マルチメディア通信と分散処理　50－19
（１９９１．５．１７）

分散協調型問題解決における分散協調プロトコル

中森勝　平田謙　井内稔　山崎晴明
山梨大学

分散協調型問題解決とは、知識や推論機構などの問題解決能力と、通信機能を持つ複数のコンピュータ（これを以下エージェントと呼ぶ）を用いて、一つの問題を協力して解決するアプローチである。本稿では、まず協調とはどのようなものなのかを分析し、その実現方法を検討する。また、検討した協調方法を実現する分散協調型問題解決システムにおいて、エージェント間の通信を支援するプロトコルを定義し、その役割、機能について考察する。

PROTOCOL FOR COOPERATIVE DISTRIBUTED PROBLEM SOLVING

Masaru Nakamori　Ken Hirata　Minoru Iuchi　Haruaki Yamazaki

Yamanashi University

4-3-11 Takeda, Kofu, Yamanashi 400, Japan

In Cooperative distributed problem solving systems, a group of agents, each of which has knowledge and reasoning mechanism solves a problem by cooperation.

Here we analyze what cooperation is, and investigate the scheme to implement the cooperation. Furthermore, we define the protocol needed for cooperation, and discuss its role and neccesary functions on problem solving.

１．はじめに

近年、人工知能の分野では問題解決に分散処理を取り入れた研究が行われるようになった。従来の問題解決システムでは単一のエージェントを用いたものが一般的である。しかし、信頼性の向上、付加の軽減、複雑な問題への対応などの面では、複数のエージェントにより構成される問題解決システムが有効である。今回我々は知識表現モデルの異なる複数のエージェントから成る分散協調型問題解決システムの試作を行った。

本稿では、複数エージェントによる分散協調のための「分散協調プロトコル」について、その役割と必要な機能を概説する。

２．協調とは

一般の分散型ＡＩと分散協調型問題解決システムには、決定的に異なる点がある。それは、前者ではエージェントが「自律的」、つまりそれ自身の解の生成において他のエージェントからの影響を受けないのに対して、後者ではエージェントが「協調的」、つまりそれ自身の解の生成過程ににおいて他のエージェントから影響を受けることである。ここでは、エージェント間で互いに影響を与え合うことを、相互作用と呼ぶ。

「協調」の持つ特徴としては、

・相互作用が動的である
・相互作用が問題解決のために効果的に利用される

が挙げられる。

一般的な分散システムではエージェント間の相互作用があらかじめ定義されているのに対して、分散協調型システムではその相互作用が処理の過程で動的に決定される。前者では、エージェントは互いに協力するための一般的な知識や全体問題に対する認識を持っていないので、あらかじめ決められた方法以外では協力を行うことができない。一方、後者では、エージェントは自らの持つ知識だけで部分的な解をまとめることができ、また全体問題への認識を持ち、協力するための一般的な知識に従って協力する方法を動的に開発することができる。

３．試作した分散協調型問題解決システム

3.1.．システムの構成

試作した分散協調型問題解決システムは、３つのエージェントと通信を集中的に管理するプロトコル・マネージャから構成されている。

３つのエージェントは、プロダクション・システム、フレーム・システム、意味ネットワーク・システムであり、それぞれ異なる知識表現形式を用い、異なる推論機構を所有している。

プロダクション・システムは「もし～ならば…」という形式のプロダクション・ルールを知識表現モデルとして用い、ワーキング・メモリに対してプロダクション・ルールを適用することにより推論を行う。

フレーム・システムは、複数のスロットから成るフレームという知識表現を用い、性質の継承・付加手続きを使用し、スロット値を求めることにより推論を行う。

意味ネットワーク・システムは、ノードと、ノード間のアークにより形成される意味ネットワークという知識表現を用い、活性化伝播を用いアークを辿ることにより推論を行う。

プロトコル・マネージャはエージェント間の通信を集中管理する機能を持つ。集中管理者を設けることは分散協調の主旨に反するが、システムをパーソナルコンピュータで実現するため、その処理能力を考慮した結果、完全に分散したシステムを構築することは困難であると判断した。プロトコル・マネージャはエージェント間の通信を管理するだけであり、システムの管理は行わない。

3.2. システムの協調方法

分散協調型問題解決システムでの最大の問題点は、エージェント間に生じた矛盾をどのように扱うべきかということである。それには以下の三つの方法が考えられ、それぞれ既存の代表的な協調技法を例として挙げておく。

1)矛盾の存在を認めない。矛盾と認識しない。

例．マルチエージェント・プランニング

2)話し合いにより矛盾を解決する。

例．ネゴシエーション

3)矛盾が存在しても実行を継続するシステムを構築する。

例．Functionally-accurate Cooperation

試作した分散協調型問題解決システムにおいては、ネゴシエーションとFunctionally-accurate Cooperation の両方を取り入れている。解に多様性を持たせるために推論の重複を極力排除する。そのため、エージェントが部分解を求める過程でもネゴシエーションにより、推論の中間結果の交換を行う。総合的な解を求める段階では、Functionally-accurate Cooperation に似た方法により部分解の交換を行う。

我々のシステムではこの協調を以下のような方法で実現している。

エージェントが他のエージェントに送る推論の中間状態の形式には、以下の４つがある。

1)提案　：推論の継続を依頼する

2)質問　：中間結果の評価を依頼する

3)応答　：質問の返事を返す

4)再評価：解をまとめる際に、解として適当か全エージェントに対し評価を依頼する。

他のエージェントから情報を受け取る場合は、原則として評価値の高い中間結果を優先的に受け取る。この機能はプロトコル・マネージャを介して行われ、システム全体として推論がより評価の高い中間結果に焦点をあてた方向に向かうことになる。

