Vol.2009-AVM-65 No.12
2009/7/9

招待論文

進化的画像符号化

高　村　誠　之[†1]

遺伝的プログラミング (GP) に基づく進化的手法は，計算機によるアルゴリズム自動生成を可能とし，プラント制御，ロボット制御，株価予想など幅広く応用されている．しかしながら，JPEG や H.264/AVC などの従来の映像/画像符号化方法は，すべて固定された (動的でない) アルゴリズムを用いている．筆者が検討を進めている「進化的符号化」は，GP に基づき入力画像に特化した画像予測アルゴリズムを自動生成するもので，従来の「固定アルゴリズム」「人が考案」「人がコーディング」からの脱却を図る，新たなパラダイムに向けた試みである．本論文では，現在も日々継続的に更新されつつある画像予測アルゴリズムの最新性能を報告すると同時に，予測器の複雑さと予測性能の関係についての考察，並列化により進化処理が 2 桁高速化した事例を紹介し，究極の圧縮効率にも言及する．

Evolutive Image Coding

Seishi TAKAMURA[†1]

Evolutive methods based on genetic programming (GP) enable dynamic algorithm generation, and have been successfully applied to many areas such as plant control, robot control, and stock market prediction. However, conventional image/video coding methods such as JPEG and H.264/AVC all use *fixed* (non-dynamic) algorithms without exception. The evolutive coding enables an automatic generation of pixel prediction algorithm. It is a radical departure from conventional “fixed algorithm”, “man-made algorithm” and “hand-made programming” toward a new paradigm. In this article, we introduce a GP-based image predictor that is specifically evolved for each input image, which have been evolving day by day. We demonstrate its up-to-date performance as well as the investigation on predictor complexity versus prediction performance. We also report about speeding up the evolution process using parallelization by a factor of 100. Ultimate coding efficiency will also be touched upon.

1. ま え が き

従来，符号化において動きベクトルや量子化パラメータ等の「符号化パラメータ」は動的に変更され得るものの,「符号化手順」自体はあくまで固定であった．即ち，新たな符号化手順は人間が試行錯誤の末考案するほかなく，故に符号化器の構成は人間の把握しうる程度の複雑さを超えることができなかった．また，入力画像に特化した符号化手順を (画像カテゴリでなく)画像毎に新規に生成するようなことも非現実的であった．

そこで映像符号化の手順自体を計算機により映像毎に動的最適構成させることを究極の目的と考え，進化的手法を用いて画像毎に自動構成する方法を検討している[1]．

本報告では，画像可逆符号化の「予測器」の自動生成を述べ，その現時点の性能に加え，進化並列化に関する報告，また予測器情報量と残差情報量の関係および限界圧縮率に関する考察を述べる．

2. 進化的手法とその画像・符号化応用例

遺伝的アルゴリズム (genetic algorithm, GA) は最適化問題の「解のパラメータ」を個体の遺伝子として 1 次元配列化し，個体集団に対して適者生存に基づく世代交代を繰り返すことによって実用解を得る探索手法であり，遺伝的プログラミング (genetic programming, GP) は遺伝子を木構造にすることで「解の手順」の最適化を行えるよう拡張した手法である．生物進化に示唆を得たこれら GA, GP は「進化的手法」と呼ばれ，プラント制御や株価予測，ロボット制御，アナログ回路設計等極めて広範に応用されている．

符号化については，2 値画像を符号化する際のコンテクストテンプレートや，符号化する単位領域の分割形状を GA により最適化する方法が提案されている[2),3)]．これらはパラメータを最適化するもので，符号化手順そのものは固定である．画像処理については，医用画像から神経細胞を抽出する手順を GA, GP で動的に生成する方法が提案されている[4)]．

2.1 進化的手法の画像処理用途と符号化用途の相違点

画像処理において，入力画像は無論固定できないため様々な入力画像に対し汎用的に動作することが期待されるが，いかなる未知入力画像に対しても良好な動作を保証することは現実的に不可能である．画像符号化においては，入力画像だけを効率的に符号化することでも

†1 日本電信電話株式会社 NTT サイバースペース研究所
NTT Cyber Space Labotratories, NTT Corporation, Y-517A, 1-1 Hikarino-oka, Yokosuka-shi, 239–0847 Japan

Vol.2009-AVM-65 No.12
2009/7/9

表 1 進化的画像処理と進化的画像符号化の相違点
Table 1 Differences between evolutive image processing and evolutive image coding

	進化的画像処理	進化的画像符号化
汎用性の要求	あり	なし
評価規準の恣意性	あり	なし
教師データ	要	不要
ツリーの bloat 抑制	要	不要

I_{10}	I_{07}	I_{05}	I_{08}	I_{11}
I_{06}	Inw	In	Ine	I_{09}
I_{04}	Iw	p		

図 1 p を予測する周辺画素
Fig. 1 Neighboring pixels of p.

