Vol.2010-AVM-70 No.9
2010/9/2

進化的画像符号化の展望
～スパコン・GPU による高速化の試み～

高村誠之†

筆者が検討を進めている「進化的画像符号化」は、遺伝的プログラミング(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 JPEG や MPEG,H.264/AVC にみられる「汎用アルゴリズム」「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といえる。しかしながら計算量が膨大にかかるという問題があったため、本論文では、進化処理を並列化し、スーパーコンピュータ上で 1,500 並列以上のシミュレーションによる 2-3 桁高速化や、GPU による 2 桁程度の高速化、並列度と進化速度の定量的関係などの事例を紹介する。

Perspective of Evolutive Image Coding
~Acceleration with Supercomputer/GPU~

Seishi Takamura†

Evolutive methods based on genetic programming (GP) enable dynamic algorithm generation such as content-specific pixel predictor. It is a radical departure from conventional “fixed algorithm”, “man-made algorithm” and "hand-made program" towards new paradigm such as “content-specific algorithm”, “machine-generated algorithm/program”. However, the evolution processing for image coding requires highly intensive computation. In this paper, we report about speeding up the evolution process by a factor of 100-1,000 with up to 1,500 parallelization, ~10 speed up with GPU and the relation between parallelization and evolution speed.

† 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 Laboratories, NTT Corporation

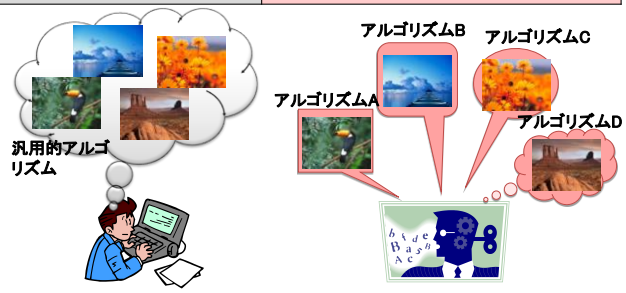
アウトライン

- 進化的画像符号化のコンセプト
- 生物進化と進化計算
- アルゴリズムの進化
- 実験結果
- 進化の高速化・高能率化
- 究極の圧縮率は?
- まとめ・今後の課題

NTT AVM研究会 (2010.9.2-3) 進化的画像符号化の展望 NTT 高村誠之 2

進化的画像符号化のコンセプト

	従来の符号化方式 (JPEG, H.264, etc)	理想の符号化方式
考案・実装者	人	計算機
アルゴリズムの特性	既製服型(固定的・汎用志向) 人が作れる規模 人が考えうるアルゴリズム	オーダーメード型(対象に柔軟に特化) 人智を超える規模 人が思いつかないアルゴリズム



NTT AVM研究会 (2010.9.2-3) 進化的画像符号化の展望 NTT 高村誠之 3

ⓒ 2010 Information Processing Society of Japan

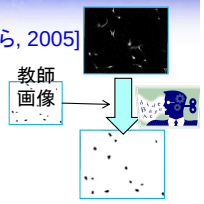
進化計算

- 生物進化にヒントを得て、最適化問題の解を遺伝子になぞらえ、この交叉・突然変異により世代交代を経て求解
- 遺伝的アルゴリズム (Genetic Algorithm, GA)
  - 個体を一次元配列として表現
  - パラメータの値を最適化探索
  - 応用先: 多次元空間の目的関数最大(最小)点の探索
- 遺伝的プログラミング (Genetic Programming, GP)
  - 個体をツリー(木)として表現
  - 処理手順を最適化探索
  - 応用先: プラント制御、株価予測、ロボット制御など

符号化アルゴリズム自動構築にはGPが向く

画像/符号化関連研究

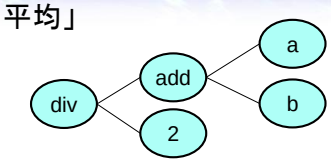
- 画像処理応用:「進化的画像処理™」 [長尾ら, 2005]
  - GAとGPを併用し、神経細胞の本体を抽出
  - ☹教師画像は人間が作る
  - ☹汎用性の検証もしているものの、保証は不可能



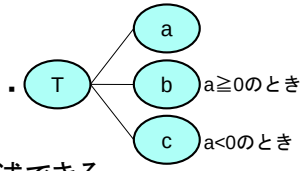
- 符号化応用
  - 2値画像を符号化する際のコンテクストを生成する「テンプレート」をGAにより最適化[田中ら, 2000]
  - 符号化単位領域の分割形状を、GAにより最適化する[高木ら, 2000]
  - ☹いずれもパラメータの最適化であり、符号化手順そのものは固定
  - 符号化手順を進化させた例は筆者が知る限りない

