LC_005

ハイブリッド並列プログラミングによるMPEG2エンコーダの高速化
Speed Improvement of MPEG2 Encoding using Hybrid Parallel Programming

山崎 勝弘† 池上 広済† 小柳 滋†
Katsuhiro Yamazaki Kosai Ikegami and Shigeru Oyanagi

1．はじめに

近年，PCやハードディスクの低価格化・高性能化により，放送番組制作においてビデオ編集をコンピュータ上で行うノンリニア編集が広く普及するようになっている．多チャンネル化と高精細度(HD)化が進む現在，いかに制作作業効率を向上させるかが特に重要な課題となっている．しかし，HD画質での非圧縮ビデオ信号は膨大なデータ量となるためストレージへの記録時間が短い．そのため，編集段階での画質をサポートするMPEG2を用いたノンリニア編集システムが必要とされている．

現在の放送番組制作における問題点は，制作が完了してから放送局に持ち込むまでに時間がかかってしまうことである． HDノンリニア編集システムには，制作段階でのワークフローの向上のため，MPEG2エンコードをできる限り短時間で処理できる性能が求められている[1].

一方，近年，ノード内に複数のプロセッサを搭載したSMPノードを複数有するSMPクラスタが，価格，性能比に優れた並列マシンとして期待されている． SMPクラスタの性能は，LINPACKやNAS Parallel Benchmarks等の性能は公表されている[3][4]が，JPEG，MPEG等に代表されるマルチメディア系アプリケーションにおける性能はあまり明らかになっていない．MPEG2は可変長符号化を行うため並列化は難しいが，ランダムアクセスの実現やエラー耐性のために，GOP，スライス，マクロブロックを単位として，独立に処理を行うことができる[2]．関連研究として，チップマルチプロセッサ上で，データ局所化と，タスク実行・データ転送のオーバーラップによる性能向上を図った研究[5]がある．

本研究では， MPEG2エンコーダを対象として，ハイブリッド並列プログラミングによる高速化の有用性を実験的に検証することを目的とする．本稿では，まずMPEG2エンコーダのアルゴリズム，及びSMPクラスタ上でのハイブリッド並列化手法について述べる．本研究では，MPIによりGOPを単位として各ノードに担当範囲を割り当て，ノード内ではOpenMPによりブロックを等分割してエンコードの各処理を並列化する．16ノード，32プロセッサのSMPクラスタ上で64秒の映像を対象とした実験を行い，ハイブリッド実行とMPIのみの実行を比較評価する．そして，動きの少ない風景映像に対してハイブリッド並列化が有効であることを示す．

2．MPEG2エンコーダのアルゴリズム

図1に示すように，MPEG2エンコードは主にフレーム間予測，DCT，量子化，逆量子化，IDCT，動き補償，及び可変長符号化の7つの処理からなる[2].

MPEG2のデータ構造は，図2に示すような階層構造を

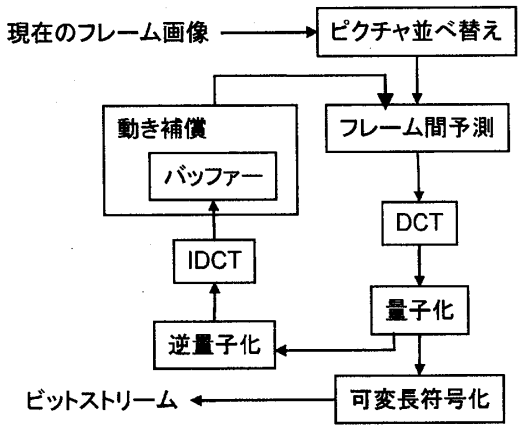


図1：MPEG2エンコーダのアルゴリズム

とっている．MPEGビデオは複数のGOP（Group of Picture）で構成されている．GOPは複数のフレームから成り，各フレームは複数のスライスから成る．各スライスは複数のマクロブロックから成る．各マクロブロックは4つの輝度ブロックと2つの色差ブロックから成り，各ブロックは８×８の大きさである．