意味があるため，未知入力対応の必要がない．この「汎化不要性」が進化的画像処理と進化的画像符号化の最大の違いである．

加えて，画像処理においては人間が事前に「教師画像」を作成し，かつそれを用いた学習が必要である．符号化においては，木と残差の情報量和が統一評価尺度であるため，教師情報が不要である．また画像処理応用では，教師情報に対する適合度の定義に設計者の恣意性が入りこむ余地があったが，符号化応用においては符号量が唯一の尺度であり，恣意性の入り込む余地がない．また符号化応用では後述のように符号化器の情報量も個体の評価尺度に含むため，一般の GP において問題となる木の肥大化 (bloat)[5] が自動的に抑制される．

これらをまとめると**表 1** のようになる．

3. 提 案 手 法

3.1 予測器の木表現

例えば**図 1** において着目画素 p を符号化する際，周辺の復号済み画素 (Inw, In 等) が一般に x と高い相関を持つことを利用して，これらを用いて x の予測値 ($\hat{x}$ とする) を生成するのが予測器である．しかるのち予測誤差 $x - \hat{x}$ がエントロピ符号化される．

例えば JPEG-LS[6] の MED 予測手順の C 言語および木による表現は**図 2** のようになる．ここで T は，第一引数が非負なら第二引数を，負なら第三引数を返す三項演算子である．このようにして任意の初等手順 (e.g., 画素値予測手順) を，木として記述することができる．

3.2 GP による予測器の自動構築と進化戦略

まず母集団を，乱数で生成した木や前述の MED 等各種予測器の木等から形成しておく．そして一般に知られた GP の手順 (複製選択，子の生成 (交叉・突然変異)，生存選択) により予測器を進化させる．複製選択および生存選択を「世代交代モデル」と呼ぶが，本論文では以下のような minimal generation gap モデル[7] を用いた．

表 2 画素予測器進化計算に用いたノード

	ノード	意味
非終端ノード	T	条件分岐
	max, min	最大, 最小
	ave, abs, sqr, sqrt	平均, 絶対値, 二乗, 平方根 ($\sqrt{\lvert\cdot\rvert}$)
	sin, cos, tan	三角関数
	arcsin, arccos, arctan	逆三角関数
	sinh, cosh, tanh	双曲線関数
	log, $\log_{10}$, pow, pow2	自然・常用対数 ($\log\lvert x\rvert$)・べき乗 ($\mathrm{pow}(a,b) = sgn(a)\lvert a\rvert^b$, $\mathrm{pow2}(a,b) = sgn(a)\lvert a\rvert^{b/10}$)
	add, sub, mul, div	四則演算
	xor, or, and	疑似論理演算
終端ノード	$In, Ine, Iw, Inw, I_{04}...I_{11}$	周辺画素値
	D	GAP 予測器が補助的に用いている変数．$D = (\lvert Iw - Inw\rvert + \lvert In - Inn\rvert + \lvert Ine - Inne\rvert) - (\lvert Iw - Iww\rvert + \lvert In - Inw\rvert + \lvert In - Ine\rvert)$
	I	同じく $I = (Iw + In)/2 + (Ine - Inw)/4$
	Igap, Imed, Ils, Ile12	GAP 予測値, MED 予測値, 最小二乗予測値 (4 画素使用), 最小エントロピ予測値 (12 画素使用)
	x, y	正規化座標．画像中心が原点，左上端が $(x, y) = (-1, -1)$, 右下端が $(1, 1)$.
	ρ, θ	正規化極座標 (L1 ノルム)．画像中心が原点，画面端が $\rho = 1$, 右上端が $\theta = \pi/4$ 等.
	定数	32 ビット浮動小数点数 (float)

実際の進化戦略は以下のような手順で行う．

(1) まず母集団を，乱数で生成したものや既存の予測アルゴリズム (MED 予測や平均予測等) などから生成しておく．
(2) 母集団の中から親集合を選択する (複製選択).
(3) 親集合から子個体集合を生成し (子の生成)，評価する (評価尺度については後述).
(4) 評価の結果に基づき子個体集合から生存させるものを選択する (生存選択).

