004).
- [11] 桑山清丈, 加藤昇平, 伊藤英則: 経験からの知識獲得に基づいたヒューマノイドの運動制御, 日本ロボット学会誌, Vol.25, No.3, pp.78-85 (2007).
- [12] 湊真一: 計算機上での BDD の処理技法, 情報処理, Vol.34, No.5, pp.593-599 (1993).
- [13] 柳谷雅之: 組合せ最適化問題の BDD による解法, 情報処理, Vol.34, No.5, pp.617-623 (1993).

- [14] Akers, S. B.: Binary Decision Diagrams, *IEEE Trans. on Comp.*, Vol.C-27, No.6, pp.509-516 (1968). Vol.2925, pp.296-338 (2004). (2007年8月24日 受付)
- [15] Bryant, R. E.: Graph-Based Algorithms for Boolean Function Manipulation, *IEEE Trans. on Comp.*, Vol.C-35, No.8, pp.677-691 (1986). (2008年5月12日 採録)
- [16] Rasmussen, J.: The General Unified Robot Simulator (GURS). <http://www.daimi.au.dk/%7Erepsej/thesis.html>. [問い合わせ先]
〒470-0393 豊田市貝津町床立101
中京大学 情報理工学部
加納 政芳
TEL: 0565-46-6910
FAX: 0565-46-1299
E-mail: mkanoh@sist.chukyo-u.ac.jp
- [17] Smith, R.: Open Dynamics Engine (ODE). <http://opende.sourceforge.net/ode.html>.
- [18] Miner, A. and Parker, D.: Symbolic Representations and Analysis of Large Probabilistic Systems, *LNCS*,

 著 者 紹 介

 か の う まさよし
 加納 政芳 [正会員]

1999年名古屋工業大学工学部知能情報システム学科卒業。2004年名古屋工業大学大学院工学研究科博士後期課程修了。同年中京大学講師。博士(工学)。知能・感性ロボティクス、探索問題などに興味を持つ。2006年日本感性工学会技術賞、電子情報通信学会、情報処理学会、人工知能学会、日本ロボット学会、電気学会、日本感性工学会、日本顔学会各会員。

 い とう ひでのり
 伊藤 英則 [非会員]

1974年名古屋大学大学院工学研究科博士課程電気電子専攻満了。工学博士号取得。1974年日本電信電話公社横須賀研究所勤務。1985年(財)新世代コンピュータ技術開発機構出向。1989年名古屋工業大学大学院教授。情報工学専攻。数理言語理論、計算機ネットワーク通信、OS、人工知能などの研究開発に従事。2006年日本感性工学会技術賞、電子情報通信学会、情報処理学会、人工知能学会、日本ロボット学会、形の科学会、日本感性工学会、IEEE各会員。

A New Dynamic Insertion Operation for n -BDD - Applying to Obtaining Robot Controller - by Masayoshi KANO and Hidenori ITOH

Abstract :

In this paper, we propose an evolution-based method for controlling robots using evolutive binary decision diagrams (BDDs). BDDs are a way of representing a logic function using a directed graph that consists of binary switches having one input and two outputs, and are mainly used to design LSIs. Moriwaki et al. proposed n -output BDDs (n -BDDs) that can evolve BDDs, and they applied the n -BDDs to a food chain simulation of evolutionary agents. However, in the evolution method, variable ordering of n -BDDs is fixed, so it cannot be applied to complex problems. In this paper, we introduce a new genetic manipulation to n -BDDs as a way of avoiding this problem of variable ordering, and we apply the new n -BDDs to obtain motion control of robots: LEGO Mindstorms and the humanoid robot HOAP-1. We also demonstrate that our method is capable of providing good performance in computer simulations.

Keywords : Binary decision diagrams, genetic programming, robot control, dynamic allocation of vertex

Contact Address : **Masayoshi KANO**

School of Information Sciences and Technology, Chukyo University
 101 Tokodachi, Kaizu-cho, Toyota 470-0393, JAPAN
 TEL : 0565-46-6910
 FAX : 0565-46-1299
 E-mail : mkanoh@sist.chukyo-u.ac.jp