GPの準備: アルゴリズムのツリー表現

- 「aとbの平均」



- 条件分岐も表現可能・・・・・・・・・



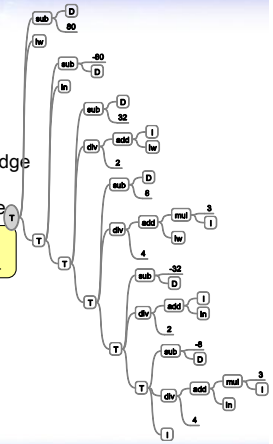
- 任意の初等アルゴリズムが記述できる

GAP (Gradient-Adjusted Prediction)

- CALICが用いる予測器[Wuら, 1997]
- C表現

```
if (D >  80) return Iw;          //Sharp Horizontal Edge
if (D < -80) return In;          //Sharp Vertical Edge
if (D >  32) return (I + Iw)/2; //Horizontal Edge
if (D >   8) return (3*I + Iw + 2)/4; //Weak Horizontal Edge
if (D <-32) return (I + In)/2; //Vertical Edge
if (D <  -8) return (3*I + In + 2)/4; //Weak Vertical Edge
return I;
```

Dは周囲エッジ強度指標,
I = (Iw + In)/2 + (Ine - Inw + 2)/4

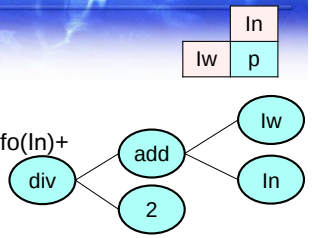


- 情報量(後述) 349.5 bits

ⓒ 2010 Information Processing Society of Japan

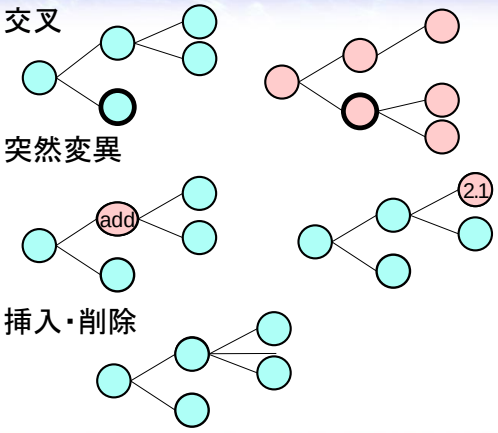
予測器の評価尺度

- 予測器(木)の情報量 I_T [bits]
  - 全ノードの情報量の和
  - I_T = info(div)+info(add)+info(Iw)+info(In)+ {info(定数)+info(2)}
  - この情報量で木を復号できる
- 予測残差情報量 I_R [bits]
  - 上記予測器を用いた残差信号の情報量
- 評価尺度 = $I_T + I_R$
  - デコーダが原画像を復号できるための情報量下限

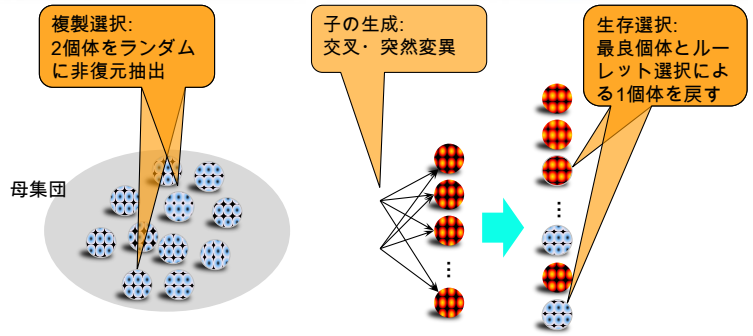


木の交叉・突然変異

- 交叉
- 突然変異
- 挿入・削除



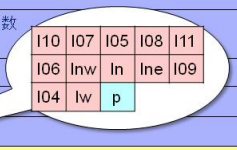
進化計算手順



複製選択・生存選択はMinimal Generation Gap法[佐藤ら, 1997]