「子の生成」は，親として選ばれた個体同士での交叉，突然変異，逆位等により行う．この様子を**図 3** に示す.

3.3 予測手順の評価

生存選択のための評価尺度として，提案手法では木構造を表現するための情報量 H_T とその木構造の手順により画素値を予測した残差の情報量 H_R の和 ($H_T + H_R$, 復号器に伝送すべき情報量に相当) を用いる．H_T は木に含まれる全てのノード (数値または関数) の情報量の和であり，H_R は予測残差を (CALIC[8] と同様に周辺エッジ強度に基づき 8 通りに) コンテクスト分離し総情報量を低減したものである．提案・比較手法いずれにも本低減処理は施される．ただし CALIC の error feedback/flipping は用いていない．

```
if (Inw >= max(Iw, In))
    return min(Iw, In);
else if (Inw <= min(Iw, In))
    return max(Iw, In);
else
    return Iw + In - Inw;
```

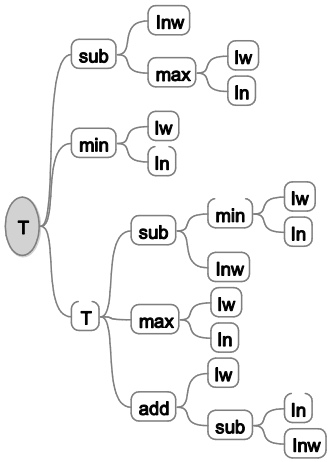

図 2 C 言語および木による MED 予測器の表現
Fig. 2 C- and Tree- expressions of MED predictor

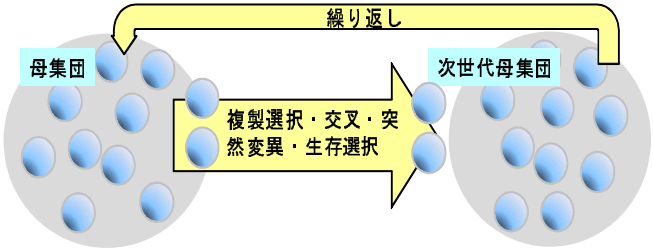

図 3 進化戦略 (単一プロセス)
Fig. 3 Evolution strategy (for single process)

今回，**表 2** に示すように，関数ノードは min, max, 絶対値, 三角関数, 四則演算, 二乗, 平方根, p の正規化座標値 ($x, y \in [-1, 1]$), 図 1 に示す周辺 4 画素値，CALIC の gradient-adjusted prediction (GAP) 値，MED 予測値など $N = 49$ 種，数値ノードは 32 bit 浮動小数点数 (C の float 型) とした．また全ノード中数値ノードの生起率を $P_{num} = 0.15$ と仮定した．n_{d_C} をコンテクスト C(8 通り) における予測誤差値 $d_C(= -255 \ldots 255)$ の生起回数，N_C をコンテクスト C に属する画素数とした．

表 3 上段: 総情報量 ($H_T + H_R$ [bpp]) および提案方式の木情報量 (H_T [bits]) 比較．下段: 提案方式 (12 画素参照) による画像特化予測器の，他画像予測性能比較．
Table 3 Top table: Residual entropy (overhead inclusive) of the predictors ($H_T + H_R$ [bpp]), their increment ("incr." in percentage) against EP12, as well as their H_T [bits]. Numbers next to the predictors mean the number of reference pixels. Lower table: proposed predictor (with 12 pixels)'s generic prediction performance ($H_T + H_R$ [bpp]).

