F-037

遺伝的プログラミングを用いた論理回路合成における出力分解の適用
A GP-based Logic Circuit Synthesis Method with Output Decomposition

松井 小百合 †　石田 康太郎 †　新井 浩志 †
Sayuri Matsui　Koutarou Ishida　Hiroshi Arai

1. はじめに

本報告では，入出力信号のサンプルを教師データとして，遺伝的プログラミング（GP：Genetic Programming）を用いて自動的に論理回路を合成する手法を提案する．我々の従来の研究[1]では 5 ビットバイナリカウンタよりも大きい回路を合成することは非常に困難であった．そこで本報告では，出力分解という手法を取り入れた結果について報告する．解の探索空間を抑制することにより，7 ビットカウンタなどを合成することに成功した．

2. GP によるハードウェアの合成

GP を適用するためには，論理回路を遺伝子として表現する必要がある．論理回路は組合せ回路と順序回路に分けられるが，本研究では順序回路を対象とする．順序回路は一般的に有向サイクリックグラフで表現される．GP の遺伝子を有向サイクリックグラフで表すと，交叉などの遺伝的操作を定義することが困難になる．そこで本手法では，順序回路を FF と組合せ回路の木に分解して，組合せ回路の木を遺伝子とみなして GP 操作を行う．

論理回路全体は，Moore 型状態遷移機械をもとに構成する．図 1 に論理回路の例を示す．論理回路を構成する組合せ回路は 2 つの部分に分かれている．出力に直結する組合せ回路の木を出力木，DFF 群の入力に繋がっている組合せ回路の木を中間木と呼ぶ．図 1 は入力 1 ビット，出力 3 ビットの論理回路の例であり，3 つの DFF と 3 つの中間木，3 つの出力木から構成される．中間木の数は DFF の数と等しく，進化の過程で個数が変化する．これに対して，出力木の数は出力の数と等しいので初期状態から一定である．この中間木と出力木に GP 操作を行うことによって回路の合成を行う．

GP 処理では，まず初期集団を生成し適合度計算を行う．適合度とは，環境に応じて求められている出力と個体の出力がどれだけ近いかを評価した値である．個々の個体に対して，教師データを与えた場合の出力をシミュレーションし，これと模範出力との一致割合を適合度とする．この適合度をもとに個体を選択し，交叉，突然変異などの GP 操作を行うことで新たな個体群を得る．この処理を繰り返すことで論理回路を合成する．

† 千葉工業大学　工学研究科　電気電子情報工学専攻，
Master's Program in Electrical, Electronics and Computer Engineering, Graduate School of Engineering, Chiba Institute of Technology.

3. 出力分解の適用

GP を用いた論理回路合成では，合成しようとする回路の規模が大きくなればなるほど解の探索空間が広くなりすぎてしまい，結果として回路が完成しにくくなる．そこで本報告では，解の探索空間の抑制を目的として出力分解という手法を取り入れる．

出力分解とは複数の出力をもつ論理関数を出力ごとに別々の関数に分解する方法である．進化型 HW をインクリメンタルに合成しようとする研究[2]では，対象とする論理関数のうちで完成している出力を省いていくことで，進化計算の対象とすべき出力を減らしている．具体的には，最初に回路を任意の世代数進化させたのち進化を一時停止させ，出力ごとに完成しているかどうかの評価を行う．完成したとみなされた出力は進化から除外され，他の出力の進化を続ける回路には全く関与しない．しかし，この研究は組合せ回路への適用を前提としているため，ある出力に関する合成は，すでに完成した出力に関する合成結果とは独立に進めることができるという前提がある．

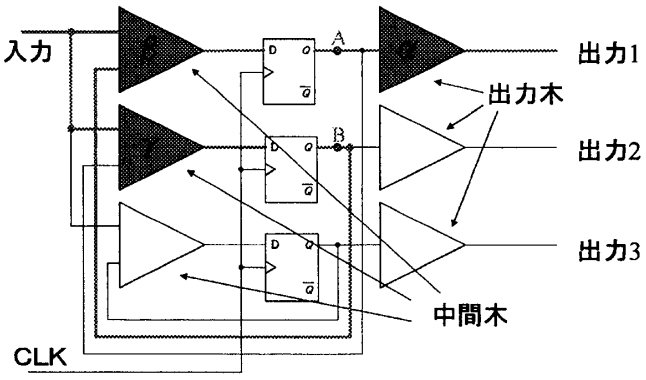

図 1：Moore 型順序回路と出力分解の適用

これに対して本研究の対象である順序回路では，ある中間木の出力を他の中間木，出力木が利用する可能性がある．例えばバイナリカウンタ回路を例とすると，N ビット目の出力は下位のビットの出力に依存しており，N ビット目を単独で合成することは，意味をもたない．そこで提案手法では，完成した出力に使用されている回路の進化は止めるが，回路自体は省かずに残すこととする．これにより出力が完成している回路が，まだ出力の完成していない回路の進化に良い影響を与え，合成の手助けをすることができると考えた．具体的には図 1 の順序回路で出力 1 が完成したと仮定したとき，まず出力 1 に関係する出力木 α と中間木 β，γ は進化の対象から外す．