1. フレーム間予測
   現在のフレーム画像と以前のフレーム画像との差分を取り，現在のフレーム画像内で類似しているブロックを探索して動きベクトルを決定する．また，順方向予測，逆方向予測，双方向予測を行う．逆方向・双方向予測ではピクチャ並べ替えを行う．
2. DCT
   マクロブロック内の6個のブロックに2次元DCTを行う．DCTは，画像信号を少数の低域係数に集中させる働きを持ち，画像の空間的な情報量を削減する．
3. 量子化
   DCT係数に対して割り算を行う．予測を行わないフレームでは丸め付き，予測を行うフレームでは丸め無しの割り算を行う．
4. 逆量子化
   再現値が偶数であった場合，0に近づくように奇数のみに再現し，IDCTの誤差を小さくする．
5. IDCT
   DCTの逆変換．標準規定では，DCTを規定せずIDCTの精度を規定している．よってIDCTの精度が十分でないとき，再生画像に誤差が生じる．
6. 動き補償
   動きベクトルを利用して参照フレームをIDCT係数より再現する．
7. 可変長符号化
   オルタネートスキャンにより係数値を並べ替え，ランレングス圧縮を行う．DCTにより，大部分の係数値が0となるので，非0の値と0の個数との組み合わせを考慮して可変長符号化を行う．

† 立命館大学大学院理工学研究科, Graduate School of Science and Engineering, Ritsumeikan University

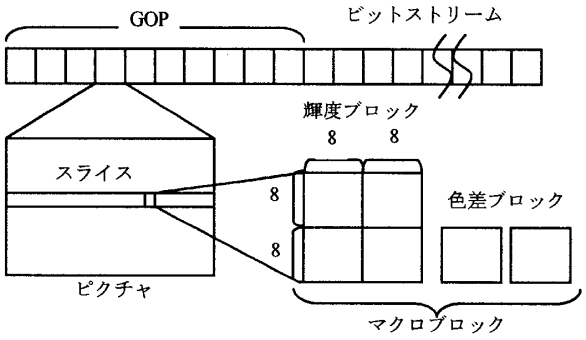


図 2: MPEG2 のデータ構造

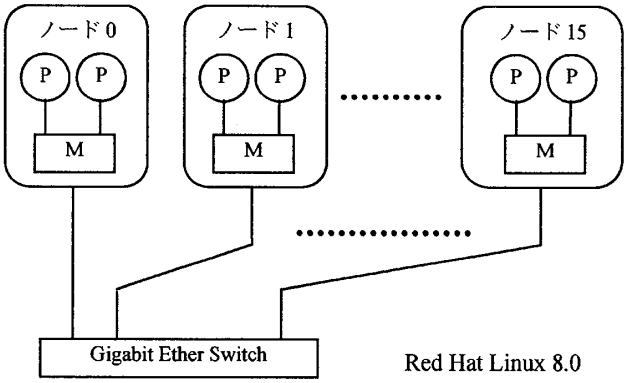


図 3: SMP クラスタ Atlantis の構成

3 . MPEG2 エンコーダのハイブリッド並列化

3.1 SMP クラスタによるハイブリッド並列化

本研究で使用する SMP クラスタの構成を図 3 に示す．16 台のノードがギガビットイーサで接続され，各ノード内にプロセッサ（Intel Xeon 2.8GHz）2 台と共有メモリ 2GB がある．従って，ノード内は共有メモリ，ノード間は分散メモリである．

本研究では，分散メモリ並列プログラミングには MPI を，共有メモリ並列プログラミングには OpenMP を用いる．本研究では，MPI（MPITCH ver1.2.5）により GOP を単位として各ノードに担当範囲を割り当てて並列化し，ノード内では OpenMP（Intel C++ Compiler ver9.0）によりブロックを単位としてエンコードの各処理を並列化する．

3.2 MPI によるノード間並列化

MPEG2 のような蓄積メディア用の画像符号化技術では，時間軸の制約がなく，早送り，巻き戻し，途中からの再生が要求される．これを満たすために MPEG2 では GOP を単位として独立に再生することができる．よって GOP 間でのデータ依存はないため，GOP を単位として分割すれば並列にエンコードすることができる．

ブロック分割では全ての GOP 数をノード数で割り，各分割単位を先頭から順番に各ノードに割り当てる．サイクリック分割では，GOP 1 個ずつを各ノードに順番に割り当て，これを最後の GOP まで繰り返す．ブロック分割ではデータの分配は 1 回だけで済むが，サイクリック分割では，分配が GOP 数/ノード数の回数発生する．

3.3 OpenMP によるノード内並列化

1 枚のフレームは複数のスライスに分けられているが，あるスライスが伝送系のエラーによって崩れたとき，次のスライスからは回復できるように規定されているため，スライス毎に独立にフレームを再現できる．よって，スライス間でのデータ依存はない．また，エンコーダの各処理はマクロブロックを単位としているため，マクロブロック間でのデータ依存はない．従って，本研究ではマクロブロックを単位として分割し，並列にエンコードを行う．すなわち，各ブロックを上半分と下半分に等分割し，それぞれをノード内の 2 つのプロセッサで並列処理する．