用いたノード（全49関数+定数）

	関数	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(√|x|)
	sin, cos, tan, arcsin, sinh, etc.	三角関数, 逆三角関数, 双曲線関数
	log, pow, etc.	自然対数, 常用対数, べき乗
	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, I04..I12	周辺画素値
	D, I	GAP予測器が補助的に用いている変数。
	Igap, Imed, Ils, etc.	GAP予測値, MED予測値, 最小二乗予測値
	x, y	正規化座標。画像中心が原点、左(上)端がx=(y=)−1, 右(下)端がx=(y=)1。
	ρ, θ	正規化極座標
	定数	32ビット浮動小数点数(float)



ⓒ 2010 Information Processing Society of Japan

実験条件

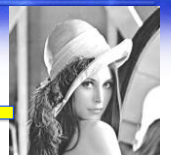
- 512x512, grayの4画像 (Lena, Baboon, Airplane, Peppers)

- 比較予測器
  - 非線形予測器: GAP, MED予測器
  - 線形予測器: 周囲4or12画素から予測
    - LS(Least Square)予測器: 最小二乗予測(4or12係数)
    - LE(Least Entropy)予測器: 最小エントロピとなるよう4or12係数を最適化

線形予測器中
効率最高

画像Lenaの予測器

- 右の画像に特化し、全自動で進化させたもの
- 「人智を超える規模」で「人が考えつかない」ような予測式が得られている



I_T = 1520.4 bits

$$\max(\theta + and(Ile_{12}, \min(\cosh(\sqrt{0.5|\log_{10}(I_{06}) + Igap|}), |2\log\sinh\sinh^4\arcsin\sinh\tanh(\frac{2I_{08}{}^2}{\max^2(\sqrt{|Ile_{12}.Igap - and(\min(Inw, \frac{\min(In, 0.5(Ine+Imed))}{\cosh\arcsin\arcsin\cos(\frac{I_{04}}{\max(I_{04}.I_{10})})}.I_{09}), I_{05}))}\sinh^2\sinh\log_{10}(\cosh\log_{10}(Ine))(I_{09}+Ine)}) - Ile_{12}|) +$$

$$\sinh\sinh\arctan\sinh(\max(0.5(Inw + \max(Imed, In)), I_{09}) - Ile_{12})), or(\min(\frac{|\sinh\tan y + \max(Imed, I_{05})|}{|y|},$$

$$\frac{Iw}{\arctan\arctan|\log_{10}(Iw)|^{\frac{D}{20}}}, \log(\frac{Ine^8}{\sin^8\sin\arctan^2\arctan\tanh^4\log_{10}(Ile_{12})\max^4(Ine.I_{05}\frac{\sqrt{|I_{05}|}}{10})\cos^8\sin^2(\frac{x}{\sqrt{1-x^2}})(Inw-I_{09})^8}) - Ils +$$

$$2Ile_{12}, 0.5(0.5(I_{09} + Inw) + \max(and(or(Imed, I_{05}), Inw - xor(D, \cos\tan(\frac{2In}{Inw+Imed}))), 0.5(Iw +$$

$$In) + 0.25(Ine - Inw), \log_{10}(I_{08}) + \sqrt{|\arcsin\tan\sinh x|} + 0.5(Inw + Ils))) + |\arccos(\frac{\max(Igap, Iw)}{I_{04}})|^{\frac{1}{4}} +$$

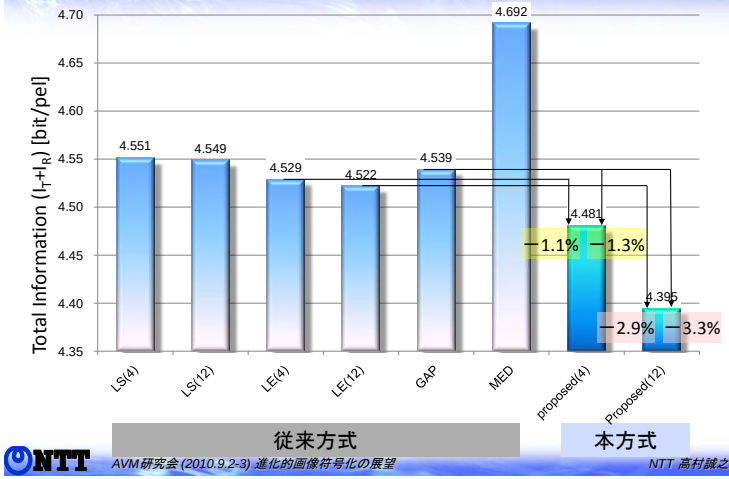
$$\theta), \min(\max(-\max(Inw, I_{06}, I_{07}) + and(Ile_{12}, \log\sinh(0.5(-xor(D, \cos\cosh(\frac{In^{\frac{1}{4}}}{|or(I_{05}, I_{09})|^{\frac{1}{4}}})) +$$

$$Inw + Igap) + \sinh\sinh(0.5(\arcsin\sinh\tan y + \theta)) - \sinh\arccos\arcsin x)) +$$

