分散画像処理環境 VIOS III とその性能評価

6 Z − 7

中田浩作 松尾啓志 岩田彰

名古屋工業大学電気情報工学科

1 はじめに

画像処理やパターン認識の分野では，画素ごともしくは小領域を単位として並列実行可能なアルゴリズムが少なくない．そこで我々は，ネットワークで結合された複数の CPU を利用することにより，大規模な画像処理を実行する分散画像処理環境 VIOS を提案し，VIOS I，VIOS II の開発を行った [1].

現在我々は，VIOS I, II の欠点を改善すべく，VIOS III の開発を行っている [2]. VIOS III は次の特徴を持つ. (1) より柔軟な並列処理記述能力を有する VPE-p Ver3. (2) 分散画像処理に適した新しい分散型画像データ構造とプログラマブル大域変数バッファ.

VPE-P Ver3 は，柔軟な並列画像処理記述能力を有する並列画像処理記述言語であるため，C 言語による記述と比べて，ある程度の性能低下は避けられない．本発表では，C 言語およびメッセージパッシング型並列プログラム記述ライブラリ PVM との実行速度の比較を行うことにより，VIOS III の並列処理のオーバヘッドについて検討する.

2 VIOS のシステム構成

VIOS は，VPE，IPU，OM の 3 種類のプロセスから成り，ネットワーク上の複数のワークステーションに分散して存在する．VPE はユーザとのインタフェースプロセスである．IPU は実際の画像処理を行う画像処理エンジンプロセスである．IPU をネットワーク上に複数配置することにより分散処理を実現する．OM はシステムの管理プロセスである.

3 並列画像処理記述言語 VPE-p Ver3

VIOS では，画像を (1) データ並列の依存関係，(2) 処理の中心となる画素集合（以後注目画素），(3) 参照だけを行う画素集合（以下周辺画素）の 3 つの情報を持つ並列画像処理ワーキングセット（以後ワーキングセット）と呼ぶ最小単位まで分割することにより分散処理を行う手法を提案している.

VIOS III では，画像処理アルゴリズムが有する様々な並列性の記述を可能にするために並列画像処理言語 VPE-p に以下の拡張を行った.

- 処理モジュール内に逐次処理記述部を導入
  逐次的な処理を含む演算を，処理モジュール内に記述できるように，モジュール内をワーキングセットごとの記述を行う複数の並列処理記述部と，ワーキングセットによる分割を行わない逐次処理記述部に別けて記述可能とした.
- 非同期実行
  モジュール間の依存関係を記述し，インタプリタとして動作するメインフロー記述部に非同期実行を導入する．ネットワーク分散処理のように，計算資源が多く配置できる場合には性能向上が期待できると考える.
- ネットワーク透過型大域変数と周辺画素以外の画素データ参照の導入
  ネットワーク上に分散配置されている複数のワーキングセット間での共有可能な変数として，モジュール内で大域的にアクセスできる大域変数を定義する．さらに，ワーキングセットが有する周辺画素以外の画素データの参照も可能とする.
- プログラマブル大域変数バッファ
  画像処理には大域変数のアクセスに必ずしも厳密な排他制御を必要としない場合や，ある処理単位の終りに，適切な規則に基づいて各 IPU 内に設けたバッファを統合することにより大域変数へのアクセスを IPU 内の局所変数へのアクセスに置き換え可能な場合も少なくない．そこで VIOS III では，分散画像処理環境に適したプログラマブル大域変数バッファを提案し，実装を行った.
- ワーキングセット外の画像データに関する様々なアクセスポリシ
  プログラム内で，ワーキングセット外のデータが参照された際の振る舞いを，ネットワーク経由での参照，補間，定数代入など，画像処理アルゴリズムに応じて記述可能とした.

図 1は，512×512 画素からなる画像を 32×32 画素からなる box ワーキングセットに分けて，最大値検出

A Distributed Image Processing Environment VIOS III

Kosaku Nakada, Hiroshi Matsuo, Akira Iwata
Dept of Electrical and Computer Eng., Nagoya Institute of Technology,
Gokiso-cho, Showa-ku, Nagoya, 466, Japan