予測器	Lena	Baboon	Airplane	Peppers	平均	増分 [%]
LS(4)	4.551	5.521	3.654	4.465	4.548	3.705
LS(12)	4.549	5.374	3.635	4.41	4.492	2.428
LE(4)	4.529	5.506	3.619	4.417	4.517	3.013
LE(12)	4.522	5.361	3.595	4.382	4.465	1.813
GAP	4.539	5.556	3.568	4.468	4.533	3.367
MED	4.692	5.592	3.644	4.646	4.643	5.886
Proposed(4)	4.481	5.462	3.521	4.352	4.454	1.574
H_T [bits]	1191.2	813.1	971.7	1207.9	1046.0	
Proposed(12)	**4.425**	**5.334**	**3.486**	**4.296**	**4.385**	—
H_T [bits]	1520.4	886.6	1237.2	2199.1	1460.8	

学習画像					平均
Lena	(4.425)	5.691	3.603	4.386	4.526
Baboon	4.745	(5.334)	3.698	4.773	4.638
Airplane	4.561	5.510	(3.486)	4.519	4.519
Peppers	**4.505**	5.621	3.628	(4.296)	4.513

H_T, H_R は以下のように教師情報を用いず画像内に閉じた計算により与えられる:

$$H_T = \sum_{n\in\text{木}} \begin{cases} -\log_2 P_{num} + 32 & (n \text{ が数値}) \\ -\log_2(1 - P_{num}) + \log_2 N & (n \text{ が関数}) \end{cases} \quad [\text{bits}] \tag{1}$$

$$H_R = -\sum_{C} \sum_{n_{d_C}\neq 0} n_{d_C} \log_2 \frac{n_{d_C}}{N_C} \quad [\text{bits}] \tag{2}$$

3.4 並列進化による高速化

本進化計算をシングルプロセスで行う場合，高速化手段としてツリー評価の能率化や高速プロセッサの利用などが考えられるが，せいぜい 2-3 倍のオーダーに留まる．そこで**図 4** に示すように，並列動作向けにプログラムフローを変更した．共有ファイルに全体最良個体 (Global Best Individual, GBI) を記録し，母集団が一世代経る毎にそのタイムスタンプを確認し，必要であれば自己の最良個体 (Local Best Individual, LBI) へ取り込みあるいは LBI を GBI へ上書きする．こうして複数プロセス間で GBI が共有される．しかる後，**図 5** に示

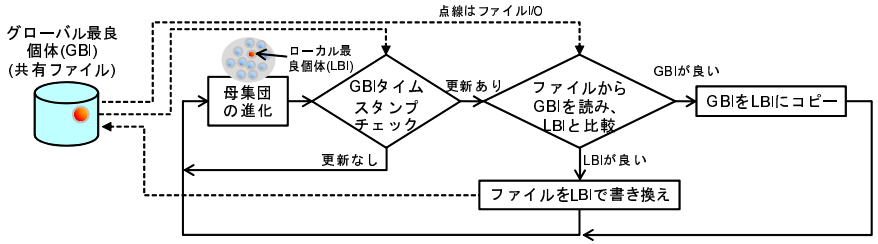

図 4 並列プロセス向け変更
Fig. 4 Modification for parallel processing

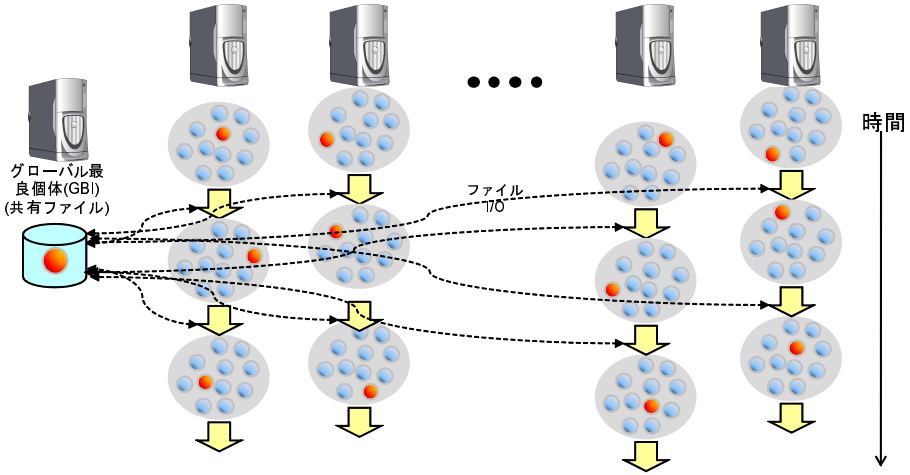

図 5 非同期並列進化の実行イメージ
Fig. 5 An image of asynchronous parallel evolution

すような複数プロセスによる進化計算を行う．

ファイル共有によるプロセス間通信のため，ソースコードも実行環境も，OS に依存しない．またプロセス間は非同期に実行できるため，プロセスの途中離脱・途中追加が随時可能であり，かつスケーラビリティが 1〜∞となる．各プロセスは一回の進化 (約 10 秒) の後，一回共有ファイルのタイムスタンプを取得する．うち数十回に一回 (記録更新があった場合)，共有ファイルの中身 (数百バイト) を読み込む．このようにファイル I/O 負荷は無視できる程度である．またプロセス間の同期をとらないので各プロセッサの load は常に 100%となる．