次に図の A，B 点の信号は完成している出力 1 に関係する中間木 β，γ の出力である．この 2 点の信号は完成していない出力の回路でも使用される可能性がある．そこで，中間木からの出力は残して進化を続けることにより，出力がまだ完成していない回路の合成の一助とする．

4. 結果と考察

出力分解を用いない従来の手法と出力分解を用いた提案手法を用いてカウンタ回路と制御回路を合成した結果を表 1 に示す．

表の成功割合とは，合成を 10 回試行したときに適合度が 100%解の個体を得られた割合を示している．世代数とは，合成に成功したときの進化にかかった世代数の平均の値である．また状態遷移機械の状態の数と遷移の数の和を複雑度と定義し，回路の複雑さの指標とする．

従来手法と提案手法の比較からわかるように，出力分解を適用することによって，より大きな回路の合成に成功していることがわかる．従来手法では，正しい出力を出す中間木，出力木が合成されても GP 操作を行うことによって崩れてしまうことがあった．この結果適合度が大きく下がってしまい，選択されずに消滅してしまう個体が多く存在していた．出力分解は正しい値を出力する部分回路に対しては GP 操作を行わないため，解の探索空間が狭まったことと，正しい値を出力する部分回路が崩れることが無くなったことにより目的の回路が合成し易くなったと考えられる．

しかし，通常のバイナリカウンタは N ビット目の決定に N−1 ビット目の値を利用するという点において，出力分解の効果を得やすいと考えられる．そこで，簡単な制御回路の合成を試みた．提案手法を用いて合成した時の複雑度と世代数との関係を図 2 に示す．この結果より，複雑度が同じ場合，合成にかかる世代数はカウンタ回路の方が少ないことがわかる．複雑度が等しい場合，制御回路はカウンタ回路よりも状態数が少ないが，遷移が複雑であり，ある FF の値の決定に他のすべての FF の値が必要になる場合が多い．このため，より多くの世代数を必要

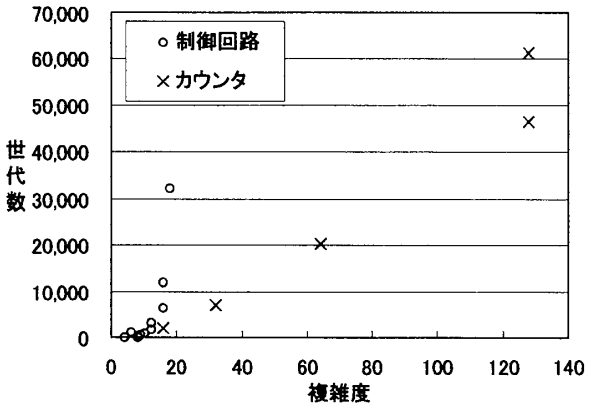

図 2：回路の複雑度と世代数の関係

とすると考えられる．また制御回路の合成において，目的とする制御回路の複雑度が小さい場合の合成にかかる世代数は従来手法と提案手法の間には差は見られない．しかし，複雑度が大きくなると提案手法の効果が現れている．このことから，回路の複雑度がある程度大きい場合には出力分解の効果があるといえる．

5. まとめ

出力分解を適用することで，新たに 5〜7 ビットカウンタや，制御回路の合成に成功した．しかし現時点で合成可能な回路の規模では，まだ実用的用途には不十分である．よって今後は，さらに大規模な回路の合成を可能にする必要がある．

参考文献

[1] 佐古修康 他，“遺伝的プログラミングを用いた順序回路生成手法について”，2005年 電子情報通信学会総合大会，D-18-4, p. 184, 2005.

[2] Tatiana Kalganova et.al., “Bidirectional Incremental Evolution in Extrinsic Evolvable Hardware”, Napier University, IEEE, 2000 The Second NASA/DoD Workshop on Evolvable Hardware.

表 1：カウンタと制御回路の合成結果

回路	複雑度	従来手法		提案手法	
		成功割合[%]	世代数	成功割合[%]	世代数
カウンタ 1 (3bit)	16	100	3,956	100	2,081
カウンタ 2 (4bit)	32	100	19,345	100	6,880
カウンタ 3 (5bit)	64	20	42,875	100	20,440
カウンタ 4 (6bit)	128	0	—*1	60	46,393
カウンタ 5 (7bit)	256	0	—*1	70	61,428
制御回路 1 (4 状態)	8	100	336	100	269
制御回路 2 (4 状態)	12	100	2,979	100	3,215
制御回路 3 (4 状態)	16	100	9,014	100	11,880
制御回路 4 (6 状態)	12	100	1,844	100	1,791
制御回路 5 (6 状態)	16	50	23,870	100	6,480

*1 10 万世代進化させても，100%解の個体が得られない