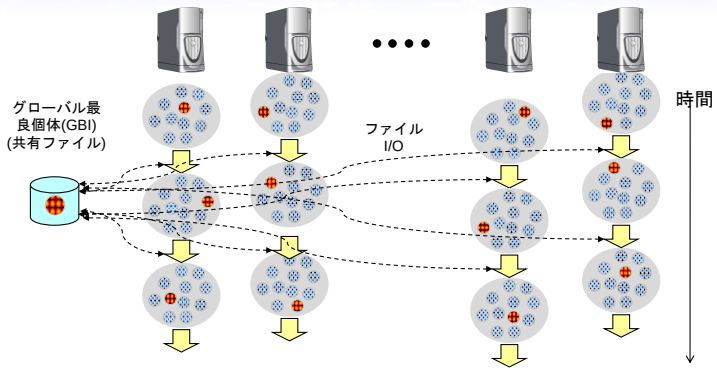
$$\max(Ile_{12}, \min(\max(Imed, In), \sinh\sqrt{|D|})), \tanh\tanh\sin(0.5(Iw + In) + 0.25(Ine - Inw) -$$

$$Ile_{12}) + Igap), \max(0.5(Iw + In) + 0.25(Ine - Inw), -12.8182(I_{10} - 0.5(I_{05} + Ine))))))$$

生成アルゴリズムの性能



実際の並列化システム



ⓒ 2010 Information Processing Society of Japan

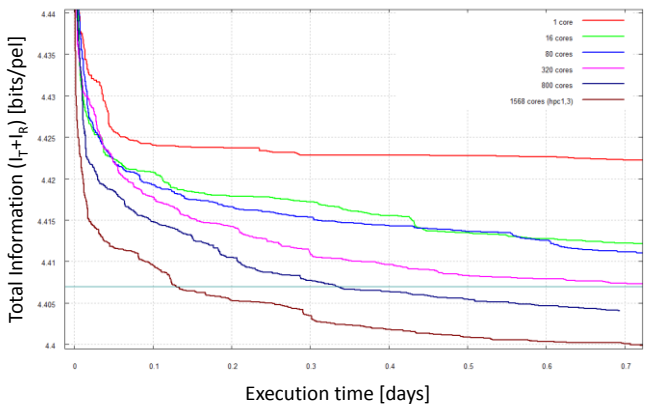
本並列システムの特徴

- ファイル共有によるプロセス間通信
  - ソースコードも実行環境もOSに依存しない(標準関数fopen()や_stat()を使用、ファイル共有はNFSに一任)
  - プロセスの途中離脱・追加も自由(スケーラビリティが1~∞)
- プロセス間の同期なし
  - 各コアのloadは常に100%
  - 各コアが非対称でも問題ない
- ファイルI/O負荷は無視できる程度
  - 各プロセスは一回の世代交代(～10秒)の後、一回共有ファイルのタイムスタンプを取得
  - 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

NTT AVM研究会 (2010.9.2-3) 進化的画像符号化の展望 NTT 高村誠之 27

並列化効果(拡大図)