4. 実験結果・考察

用いた画像は Lena, Baboon, Airplane, Peppers の 4 画像 (いずれも 512 × 512 画素，8 bpp, 輝度のみ) である。比較対象は線形予測 2 種 (最小二乗予測 (LS) および最小エントロピ予測 (LE)，GAP(以上，提案手法も含め図 1 の近傍 4 画素あるいは 12 画素を利用) および MED 予測 (同 3 画素を利用) とした．LE 予測は LS 予測の (定数項込みのため 4 + 1 =)5 係数 (あるいは 12+1=13 係数) を初期値とし Powell 法による多次元探索で H_R を最小化したもので，線形予測では最高効率と言える．

4.1 効 率 評 価

画像 Lena に対し進化計算の結果得られた予測器を，整形・最適化し通常の中値記法により表現したものを**図 6** に示す．同様に，画像 Peppers に対し生成された予測器を**図 7** に示す．極めて複雑な予測アルゴリズムであることがわかる。

表 3 上段に，使用した 4 画像毎のオーバーヘッド (LS(4), LE(4) は (32 × 5 =)160, LS(12), LE(12) は 416, GAP, MED は 0, 提案手法は H_T bits) 込みの残差情報量と，それらの提案手法からの増分，また提案手法の H_T を示す．括弧内の数字は使用した周辺画素数である．全画像について提案手法が最も効率が高いことが確認できる．また、より多くの周辺画素を参照することで性能が高まることも確認できる。木情報量 H_T は平均 1460.8 bits で，GAP, MED 予測器 (それぞれ H_T を見積もると 349.5, 116.0 bits. 既知方式のため本実験ではともに 0 とおいた) よりも相当複雑となっている．

4.2 汎用性評価

表 3 下段は，左列の画像に特化し自動生成した予測器を用い，それ以外の 3 画像を予測した結果である．太字で示された，Peppers 用予測器で Lena を予測した場合のみは，比較手法いずれにも勝っていたが，他のケースでは比較手法いずれかに劣る結果となった．特に Baboon 用予測器は平均ビットレートが 4.638 bpp と高く，汎用性が低い．提案方式は汎用性を考慮しないため，この結果は妥当と思われるが，Peppers → Lena 予測器のように，共通に使える符号化器が生成される可能性もうかがえる．

4.3 並列化による進化速度改善

単一プロセスによる進化は Core2 Extreme QX9775 (3.2GHz, 4GB Memory, MS Windows Vista Ultimate (32bit) SP1) にて行った．並列進化には，Core2 Quad Q6600 (2.4GHz, 2GB Memory, MS Windows XP Professional x64 Ed. Ver. 2003 SP2) 30 台からなるクラスタ (計 120 コア) を用いた．

