並列論理プログラムの特性解析（2）

—— 小規模プログラムの解析例 ——

5Q-10

市吉伸行　和田久美子　芳崎かすみ　瀧和男
（ICOT）　（沖電気）　（ICOT）

1．はじめに

我々は，ネットワーク型並列マシンにおける並列論理型言語の効率的実行方式を研究しており，これまでにマルチＰＳＩ第一版にＦＧＨＣの処理系を実現した［1］．最終目標は大規模な問題を高速に解くことにあるが，そのためには多くのプロセッサに実質的な処理を行わせることが必要である．ところが，これらの要求はしばしば矛盾する．すなわち，小さい処理単位で分散を図ると，多くのプロセッサにより均一に仕事が割り振られるが，一方で通信のオーバーヘッドが増大する．我々は，今までに出てきたいくつかのプログラムのスタイルについて，逐次および分散処理系における動特性を測定し，プログラム・スタイルおよび負荷分散の違いが動特性に与える影響を調べた．

2．プログラム・スタイル

ここでプログラム・スタイルとは，プロセス構造やデータ・フローの観点から見たプログラムのクラスを指しており，アルゴリズムそのものやコーディング・スタイルとは別のものである．我々がこれまでに書いたＦＧＨＣの主なプログラム・スタイルの内には以下のようなものがある．

（1） 制限ＡＮＤ並列型
（独立なＡＮＤ並列プロセスの集り）
（2） ストリーム・フィルタ型
（直列につながれたフィルタ・プロセスの集り）
（3） ネットワーク型
（隣接点と交信する均一なノード・プロセスの集り）
（4） レイヤード・ストリーム型
（部分解を共用する生成・試験型プログラム）

3．負荷分散

プログラム・スタイルがプログラムの論理構造の大枠を規定しているのに対し，負荷分散はプロセスの物理的プロセッサへのマッピングを示す．マルチＰＳＩ上のＦＧＨＣ処理系ではプラグマ［3］と呼ばれる記法によりボディ・ゴールのプロセッサへの割り当てを指示している．例えば，

```
compile(F,G,PE) :- true |
        PE1 is PE+1, PE2 is PE+2,
        alloc(PE1)@@generate_code(F,X),
        alloc(PE2)@@assemble(X,G).
```

というＦＧＨＣ節において，alloc(PE1)@@とalloc(PE2)@@がプラグマであり，PEが０だとするとgenerate_code/2 はプロセッサ１に，assemble/2はプロセッサ２に割り当てられる．プログラムの論理的意味はプラグマを除いたものと同じである．このプラグマ方式は，個々の節が子ゴールを実行すべきプロセッサを動的に決められるので細かいレベルまでプログラマーが負荷分散を指定できる利点がある．

4．動特性の測定結果

測定対象のプログラムは，
（1）素数生成プログラム（エラトステネスの篩をフィルタ・プロセスとしたプログラム）
（2）最適経路発見プログラム１（探索経路をプロセスとした枝刈りの少ないプログラム）
（3）最適経路発見プログラム２（ノードをプロセスとした枝刈りの多いもの．ショート・サーキット法による終了判定方法）
（4）Ｎクィーン・プログラム１（行の何カラム目にクィーンを置いたかで場合分けする制限ＡＮＤ型）
（5）Ｎクィーン・プログラム２（レイヤード・ストリーム法［2］）
の五つで，測定用のツールとして我々は［4］で述べてある並列度解析ツールおよびPseudoマルチＰＳＩ上のＦＧＨＣ分散処理系を用いた．前者で測定したのは（1）から（5）の全てで，後者は（4）と（5）の二つである．測定結果の一部を図１および２に掲げた．測定結果をまとめると次のようになる．

・並列度解析ツールによる測定
（i）一般に，実行可能なゴールの数は実行開始時点より増加し，ピーク達成後，減少する．（図1-a,b,c）

Analysis of Parallel Logic Programs (2) - Analyzing Small Programs
N.Ichiyoshi(ICOT), K.Wada(Oki Electric Ind.), K.Susaki, K.Taki(ICOT)

