7M−8

メッセージフローエージェントのデータフロー実行方式

日下部茂　友清孝志　谷口倫一郎　雨宮真人
九州大学総合理工学研究科

1 まえがき

知識処理など高度の情報処理システムとして，並列協調型処理システムが有望視されている．並列協調処理システムは，多数の処理体(agent)が互いに通信を行いながら情報を交換して自律的に動作し，協調して問題を解決する[1]．このような並列協調型の計算では各処理体が並行して計算を進めるため高い処理能力が期待できる．協調型の計算モデルとしては様々なものが提案されているが，我々はactorモデル[2]のようなメッセージ交換に基づく並列計算システムの実現を目指している[5]．

我々はこのようなメッセージフローによる協調計算システムを実現する並列処理計算機の基礎方式としてデータフロー方式が有用であると考えている．データフローアーキテクチャは並列処理の同期制御をデータ依存関係にしたがって自然に実現できるという利点を持っており，非同期メッセージ交信を行うメッセージフローシステムの実現に適している．システムを構成する各処理体を並列計算機が持つ並列に動作可能な物理的計算資源に割り付け，処理体間のメッセージ交換をその計算資源を結ぶネットワークを介して行うことにより処理体間の並列性も活用することができる．また各処理体内の演算をデータ駆動メカニズムで処理することにより処理体内での並列性を活用することができる．このようにしてデータフロー方式に基づく並列計算機上にメッセージフローシステムを実現することで，処理体内及び処理体間の2つのレベルでの並列性を活用する協調計算システムの実現を目指している．

2 メッセージフロー

メッセージフローによる並列協調型システムを構成する処理体は次のような性質を持つものとする．

- それぞれ固有の内部状態を持つことができる．自分自身の内部状態には直接アクセスできるが，自分以外の処理体の内部状態には直接アクセスできない．
- それぞれの処理体は独立していて，並列に処理を行う能力を持つ．
- 各処理体は任意の処理体と交信することができる．メッセージにより情報の交換を行い，メッセージの内容に従って処理を行う．

メッセージフローシステムではこのような処理体の間にメッセージが流れそのリンクは動的なものである．処理体は固有の内部状態といくつかの関数を持つことができ，実行時に動的に生成，消去される．処理体はメッセージを受理することにより実行を開始し，メッセージに応じた状態の更新やメッセージ送信，新たな処理体のインスタンスの生成などを行う．

そのようなシステムをプログラミングするため，関数型言語Validにオブジェクト指向的な概念を導入しmoduleという新しいプログラム単位を加え前述のような処理体が容易に記述できるように拡張した[6]．またメッセージ送信の記述を加え，メッセージによる実行式の選択が容易に記述できるようにパターンマッチの機能を導入した．処理体の生成，消去の記述等も加えた．Validでは単一代入則をとっており状態値の変更を副作用として記述することは出来ないため，処理体の本体を再帰式として記述し，状態値を引数として自分自身を再帰的に呼ぶように記述する．

3 計算メカニズム

前述のようなメッセージフロー協調計算システムを実現するには状態を持つ処理体を実現する必要があるが，データフローアーキテクチャでは状態を直接保持することは出来ない．また処理体は次々に送られてくるメッセージのストリームを処理する必要がある．このような計算システムを再帰とストリーム処理の概念を用いたデータフロー方式によって以下のように実現することができる．

処理体のインスタンスIへはメッセージを介したアクセスだけを許し，内部の処理の詳細は外へ見せない．処理体のインスタンスは概念的にはメッセージバッファ部Qと処理ルーチン部Pの2つに大別される．処理体に送られるメッセージm_iはバッファの中に取り込まれ，キューに並べられる．処理体のインスタンスIはメッセージm_iを受け取るごとに処理ルーチンのインスタンスp_iを確保しその実行を管理する．つまり処理体のインスタンスの中にメッセージを処理するルーチンのインスタンスがつくり出されるという二重構造になっている．

処理体において状態値の更新がなく履歴依存処理ではない場合は，ループのunfoldingによって到着する各メッセージm_iにループインスタンスp_iが与えられ処理ルーチンは並列に活性化される．この場合インスタンス間には依存関係がないので各処理は完全に非同期に行われる．履歴依存のある処理体の場合もメッセージm_i毎に新たなループインスタン

Implementation of Message-flow Agent Based on Data-flow Mechanism
Shigeru KUSAKABE, Takashi TOMOKIYO, Rin-ichiro TANIGUCHI and Makoto AMAMIYA
Graduate School of Engineering Sciences, Kyushu University

ス p_i が与えられ処理ルーチンは並列に活性化される．しかしこの場合はループインスタンス間には状態値に関して依存関係がある．そのためインスタンス間で状態値を適切にリンクする必要がある (図 1)．このように 1 つ前のメッセージの処理による状態に依存している部分はその状態値の決定まで処理は中断されるが，メッセージ毎にインスタンスを確保しておけば，その状態値に依存していない部分は先に処理を進めておくことができる．

4 実行方式

以上のような計算メカニズムによって処理されるメッセージフローシステムを，現在我々の研究室で研究を進めている Datarol アーキテクチャ[4]上で実現するために次のようなことを検討した．