$$\max(\theta+and(Ile_{12},\min(\cosh(0.7071\sqrt{|\log_{10}(I_{06})+Igap|}),|2\log\sinh\sinh^4\arcsin\sinh\tanh(\frac{2{I_{08}}^2}{\max^2(\sqrt{Ile_{12}},Igap-and(\min(Inw,\frac{\min(In,0.5\,(Ine+Imed))}{\cosh\arcsin\arcsin\cos(\frac{I_{04}}{\max(I_{04},I_{10})})},I_{09}),I_{05}))\ \sinh^2\sinh\log_{10}(\cosh\log_{10}(Ine))\,(I_{09}+Ine)})-Ile_{12}|)+\sinh\sinh\arctan\sinh(\max(0.5\,(Inw+\max(Imed,In)),I_{09})-Ile_{12})),or(\min(\frac{|\sinh\tan y+\max(Imed,I_{05})|}{|y|},\frac{Iw}{\arctan\arctan|\log_{10}(Iw)|^{\frac{D}{20}}},\log(\frac{Ine^8}{\sin^8\sin\arctan^2\arctan\tanh^4\log_{10}(Ile_{12})\max^4(Ine,I_{05}\frac{\sqrt{|I_{05}|}}{10})\cos^8\sin^2(\frac{x}{\sqrt{1-x^2}})\,(Inw-I_{09})^8})-Ils+2Ile_{12},0.5\,(0.5\,(I_{09}+Inw)+\max(and(or(Imed,I_{05}),Inw-xor(D,\cos\tan(\frac{2In}{Inw+Imed}))),0.5\,(Iw+In)+0.25\,(Ine-Inw),\log_{10}(I_{08})+\sqrt{|\arcsin\tan\sinh x|}+0.5\,(Inw+Ils)))+|\arccos(\frac{\max(Igap,Iw)}{I_{04}})|^{\frac{1}{4}}+\theta),\min(\max(-\max(Inw,I_{06},I_{07})+and(Ile_{12},\log\sinh(0.5\,(-xor(D,\cos\cosh(\frac{In^{\frac{1}{4}}}{|or(I_{05},I_{09})|^{\frac{1}{4}}}))+Inw+Igap)+\sinh\sinh(0.5\,(\arcsin\sinh\tan y+\theta))-\sinh\arccos\arcsin x))+\max(Ile_{12},\min(\max(Imed,In),\sinh\sqrt{|D|})),\tanh\tanh\sin(0.5\,(Iw+In)+0.25\,(Ine-Inw)-Ile_{12})+Igap),\max(0.5\,(Iw+In)+0.25\,(Ine-Inw),-12.8182(I_{10}-0.5\,(I_{05}+Ine))))))$$

図 6 画像 Lena に対し生成された予測器 (情報量 1520.4 bits)
Fig. 6 Generated predictor for Lena (tree size=1520.4 bits)

$$\textbf{if}\ D\geq 90.7266\ \textbf{then}\ Igap\ \textbf{else if}\ D\leq 30.9851\ \textbf{then}\ 0.50107(|xor(\min(\max(Ine,Iw),|I_{05}-\max(\min(Ine,Inw),|and(I_{10},Ine)-I_{06}+\theta+Ils+Ile_{12}|-I_{08}-I_{07}-\rho+\max(Inw,\textbf{if}\ x\geq 0\ \textbf{then}\ y\ \textbf{else}\ \min(Imed,I_{11}-I_{10})))-In|),Igap)|+Igap)\ \textbf{else if}\ -\sqrt{|I_{10}-I_{04}|}-\sqrt{|I_{10}-Iw|}-\sqrt{|I_{10}-Ile_{12}|}+\theta-\max(Inw,\sqrt{|I_{07}-and(Ile_{12},I_{09})|}+\max(Ine,\max(\frac{|or(I_{07},Inw)-I_{04}-\sinh\rho-Inw-Imed|}{2},or(I_{10},I_{07}))))+\min(or(\min(\min(Ils,I_{08}),I_{06},I_{09}),\|D|-\frac{Ils-30.9052}{2}-89.862|),I_{05})+30.0724\geq 0\ \textbf{then}\ 0.250489\,(3\,(|0.28107\,(|0.608201\,(|\sin\tan(\frac{Ine}{\sqrt{1-Ine^2}})\,I_{09}+|\sin\cos\theta|\,\|I_{04}-Iw-Ils|-Inw+\max(xor(xor(\sqrt{Ine},Ine),I_{11}),-I_{05}+Iw+Ils)\|+0.316839\,(|I_{07}-I_{04}+Ine|-Inw+Ine)+xor^2(3.09001,|\tan\cos y|))+\sin(\frac{\tan\sin\tan(\frac{Ile_{12}}{\sqrt{1-{Ile_{12}}^2}})}{\sqrt{\tan^2\sin\tan(\frac{Ile_{12}}{\sqrt{1-{Ile_{12}}^2}})+1}})\,I_{04}|+0.32813\,I_{06}+0.32031\,Ine)-0.61719\,Iw-0.33812\,Ine|+0.28906\,Ine)+In)\ \textbf{else}\ \frac{\frac{|I_{07}-\arctan\theta-\frac{Iw+In}{2}-Iw-Inw-\frac{Ine-Inw}{4}|}{2}+\min(\max(\min(\min(Ine,Inw),I_{04},I_{09}),|\theta+\min(\sinh Iw,-I_{06}+\frac{Iw+In}{2}+and(Inw,Ine)+\frac{Ine-Inw}{4})+Ile_{12}|-I_{08}),\max(Iw,I_{09}))}{2}+Igap}{2}$$

