

5J-1

並行化プロダクションシステム
ARCH-3

中村明, 稲葉浩人
(株)東芝 システムソフトウェア技術研究所

はじめに

前向き推論の高速化と手続き言語との統合化を目指してプロダクションシステムARCHシリーズを開発してきたが、今回並列マシンによる並列プロダクションシステムの開発に先立ち、実験用として構想された並行化プロダクションシステムARCH-3の概要について報告する。

プロダクションシステムの高速化には文献[1][2]に述べられているように

- (1) 照合アルゴリズム
- (2) コンパイラ化
- (3) 並列化

のアプローチから研究が盛んに行われている。ARCHシリーズでは、ARCH-1でRETTEアルゴリズムを改良した照合アルゴリズムを開発し、ARCH-2では言語仕様を拡張しC言語コードを生成するコンパイラ化を実現し前向き推論の高速実行を高めてきた。この成果を生かし並列マシン上でのより高速なプロダクションシステムの開発を計画しているが実機による実装の準備段階として並行化シミュレータを開発しそこで並列化アルゴリズムの開発を行うこととした。

ARCH-2 コンパイラ

ARCH-2でのコンパイラアルゴリズムの特徴は、競合解消集合の中から、動的に設定される選択アルゴリズムによってinstantiationを選択し左辺部actionを開始した後は、2inputノード列に対応したC言語関数呼び出し列になっていることにある。即ち、RETTEアルゴリズムで我々が独自に定義したInitial、Flowトークンの流れは、関数呼び出し列に置き代わっている。我々の並列化の出発点は、この関数呼び出し列を並列実行することにある。Storedトークンに対する排他制御がRETTEアルゴリズムを用いたコンパイルの際には必要である。

XINUシミュレータ

上記の関数呼び出し列を並行実行するには、“軽いプロセス”(Light Weight Process)の機構が有効である。この“軽いプロセス”とプロセス間同

期手段の機構がまず必要であった。

そのためにComer[3]によって開発されたXINUシステムのシミュレータをUNIX上の1プロセス内に実現した。このXINUシミュレータには、

- (1) 複数スタック機構によって関数呼び出し列を軽いプロセスとして並行実行し、
- (2) それら軽いプロセスにたいする制御機構(create, resume, kill, suspend)と
- (3) 軽いプロセス間同士の同期手段(wait, semaphore, message)

が、用意されている。XINUシステムは、本来はLSI-11マシン用に設計されたオペレーティングシステムで、実際の割り込みやプロセス切り替えの処理を行うが、本シミュレータでは、割り込みの代わりに(特にタイマ)signal、プロセス切り替えにsetjmp、longjmpを用いている。

並列化アルゴリズム

上記のXINUシミュレータ上にARCH-2が生成するC言語関数列を並行実行する仕組みを組み込む。まずSelectProcessが、システム初期時に生成され、これは常に存在する。

- (1) SelectProcessは、競合解消集合にある全てのinstantiationに対応して左辺部のactionを実行するFireProcessをcreateしresumeする。
- (2) FireProcessはmake, remove関数に対応してInitialProcessをcreateしresumeする。Action処理を終えたら、FireProcessは、SelectProcessにsignalを出し自らをkillする。
- (3) InitialProcessは2input関数を呼び出す際にFlowProcessをcreateしresumeする。InitialProcess自身はSelectProcessにsignalをおくり自らをkillする。
- (4) FlowProcessは、2input関数を実行する。その際storedトークンを探索するがleft, rightメモリへのアクセスは、semaphoreによって排他制御されている。FlowProcessは、次の2input関数実行の為、新たなFlowProcessをcreate, resumeするか、term関数を実行してSelectProcessにsignalを送り自らをkillする。
- (5) SelectProcessは、送られてくるsignalで、全ての子孫プロセスが消滅したことが確認されたならば、再度(1)に戻り、新たな並列発火を行う。

このアルゴリズムは、並列発火、並列action実行、並列ノード実行、並列ノード内実行の実現を目指しているが、どの並列性が高速化に最も効果があるかを捜し出す目的がある。ARCH-2には、ルールセットの機構が設けられており、過大な並列実行が発生しプロセス数の著しい増大を引き起こすのを防ぐことができる。プロダクションシステムの並列化には、なるべく多くの並列化を引き起こす課題と、次にはそれを制御可能とする課題の相反するものが存在する。

システム構成

ARCH-3コンパイラは、上記の並列化アルゴリズムを組み込んだC言語コードを生成する。それには、semaphoreなどのXINUのデータ構造も含まれる。XINUシミュレータ本体と生成コードを同時にコンパイルして実行コードを作り出す。実行時のシステムは次の構成要素から成り立つ。

- (1) テキスト部 XINUシミュレータコード, LHS, RHSから成り立つルールコード。
- (2) データ部 left,rightメモリ, 大域変数などのルールデータとプロセス構造, semaphoreなどのXINU制御データ
- (3) スタック部 SelectProcess, FireProcess, InitialProcess, FlowProcessが生成消滅を繰り返す。

結論

本システムは試作の段階にあり、並列化がもたらす高速化への効果測定の予想までは至っていないが、並列化アルゴリズムの正当性の検証には有効である。本システムでは、照合アルゴリズムにRETEアルゴリズムを用いたがleft,rightメモリの排他制御の実装には、非常に面倒なものが有り、高速化の妨げになることが予想される。現在ARCH-2の照合アルゴリズムとして新たにTREATアルゴリズム[4][5]を、実装中であるが、TREATでの並列化アルゴリズムの研究をも進めていく予定である。

参考文献

- [1]Gupta A. :Parallellism in Production system, Pitman,(1987)
- [2]石田, 他, :プロダクションシステムの高速化技術, 情報処理29.5, pp467-477(1988)
- [3]Comer,D. :Operating System Design: The XINU approach,Prentice-Hall(1984)
- [4]Miranker,D.P. ,:TREAT A Better Match Algorithm for AI Production System, AAAI87,42-47,(1987)
- [5]Miranker,D.P. ,:Recent Developments in Parallel Production System Algorithm, Technical Report, Dept.of Computer Science, University of Texas at Austin,(1987)