4 . 実験条件

入力動画は，自然の風景を撮った動きの少ない映像であり，解像度は Main Profile における High1440 Level(MP@HL)，Main Level(MP@ML)，Low Level(MP@LL)の 3 種類で，1920 フレーム(64 秒間)である．表 1 に各 Level の設定を示す．これら 3 つの動画に対して，ハイブリッド並列プログラムで実行した場合（ハイブリッド実行）と，MPI のみで実行した場合（MPI 実行）を比較する．ハイブリッド実行では，各ノードに MPI プロセス 1 個を，ノード内に OpenMP スレッド 2 個を生成する。MPI 実行では各ノードに MPI プロセス 2 個を生成する。ノード数を 1 から 16 に変化させ，ブロック分割とサイクリック分割それぞれで実験を行った．

表 1: Main Profile における各 Level の設定

パラメータ＼Level	High 1440	Main	Low
最大ビットレート(Mbps)	60	15	4
最大画素数/ライン	1440	720	352
最大ライン数/フレーム	1088	576	288
最大フレーム数/秒	60	30	30

5 . 実験結果

5.1 実行時間と速度向上

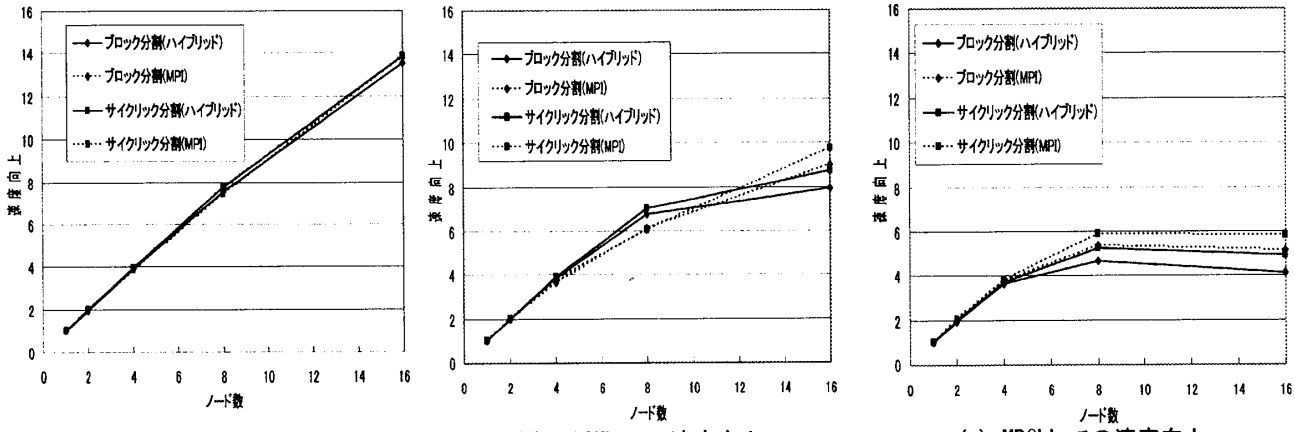
実行時間を表 2 に，ハイブリッド実行と MPI 実行における速度向上を図 4 に示す．

ハイブリッド実行の方が MPI 実行よりも殆どの場合で，実行時間が短い．特に，1 ノードでは実行時間が 11～17%短縮されている．MPI 実行の方が速いのは，LL で 16 ノードの場合だけである．ブロック分割とサイクリック分割を比較すると，ML と LL ではサイクリック分割による負荷分散の効果が現れている．すなわち，ML の 8 ノード以下ではブロック分割が速いが，16 ノードではサイクリック分割の方が速い．

速度向上の点では，HL では 8 ノードで 7.7 倍，16 ノードで 13.8 倍とほぼリニアな速度向上が得られている．ML では負荷均衡化の効果が見られる．LL では 8 ノードでの速度が最も速く，それ以上では速度向上が得られない．