図 7 画像 Peppers に対し生成された予測器 (情報量 2199.1 bits)
Fig. 7 Generated predictor for Peppers (tree size=2199.1 bits)

画像 Lena に対する並列進化の結果は**図 8** に示すとおりで，一週間でほぼ予測性能が収束した．単一プロセスで 5 か月程度かかって到達した予測器性能に相当する進化は，並列プロセスの場合約 2 日で得られた．Peppers については、同じく 5 か月程度の進化が 7 時間で得られた．コア単体の性能はクラスタの方が低いにもかかわらず，75 倍〜500 倍の高速化が実現され，並列化が効率的に行われたことが確認できる．

4.4 予測器の複雑さと性能、その極限値

同じく図 8 に，時間と木情報量の関係も示す．進化が進むにつれ次第に木が複雑になっていくこと，またこの図では $7.5H_T$ がほぼ総情報量 (H_R+H_T) と上下対称関係にあることもわかる．この結果は「木が 1bit 複雑になると，予測残差は 8.5bit 減る (総情報量は 7.5bit 減る)」ということを表している．

さらにこの関係を詳しく見るため，**図 9** に，画像 Lena についての木の情報量 (H_T) と総情報量 (H_R+H_T) の関係を示す．二回の試行でそれぞれ傾向が異なるが，いずれにおいても

- 初期探索段階では急速に性能が向上する
- その後はほぼ一定の割合で性能が向上していく

という傾向がある．グラフに重畳して示した直線は，試行 1, 2 それぞれ $y=1.163\cdot 10^6-7.5x$ および $y=1.166\cdot 10^6-13x$ (但し $x=H_T, y=H_R+H_T$) である．両直線の傾きが異なることから，ローカルミニマに探索が陥っている可能性も考えられるが，上述の傾向がそのまま継続すると仮定すると，$H_R\geq 0, H_T\geq 0$ であることを考え，$H_R\to 0$ とした極限圧縮率は 0.5281 [bpp](試行 1) および 0.3215 [bpp](試行 2) となる．これらの値が，時間を無限にかけた場合の「究極の可逆符号化効率」を示唆していると考えられる．

5. む す び

画素値を予測する手順自体を，その情報量を見積りつつ計算機に自動進化させることで，手順の複雑さと符号量のバランスを最適にする方法を提案し，実験の結果 GAP を若干超え

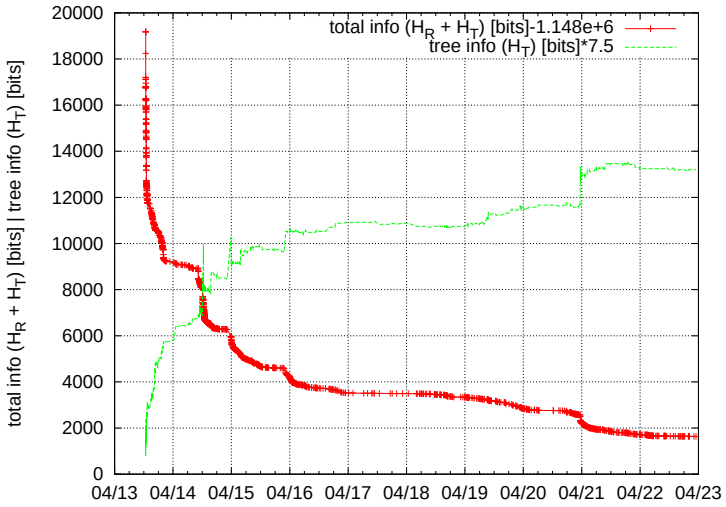

図 8 総情報量 ($H_R + H_T - 1.148 \cdot 10^6$) と木情報量 ($H_T * 7.5$) の遷移 (並列化後)．横軸は日付. 画像: Lena, 第一試行.
Fig. 8 Transitions of total information ($H_R + H_T - 1.148 \cdot 10^6$) and tree information ($H_T * 7.5$), after parallelization. Horizontal axis means the date. Image: Lena, trial: 1.