Image: Lena

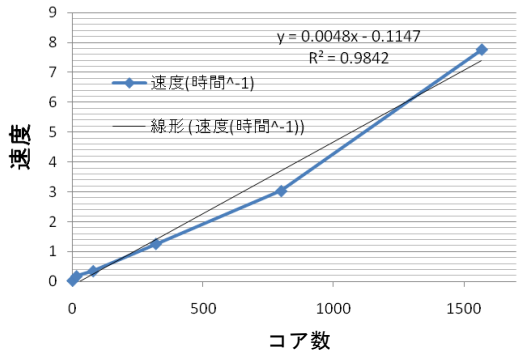


NTT AVM研究会 (2010.9.2-3) 進化的画像符号化の展望 NTT 高村誠之 32

コア数と速度の関係

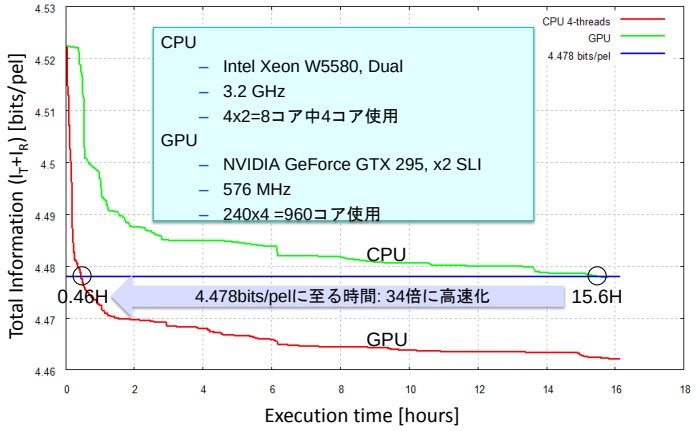
Image: Lena

速度 =「4.407bit/pelに達するまでの時間(日)の逆数」



NTT AVM研究会 (2010.9.2-3) 進化的画像符号化の展望 NTT 高村誠之 33

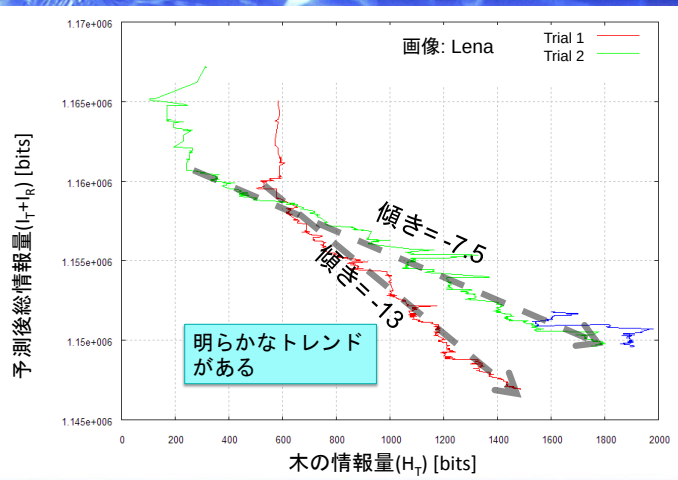
GPUによる高速化



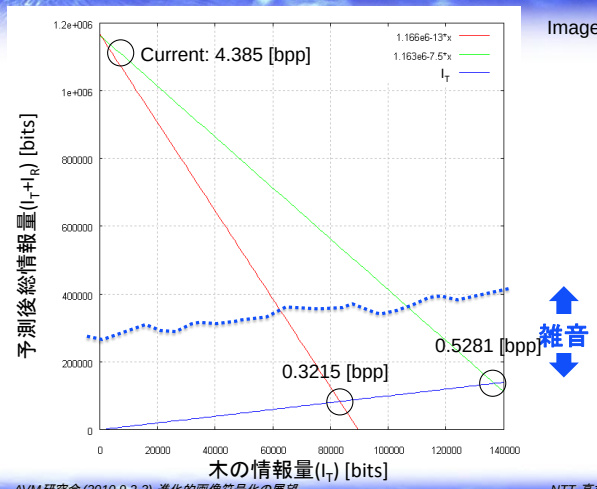
NTT AVM研究会 (2010.9.2-3) 進化的画像符号化の展望 NTT 高村誠之 35

ⓒ 2010 Information Processing Society of Japan

木の情報量 vs. 総情報量

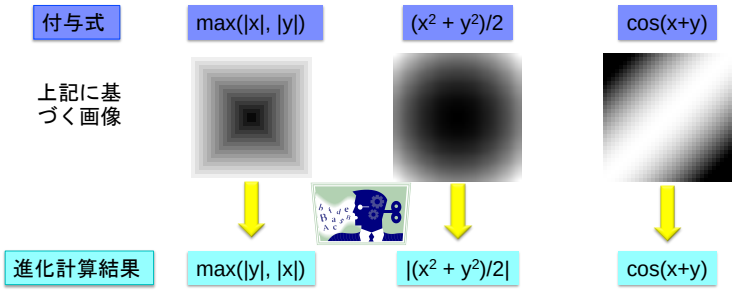


究極の圧縮率の外挿



GPのモデル推定能力

- 人が式(モデル)により与えた画像を入力とし、進化計算
- 元の式を逆導出できることを確認



まとめ・今後の課題

- 符号化アルゴリズムの自動構築の紹介
- 進化符号化の高速化
  - 並列化システム
  - 従来方式比の性能改善: 約3%
  - 資源利用効率の高さ
  - 進化速度改善: コア数にほぼ比例
  - GPUによる最新の高速化結果
- 究極の圧縮率に関する考察
- 今後の課題
  - 非可逆符号化への展開

ⓒ 2010 Information Processing Society of Japan