F_025

遺伝的アルゴリズムを用いた線形変換回路合成における分割最適化に関する考察

Considerations on partitive optimization of linear transform circuit synthesis using genetic algorithm

鈴木 麻衣† 佐々木 孝雄† 豊嶋 久道†
Mai Suzuki Takao Sasaki Hisamichi Toyoshima

1. はじめに

線形変換回路はDFT・DCTなど様々な変換において使われている回路であり，ハードウェアで実現する場合，遅延時間・回路規模をできるだけ小さくすることが要求される．そのために定数乗算器をシフト加算に置き換え，それらを共有化する方法が提案されているが，それらの組み合わせの最適化問題では，係数行列のサイズが大きくなるに従い，計算時間や最適性に問題が生じることが知られている [1].

そこで本研究では，係数行列を分割してそれぞれを最適合成することでその問題を解決することを提案する．その際，分割方法によって遅延時間・回路規模が変化するため，分割方法を組み合わせ最適化問題として扱い，係数行列の分割最適化の検証を行う．

2. 線形変換回路

2.1 複数の定数乗算 (MCM) 回路

複数の定数乗算回路とは，複数の定数 (N 個) と変数間の乗算処理を行う回路で，次式で表される．

$$y_i = a_i x \qquad i = 0, 1, 2, \cdots, N-1 \tag{1}$$

乗算は，加減算とシフトの組み合わせで置き換えることで同等の回路が得られ，演算を共有することで加算器数の削減が可能である．

2.2 線形変換回路

線形変換は，n 個の入力 $x_j (j = 0, 1, \cdots, n-1)$ と m 行 n 列の係数行列 $\mathbf{T}$ との乗算で，m 個の結果 $y_i (i = 0, 1, \cdots, m-1)$ を出力する変換で，次式で表される．

$$\begin{bmatrix} y_0 \\ \vdots \\ y_i \\ \vdots \\ y_{m-1} \end{bmatrix} = \begin{bmatrix} t_{00} & \cdots & t_{0j} & \cdots & t_{0\,n-1} \\ \vdots & & \vdots & & \vdots \\ t_{i0} & \cdots & t_{ij} & \cdots & t_{i\,n-1} \\ \vdots & & \vdots & & \vdots \\ t_{m-1\,0} & \cdots & t_{m-1\,j} & \cdots & t_{m-1\,n-1} \end{bmatrix} \begin{bmatrix} x_0 \\ \vdots \\ x_j \\ \vdots \\ x_{n-1} \end{bmatrix} \tag{2}$$

線形変換回路合成の従来法として，同一入力 x_j に対して係数行列 $\mathbf{T}$ の j 列目の係数列 $\mathbf{T}_j = [t_{0\,j}, \cdots, t_{m-1\,j}]$ をMCM回路として合成し，それらを互いに加算する方法が知られている [2].

式 (2) に対して，入力を $\mathbf{X}$，i 行目の係数列を $\mathbf{T}_i$ とすると，

$$y_i = \mathbf{T}_i \mathbf{X} \tag{3}$$

となり，これをMCM回路の入力を複数に拡張した回路とみなすことができ，MCM回路の一係数を係数行列の係数列とすることで線形変換回路の合成を行う．

†神奈川大学大学院工学研究科電気電子情報工学専攻

3. 合成手法

3.1 分割方法

線形変換回路合成では，係数行列のサイズが大きくなるに従い，計算時間や最適性に問題が生じる．そこで本研究では，係数行列を分割して線形変換回路を合成する．式 (4),(5) に分割数2の場合の例を示す。

$$\left[\begin{array}{c} y_0 \\ y_1 \\ y_2 \\ y_3 \end{array}\right] = \left[\begin{array}{cc|cc} t_{00} & t_{01} & t_{02} & t_{03} \\ t_{10} & t_{11} & t_{12} & t_{13} \\ t_{20} & t_{21} & t_{22} & t_{23} \\ t_{30} & t_{31} & t_{32} & t_{33} \end{array}\right] \left[\begin{array}{c} x_0 \\ x_1 \\ x_2 \\ x_3 \end{array}\right] \tag{4}$$

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} t_{00} & t_{01} \\ t_{10} & t_{11} \\ t_{20} & t_{21} \\ t_{30} & t_{31} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} + \begin{bmatrix} t_{02} & t_{03} \\ t_{12} & t_{13} \\ t_{22} & t_{23} \\ t_{32} & t_{33} \end{bmatrix} \begin{bmatrix} x_2 \\ x_3 \end{bmatrix} \tag{5}$$

分割することにより，それぞれの係数行列が小さくなり，シフト加減算の組み合わせが少なくなるため，計算時間の削減が可能となる．

3.2 合成方法

分割されたそれぞれの線形変換回路は，各入力 x_j をシフト加減算することにより合成される．合成は，係数列 $\mathbf{T}_i$ 単位に行い，シフト加減算を決定するために各 x_j に対応する中間係数列と評価値を用いる．合成における中間値を $s_0 x_0 + \cdots + s_{n-1} x_{n-1}$ とすると，その中間係数列は $\mathbf{S} = [s_0, \cdots, s_{n-1}]$ と表現でき，以下の手順で係数行列を合成する．

(a) 初期値として入力数 n に対して，n 個の中間係数列 $\mathbf{S}_k (k = 0, \cdots, n-1)$ を用意する．
($n = 2$ の場合，$\mathbf{S}_0 = [1\ \ 0], \mathbf{S}_1 = [0\ \ 1]$)