処理系では履歴依存の処理体を非同期ループとして実現する．処理体の再帰処理を再帰式として扱い，履歴依存の処理体の場合に必要な状態値の更新処理を以下のように処理する．処理系では異なる状態値に対し再帰的な処理を行う処理インスタンスをそれぞれ用意し，更新される状態値をそれらのインスタンス間で適切にリンクすることで履歴依存処理を実現する．Datarol プログラムでは nins 命令によって新しいインスタンスを確保し slink 命令によってインスタンス間の引き数のリンクを行う[3]．履歴依存処理体の実行の概略を図 2 に示す．

また我々のシステムではメッセージの送出は明示的に記述するが，メッセージの受理は処理系において自動的になされる．メッセージの受理においては先行 cons を用いてメッセージキューをつくることにより，メッセージストリームの部分的評価を可能にする．処理体の消去は，処理体に消去メッセージを送ることにより行われる．処理体では消去メッセージを受理するとメッセージキューに終端の記号を入れる．処理体では再帰的な繰り返し毎にメッセージをキューから取り出す．メッセージキューの最後には消去メッセージを受け取った際の終端の記号が入れられているので取り出した要素が終端の記号であれば処理体を消去の処理を行う．

履歴に依存しない処理体の場合，処理結果はタイミングに関係なくメッセージの内容だけに依存しているため，メッセージをキューに並べてその処理順序を管理する必要はない．これは実行結果が引数の値だけに依存する関数呼び出しと類似している．この点から，履歴に依存しない処理体のメッセージによる呼び出しを関数呼び出しと同様の処理で行うことで関数呼び出し程度の負荷に抑えるようにする．

5 まとめ

本稿では我々が実現を目指しているデータフロー方式にもとづく並列協調処理システムの直感的計算メカニズムを示し，データフローメカニズムのもとで再帰とストリームの概念によりそのようなシステムを実現する方式について述べた．

今後はシステムのより厳密なモデル化を行いその動作を検証する．また，問題のより自然な記述のためさらに言語仕様の洗いだしを行う．そして我々の研究室で研究を進めている Datarol アーキテクチャ上での実現を目指す．

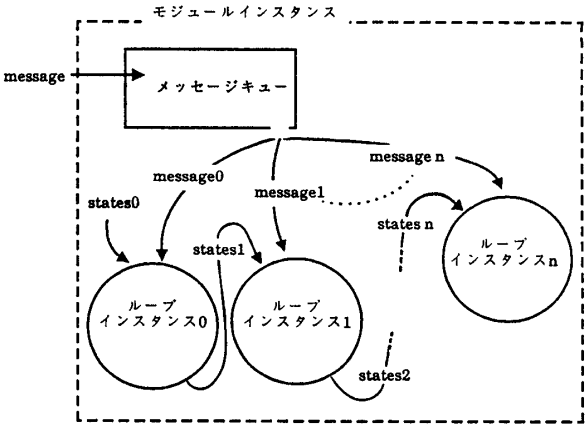

図 1:履歴依存処理体

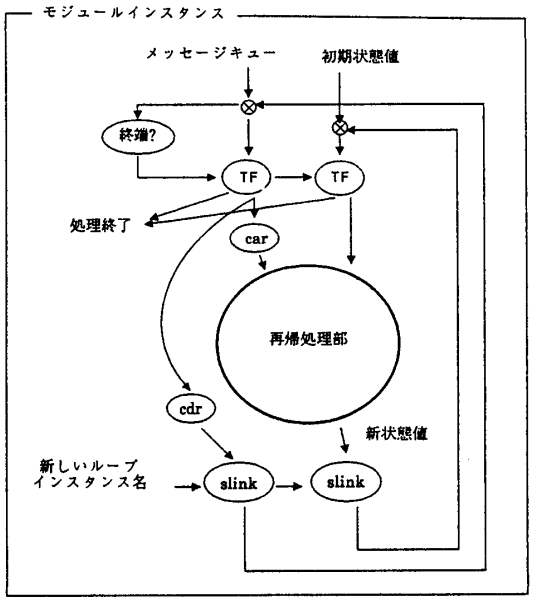

図 2:データフローでの履歴依存再帰処理

参考文献

[1] R. E. Filman and D. P. Friedman. “Coordinated Computing”, MIT Press(1984)

[2] Gul A. Agha.“Actors: A Model of Concurrent Computation in Distributed Systems”,MIT Press(1986)

[3] 立花, 谷口, 雨宮. “データフロー解析による 関数型言語 Valid のコンパイル法”, 情報処理学会誌 Vol.30, No.12, p.p.1628–1638 (1989)

[4] 雨宮.“超並列多重処理のためのプロセッサアーキテクチャ”,「コンピュータアーキテクチャ」シンポジウム論文集,p.p.99–108(1988)

[5] 雨宮, 長谷川.“並列協調システムにおけるメッセージ処理機構”, ソフトウェア科学会第 4 回大会論文集,p.p.315–318(1987)

[6] 友清, 日下部, 谷口, 雨宮.“データフローに基づく並列オブジェクト指向言語による分散協調処理”, ソフトウェア科学会第 7 回大会論文集,p.p.61–64(1990)