（ii）パイプ・ライン型のプログラムでは実行可能なゴールの数はパイプ・ラインの長さの伸びに比例して漸増し，パイプ・ラインの入力の終了に伴って急激に収束する．（図1-a）
（iii）枝刈りの少ない探索型プログラムでは実行可能なゴール数は急激に増加し，解の発見後急激に減少する．（図1-b）
（iv）ネットワーク型のプログラムではプロセス数はほぼ一定に保たれ，計算終了検出後，すみやかに収束する．計算中には過半数のプロセスがサスペンドしている．（図1-c）
これと同じ性質は［5］においても報告されている．

・分散処理系による測定

（i）制限ＡＮＤ並列型プログラムのＡＮＤプロセスを分散させると，ＡＮＤプロセスたちの計算量が均等であれば台数効果が得られる．（図2実線）
（ii）レイヤード・ストリーム法の単一プロセッサでにおける有効性が報告されているが［2］，分散実行で効率が上がりにくい．これは，同一のデータが多くのプロセスによって共有されておりデータ読み出しのための通信のオーバーヘッドがかなり大きいためである．（プロセッサによってはＣＰＵ時間の半分を通信処理に使っている．）負荷の均一化もレイヤード・ストリームでは簡単でないことがわかった．（図2破線）

5．おわりに

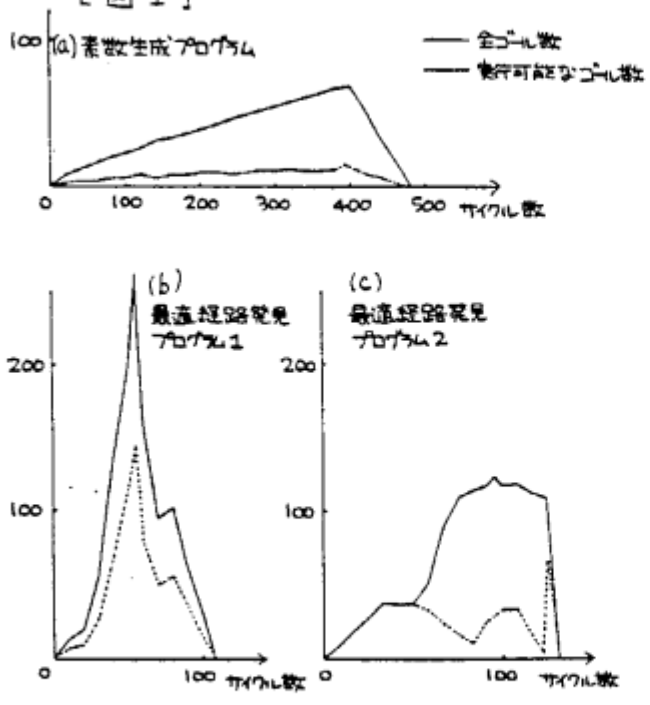
測定実験の結果，我々の分散処理系特有の問題点もいくつか発見された．一つは，データ生成場所の制御がしにくいことで，プロセス実行場所をプラグマによって指定しても処理するデータが異なるプロセッサに作られてしまうために通信量が不当に増えてしまうという現象がみられた．また，プラグマの拡張としてデータ生成場所を制御する機能やデータのあるプロセッサへ向けてゴールを投げるための機能の追加が必要になるかも知れない．

しかし，最大の困難は並列プログラム評価のために注目すべき適切なパラメータがまだ良くわかっていないことにある．今後は，より多くのプログラム・スタイルについて，またより大きなプログラムについて計測するとともに，解析方法および評価パラメータを開発していきたい．

参考文献

〔1〕 [illegible] 他．"Multi-PSIシステムの概要"，第32回情報処理学会(5B-8)，昭和61年3月
〔2〕 奥村 他．"レイヤードストリームを用いた並列プログラミング"，The Logic Programming Conference '87 (11.2)，June 22-24, 1987
〔3〕 N.Ichiyoshi, et al. "A distributed implementation of Flat GHC on the Multi-PSI", Proc. of the 4th International Conference on Logic Programming, J.L.Lassez ed., MIT Press, 1987, p257-275.
〔4〕 杉野 他．"並列論理型プログラムの特性解析(1) -動特性解析ツール-"，本大会予稿集．
〔5〕 M.Kishimoto, et al. "An Evaluation of FGHC via Practical Application Programs", the Fourth Symposium on Logic Programming, San Fransisco, August 1987

［図1］



［図2］