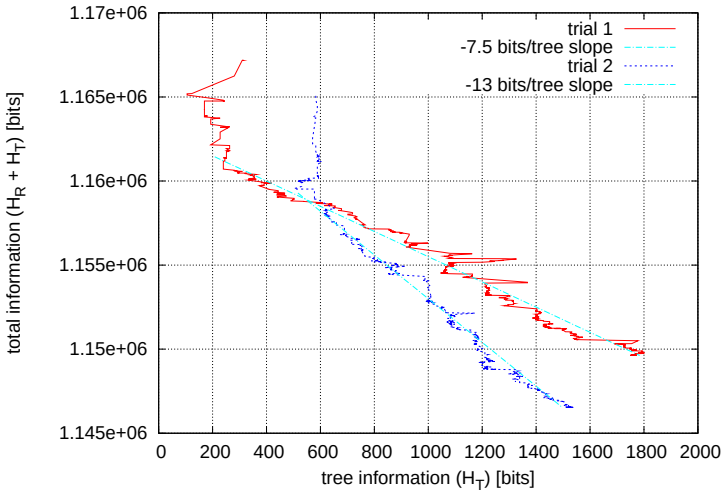

図 9 木情報量 (H_T) と総情報量 ($H_R + H_T$) の関係およびそのトレンド. 画像: Lena
Fig. 9 Tree information (H_T) vs. total information ($H_R + H_T$) and their trends.

る複雑さで，効率的な画素予測 (LE 予測比 3.0 %, GAP 比 3.4 %のエントロピ削減) が行えることを報告した．また進化的手法により生成された画素予測器の基本性能，汎用性をそれぞれ評価・考察した．

今後の検討課題として，GPU を利用したさらなる高速化[9]，Genetic Network Programming[10] の応用，非可逆符号化や映像符号化への発展等が挙げられる．

参 考 文 献

1) 高村誠之, 松村誠明, 八島由幸: “遺伝的プログラミングに基づく画素予測器の生成と評価”, 電子情報通信学会技術報告, vol. 108, no. 425, IE2008-210, pp. 37–40, Feb. (2009)
2) 田中 雅晴, 坂無 英徳, 溝口 正信, 樋口 哲也: “遺伝的アルゴリズムを用いたディジタル印刷画像の 2 値画像符号化”, 電子情報通信学会論文誌 D-II, vol. J83-D-II, no. 5, pp. 1274–1283, May (2000).
3) 高木 幸一, 小池 淳, 松本 修一, 山本 英雄: “遺伝的アルゴリズムを用いた動画像動き補償・領域分割符号化方式”, 電子情報通信学会論文誌 D-II vol. J83-D-II, no. 6, pp. 1437–1445, June (2000).
4) 藤嶋 航, 長尾 智晴: “GP による構造最適化と GA による数値最適化を併用した画像処理自動生成法 PT-ACTIT”, 映像情報メディア学会論文誌, vol. 59, no. 11, pp. 1687–1693, Nov. (2005).
5) 沢田 賢治, 狩野 均: “多目的手法を用いた構造化進化戦略による最適化問題の解法”, 計測自動制御学会 第 30 回知能システムシンポジウム, Mar. (2003).
6) ISO/IEC 14495-1: “Lossless and near-lossless compression of continuous tone still images” (2000).
7) 佐藤 浩, 小野 功, 小林 重信: “遺伝的アルゴリズムにおける世代交代モデルの提案と評価”, 人工知能学会誌, vol. 12, no. 5, pp. 734–744, Sep. (1997).
8) Wu, X. and Memon, N.: “Context-based, adaptive, lossless image coding”, IEEE Trans. Commun., vol. 45, no. 4, pp. 437–444, Apr. (1997).
9) 安藤 淳, 長尾 智晴: “複数の GPU を用いた超高速進化的画像処理システム”, 情報処理学会研究報告. MPS, 数理モデル化と問題解決研究報告, vol. 85, pp. 71–74, Sep. (2008).
10) 平澤 宏太郎, 大久保 雅文, 片桐 広伸, 胡 敬炉, 村田 純一: “蟻の行動進化における Genetic Network Programming と Genetic Programming の性能比較”, 電気学会論文誌 C，vol. 121, no. 6, pp. 1001–1009 (2001).