表2：実行時間　　単位：秒

	ノード数	1	2	4	8	16
MP@HL	ブロック分割(ハイブリッド)	6014.4	3094.3	1543.0	795.9	444.2
	ブロック分割(MPI)	7014.9	3551.5	1791.3	905.7	508.5
	サイクリック分割(ハイブリッド)	6344.9	3247.2	1611.7	819.1	458.5
	サイクリック分割(MPI)	7137.1	3673.9	1863.7	961.1	514.5
MP@ML	ブロック分割(ハイブリッド)	1248.5	629.3	322.5	184.9	158.0
	ブロック分割(MPI)	1428.6	721.6	389.9	234.7	159.1
	サイクリック分割(ハイブリッド)	1314.7	661.9	338.4	187.3	150.3
	サイクリック分割(MPI)	1501.1	762.9	397.0	247.7	154.3
MP@LL	ブロック分割(ハイブリッド)	291.4	150.5	80.1	62.8	70.9
	ブロック分割(MPI)	349.7	177.8	93.5	65.2	68.2
	サイクリック分割(ハイブリッド)	315.8	158.9	85.6	60.1	63.9
	サイクリック分割(MPI)	372.0	183.1	97.5	63.4	63.7



(a) MP@HL での速度向上　(b) MP@ML での速度向上　(c) MP@LL での速度向上

図4：　ハイブリッド実行と MPI 実行における速度向上

5.2　サイクリック分割における分割粒度

サイクリック分割における分割の最小単位を GOP1 つから 4 つに変化させた場合と，ブロック分割の実行時間と速度向上を表 3 に示す．入力動画は MP@ML，1920 フレームのものを用いる．

16 ノードでは 4GOP を最小分割とした場合が実行時間，速度向上共に最も良い．8 ノード以下ではブロック分割が最も速い．

5.3　動きの激しい画像の処理

入力動画を動きの激しいサーフィン映像に変えた場合の実行結果を表 4 に示す．ここでのサイクリック分割は，1GOP によるものである．入力動画は，MP@ML，1920 フレームのものを用いる．

ハイブリッド実行と MPI 実行を比較すると，8 ノード以下では MPI 実行の方が速い．16 ノードでは，ハイブリッド実行・サイクリック分割が最速である．すなわち，ハイブリッド実行は動きの少ない画像に有効であること分かった．MPI 実行でブロック分割とサイクリック分割を比較すると，8 ノードでほぼ同じで，16 ノードでサイクリック分割の方が速くなる．動きが激しい映像では，フレーム間予測における差分が大きいため，サイクリック分割による負荷分散が有効に働くものと考えられる．

表3：サイクリック分割の最適化

実行時間　　単位:秒

ノード数	1	2	4	8	16
1GOP	1314.7	661.9	338.4	187.3	150.3
2GOP	1366.2	689.8	361.0	208.2	149.1
4GOP	1362.9	683.4	355.6	195.0	147.0
ブロック分割	1248.5	629.3	322.5	184.9	158.0

速度向上

ノード数	1	2	4	8	16
1GOP	1	1.99	3.88	7.02	8.75
2GOP	1	1.98	3.78	6.56	9.16
4GOP	1	1.99	3.83	6.99	9.27
ブロック分割	1	1.98	3.87	6.75	7.90

表 4: 動きの速い入力動画による実行結果

実行時間　　単位:秒

ブロック分割					
ノード数	1	2	4	8	16
ハイブリッド	1519.9	774.6	421.4	254.1	165.4
MPI	1440.7	728.3	371.7	206.1	161.1
サイクリック分割					
ハイブリッド	1643.4	835.2	423.2	224.9	155.3
MPI	1496.5	742.1	380.3	207.0	155.5

速度向上

ブロック分割					
ノード数	1	2	4	8	16
ハイブリッド	1	1.96	3.61	3.61	9.19
MPI	1	1.98	3.88	3.88	8.94
サイクリック分割					
ハイブリッド	1	1.97	3.88	3.88	10.58
MPI	1	2.02	3.94	3.94	9.62

6．考察

（1）ハイブリッド並列プログラミング

MPEG2 エンコーダでは，動きの少ない風景映像を対象とした場合に，ハイブリッド実行により MPI 実行と比べ，10%～20%実行時間を短縮することができ，OpenMP を併用した方が高速であることが確認できた．高い並列化効果が得られることで有名な多くの問題において，MPI と OpenMP を組み合わせたハイブリッド並列プログラミングを用いた場合，あまり性能が良くならないか，逆に下がってしまうことが多い[3]．今回の結果は，共有メモリ並列プログラミングと DCT，IDCT の相性が良く，理想的な速度向上が得られるため，MPI のみの並列化よりもハイブリッド並列化の方が良くなったと考えられる．