(b) 2つの中間係数列をシフト加減算する．

$$\mathbf{S}' = 2^a \mathbf{S}_{i1} + 2^b \mathbf{S}_{i2} \tag{6}$$

ただし，S_{i1}, S_{i2} の段数は，上限を設定する．

(c) 式 (7) によって評価値を計算し，F が最小となる $\mathbf{S}'$ を中間係数列に追加する．

$$\text{評価式：} F = \sum_{j=0}^{n-1} |t_{ij} - \mathbf{S}'_j| \tag{7}$$

(d) $F \neq 0$ の場合，係数列 $\mathbf{T}_i$ を $\mathbf{T}_i = \mathbf{T}_i - \mathbf{S}_K$ として更新し，$F = 0$ となるまで，(b)〜(d) を繰り返す．

(e) 追加した中間係数列を互いに加算し係数列とする．

4. 最適化

係数行列の分割位置によって遅延時間・回路規模が変化する．また，入力の順序を変化させることにより係数行列が変化し，遅延時間・回路規模が変化するため，分割方法を組み合わせ最適化問題として扱い，このような問題に対し有効性が知られている遺伝的アルゴリズム (GA) を用いて最適化を行う．

また，遅延時間と回路規模はトレードオフの関係にあるので，両方を考慮する必要がある．そこで，加算器数の上限を設定した上で加算段数の最適化を行うことにより，回路規模の制約を考慮した遅延時間最適化を行う．

4.1 遺伝子表現

係数行列の分割方法を遺伝子とし，分割方法は，入力 x_j の順序の入れ換えと分割位置によって表現する．入力の順序は，交叉による致死遺伝子の発生を抑えるため，図1のように入力 x_j の順番を順序表現 [3] を用いて表現する．この方法では，まず入力のリストを作成し，次に来る入力が残りの入力のリストの中で何番目に相当するかの番号を並べる．

さらに，係数行列の分割数 D に対して，$0 \sim (D-2)$ 番目までの遺伝子を分割位置とし，それを遺伝子座として追加する．

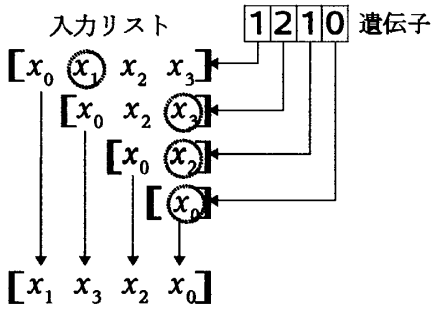

図 1: 順序表現の例

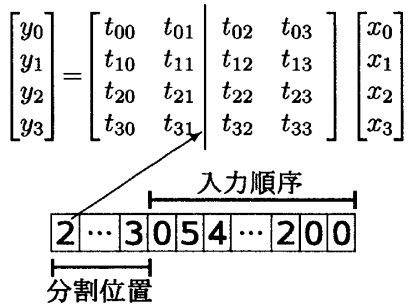

図 2: 遺伝子表現の例

4.2 GA オペレータ

選択淘汰 ：ルーレット選択、エリート保存戦略
交 叉 ：一様交叉
突然変異 ：選ばれた1点をランダムに選んだ遺伝子に置き換える

適応度計算 ：適応度 $= \dfrac{1}{\text{加算段数}}$

- 加算器数の上限を設定し，それを超えた遺伝子の適応度は低く設定する
- エリート保存戦略によって保存された遺伝子が15世代変化しなければ GA を終了する

5. シミュレーション

線形変換回路の各パラメータを以下のように設定し，シミュレーションを行った．

入力数：16　　出力数：16
係数の bit 幅：4bit　加算器数上限：なし

GA の各パラメータを以下のように設定する．

個体数：20　世代数：100
交叉率：70%　突然変異率：15%　エリート保存率：10%

表1に提案法で分割数を変化させた場合と，分割せずに合成した場合の結果を示す．ただし，計算時間は適応度計算1回にかかる時間の平均とする．

また，合成方法 (b) で中間係数列 S_{i1}, S_{i2} の段数に制限をかけた場合の結果を表2に示す．

表 1: シミュレーション結果 1

	分割	加算段数	加算器数	計算時間 [s]
分割なし	-	11	220	1475.89
提案法	3	10	245	150.36
	4	10	252	92.31
	5	11	266	56.97

表 2: シミュレーション結果 2(分割数 4)

	中間係数列段数制限	加算段数	加算器数
提案法	3	7	264
	4	8	260
	5	9	253
	6	10	247

6. むすび

本研究では，線形変換回路合成における問題点を指摘し，それらを解決するための手法を提案した．係数行列を分割してそれぞれを最適合成することにより線形変換回路の合成を行った．

その結果，提案法では，分割せずに合成した場合に比べ加算器数は多少増加したが，計算時間の大幅な削減が可能となった．また，中間係数列合成の際，段数に制限をかけることで，加算段数のさらなる削減が可能となった．そのため，線形変換回路合成の分割最適化は有効であると言える．

参考文献

[1] 佐藤圭介，佐々木孝雄，豊嶋久道，“線形変換回路の係数列合成順序を考慮した演算コスト削減アルゴリズム”，電子情報通信学会，技術研究報告，CAS2004-46, pp.25-28, Nov.2004.

[2] M. Potkonjak, et al.,“Multiple constant Multiplications: Efficient and Versatile Framework and Algorithms for Exploring Common Subexpression Elimination”, IEEE Trans. on CADCAS, vol.15, No.2, pp.151-165, Feb. 1996.

[3] 坂和 正敏，田中 雅博，“遺伝的アルゴリズム” 朝倉書店，1995.