モジュール名	記述言語						
	C言語	VIOS III			PVM		
	CPU数						
	1	1	2	4	1	2	4
ソーベルフィルタ	0.69	0.78(1.0)	0.47(1.7)	0.29(2.7)	0.73(1.0)	0.38(1.9)	0.20(3.7)
細線化	0.09	0.15(1.0)	0.21(0.7)	0.29(0.5)	0.14(1.0)	0.08(1.8)	0.06(3.5)
大津の２値化	0.06	0.42(1.0)	0.56(0.8)	0.67(0.6)	0.10(1.0)	0.06(1.7)	0.05(2.0)
ハフ変換	6.44	6.97(1.0)	3.60(1.9)	2.10(3.3)	7.44(1.0)	6.22(1.2)	6.58(1.1)
局所最大値検出	2.05	2.86(1.0)	1.53(1.9)	0.82(3.5)	2.21(1.0)	1.42(1.6)	0.81(2.7)
最大値検出	0.05	0.13(1.0)	0.15(0.9)	0.26(0.5)	0.06(1.0)	0.04(1.5)	0.03(2.0)
合計	9.38	11.31(1.0)	6.52(1.7)	4.43(2.6)	10.68(1.0)	8.20(1.3)	7.73(1.4)

表 1: 並列認識処理過程の実行時間（単位：秒，カッコ内は，CPU 台数 1 との速度比）

を行う画像処理モジュールの記述例である．

```
module Max(a:input, x:output)  モジュール宣言部
int a on box[32][32];          引数宣言部
int x;
{
 int max[16][16];              大域変数宣言部
 int i,j;
 parallel{                     並列処理記述部
 }
 x = max[0][0];                逐次処理記述部
 for (i = 0; i < 16; i++)
  for (j = 0; j < 16; j++)
   x = (x > max[i][j]) ? x:max[i][j];
}
```

図 1: 処理モジュール定義例（最大値検出）

4 VIOS III の性能評価

VIOS III の性能評価を行うために，松山らの実装した並列認識処理過程 [3] を，(1) 非並列化アルゴリズムを C 言語により記述，(2) 並列アルゴリズムを C 言語および PVM を用いて記述，(3)2 と同じアルゴリズムを VPE-p Ver.3 により記述，の 3 種類の方法で実装し，実行時間の比較を行った．実験環境として，CPU:PentiumPro 200MHz，メモリ:128Mbyte，OS:Solaris 2.5.1 の計算機を，100baseTX のスイッチングハブで構成されたネットワークにより結合された環境を用いた．

結果を表 1および図 2に示す．CPU 台数が 1 台の場合は，VIOS では，非並列化アルゴリズムを C 言語で記述した場合に比べて，約 20%，C 言語および PVM を用いた場合に比べて 5%の性能低下が見られた．この原因として，VIOS では，たとえ CPU が 1 台であっても汎用的な並列化コードを用いている，また記述言語に C++を用いていることが考えられる．

また，CPU の台数が変化した時には，VIOS による実装が，C と PVM を用いた実装より，良い速度向上結果が得られた．この理由として，ハフ変換時に大量のテーブルを転送する際，VIOS ではネットワーク部分をスレッドとして実装しているため，並列に転送可能であることが原因であると考える．

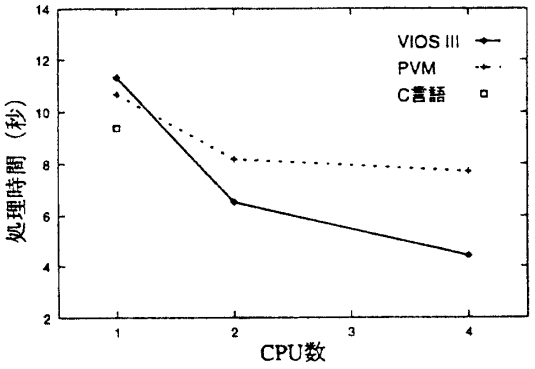

図 2: C 言語および PVM との比較

5 まとめ

並列画像処理記述言語 VPE-p の拡張を行い，様々な並列性を持つ画像処理アルゴリズムの記述が可能であることを確認した．また，C 言語および PVM を用いた実装との比較を行うことにより，VIOS III の有効性を示した．

参考文献

[1] 例えば H.Matsuo and A.Iwata: “A distributed image processing enviroment VIOS II”, ACCV93, pp.715-718(1993)

[2] 山本伸一ほか: “分散画像処理環境 VIOS III の開発”，情報処理学会 CVIM 研究会，Vol.97, No.31，pp.63-70(1997)

[3] 松山隆司ほか: “再帰トーラス結合アーキテクチャにおける並列対象認識のためのデータレベル並列プロセスの構成”，情報処理，Vol.36，No.10，pp.2310-2320(1995)