動きの少ない森の風景映像では，全体が同じような色なので，DCT による圧縮率が高い．一方，動きの激しい波のサーフィン映像は複雑な映像なので，圧縮率が低い．
すなわち，1 コマの静止画像の圧縮率が高いほど，ハイブリッド実行が有効であると考えられる．

ML では 8 から 16 ノードで速度向上率が低下（図 4(b)）し，LL では 8 ノードで最大の速度向上を示している（図 4(c)）．16 ノードでのノード当りのデータサイズは，ML,LL でそれぞれ 120MB，36MB である．ML では 120MB 以下でハイブリッド実行の効果が低下し，LL では 36MB でハイブリッド実行が MPI 実行より遅くなっている．

すなわち，並列化効果は各ノードで処理するデータサイズとのトレードオフとなるため，ハイブリッド実行による十分な効果を出すためには，ノード数に応じて分割粒度を変更する必要がある．

（2）速度向上

速度向上は，MPI によるサイクリック分割での結果が最も良い．LL では 8 ノードで 5．8 倍，ML では 16 ノードで 9．7 倍，HL では 13．7 倍の速度向上となっている．LL では 4 ノードまで，ML と HL では 8 ノードまではほぼ理想的な速度向上が得られた．すなわち，データサイズが大きいほど理想的な速度向上に近づくと考えられる．また，動きの激しい映像では，前フレーム画像との差分が大きく，フレーム間予測の処理量が大きいため，動きの少ない映像に比べて，高い速度向上が得られた．

（3）ブロック分割とサイクリック分割

全ての場合で，4 ノードまではブロック分割の方が速い．サイクリック分割の方が速いのは表 2 から ML で 16 ノード，LL で 8 ノード以上の場合である．

サイクリック分割では，分割の最小単位を 1～4 個の GOP としており，16 ノードでは 4 個の GOP での結果が最も良かった．しかし，GOP 内のフレーム数は可変であり，総フレーム数が増えた場合は数十～100 フレーム程度まで増やすことがある．このような場合において最小分割単位を変化させ，総フレーム数，GOP 内のフレーム数，及び最大ノード数に対する最適な分割方法の検討が必要である．

7．まとめ

本研究では，ハイブリッド並列プログラミングを用いた MPEG2 エンコーダの並列化手法について検討し，2Way16 ノードの SMP クラスタ上に実装して，MPI 実行との比較による性能評価を行った．

動きの少ない風景映像では，ハイブリッド並列プログラミングにより，ブロック分割，サイクリック分割ともに MPI 実行に比べて 10～20%実行時間が短縮された．また，1 ノード当りのデータサイズが 120MB 以下の時ハイブリッド実行による効果が小さくなることを確認した．ブロック分割とサイクリック分割を比較すると，4 ノードまではブロック分割による並列化の方が実行時間は短いが，8 または 16 ノード以上ではサイクリック分割の方が 10%程度高速であるという結果が得られた．また，Main Profile における各 Level の速度向上の最大は，HL で 13.7 倍（16 ノード），ML で 9.7 倍（16 ノード），LL では 5.8 倍（8 ノード）の結果が得られた．

今後の課題として，他の圧縮コーデックやレンダリングに時間のかかるトランジションやエフェクト，フィルタなども合わせて高速化し，トータルとして快適なビデオ制作環境を構築することが挙げられる．

参考文献

[1] 清水文雄他:”小特集 最近の放送番組制作技術”，映像情報メディア学会誌，Vol.56，No.10，pp.1542-1548，2002.
[2] 藤原洋，安田浩:“ポイント図解式 ブロードバンド+モバイル 標準 MPEG 教科書”，アスキー，2003.
[3] 吉川茂洋他:“SMP-PC クラスタにおける OpenMP+MPI の性能評価”，情報処理学会研究報告，Vol.2000，No.023，pp．155-160，2000.
[4] Ta Quoc Viet et al.:“Construction of Hybrid MPI-OpenMP Solutions for SMP Clusters”，情報処理学会論文誌，Vol.46，No.SIG3，pp.25-37，2005.
[5]小高剛他:“チップマルチプロセッサ上での MPEG2 エンコードの並列処理”，情報処理学会論文誌，Vol.46，No.9，pp.2311-2325，2005
[6]池上広済他:“PC クラスタ上での JPEG エンコーダ・デコーダの並列化”，FIT2005，No.A-030，pp．69-72，2005.