試作したシステムでは、全体問題を複数の段階（我々はこれをステージと定義している）に分割して解を導出する。各ステージごとに、エージェントが中間解の再評価を行い、総合的な評価値を決定する。一定数の中間解の再評価が終了したとき、総合的な評価値の高いものを次のステージの初期問題として選択する。

今回のシステムはアプリケーションに探索問題を想定したため、推論の進行状況を明確に表現する論理的な構造として木構造を採用した。(図1)

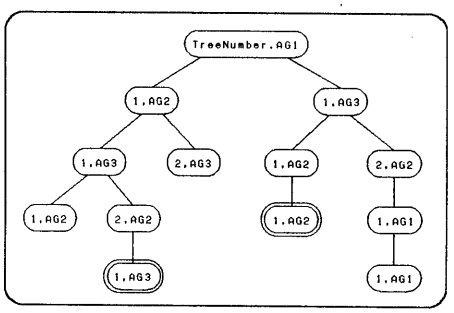


図1 木構造

推論の中間状態は木のノードとして保持され、子ノードは、その親ノードから推論が一歩進んだ状態を表す。従って、推論が進行するにつれて木は深くなる。

４．分散協調プロトコル

今回試作したシステムにおけるプロトコルを分散協調プロトコルと呼ぶ。分散協調プロトコルは、下位レベルプロトコルと上位レベルプロトコルに分けられる。下位レベルプロトコルは、ＩＳＯ７層モデルの物理層からセッション層までに対応し、エージェントとプロトコルマネージャとの間の信頼性の高い通信を保証する。上位レベルプロトコルは、同モデルのアプリケーション層に該当し、エージェント間の協調動作を支援する。(図2)

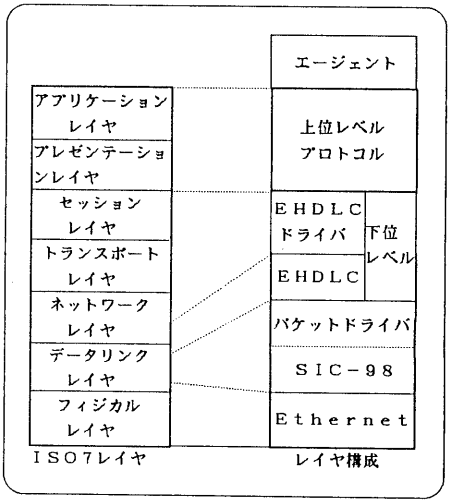


図2 分散協調プロトコルの階層構造

4.1.下位レベルプロトコル

エージェントとプロトコルマネージャの間の信頼性のあるデータ通信機能を提供するのが下位レベルプロトコルの目的である。

物理層には、Allied Telesis社製の CentreCOM SIC-98 10BASE5/10BASE2を使用する。データリンク層には、同製品に添付されているパケットドライバ（FTP Softwear Inc. 製）を使用する。また、このパケットドライバには、パケットの送達確認、エラー時のパケットの再送、および受信したパケットのエラー処理などを行わないため、ＨＤＬＣ（High level Data Link Control procedure）とと呼ばれるプロトコルを拡張したＥＨＤＬＣを定義し、上位レベルプロトコルとのインタフェースの機能を持つＥＨＤＬＣドライバを作成した。

ＨＤＬＣは、ＯＳＩ参照モデルのデータリンク層にあたる部分の標準方式として、もっとも一般的な方法である。ＨＤＬＣは、以下のような特徴を持つ。

1)任意のビットパターンの伝送が可能である。
2)受信側からの応答を待たずに連続してデータを送信できる。
3)誤り検出が厳密である。
4)コンピュータ間の通信に最適である。

Ｅｔｈｅｒｎｅｔ上のデータ伝送はパケットを単位として行われるが、ＨＤＬＣにはこの機能が存在しないため、ＨＤＬＣを拡張し、パケットを単位としたデータ伝送機能を行うよう拡張を施してある。また、フラグメントに関する処理についても同時に拡張を行った。

4.1.1 ＥＨＤＬＣ

ここではＥＨＤＬＣについて説明する。図3 にＥＨＤＬＣのフレーム構成を示す。ＥＨＤＬＣのフレームはイーサネットのフレームに対応する。ＥＨＬＤＣのフレームは、それぞれイーサネットとは独立なヘッダを持つ。以下にＥＨＤＬＣのフレームの各要素について述べる。

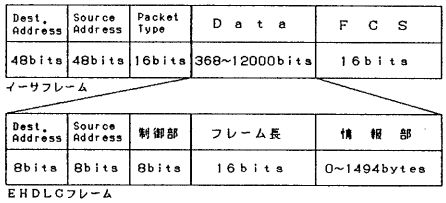


図3　ＥＨＤＬＣのフレーム構造

(1) アドレス部

アドレス部には、デスティネーションアドレスとソースアドレスがある。デスティネーションアドレスは宛先のＥＨＤＬＣアドレスを示し、ソースアドレスは送信元のＥＨＤＬＣアドレスを示す。

(2) 制御部

制御部は、コマンドとしては相手局に対する動作の指令に、またレスポンスとしてはその指令に対する応答に使用する。コントロール部の形式には、情報転送、監視及び非番号制の３つがある。

情報転送形式　情報転送形式のフレームは、情報部を持ち、順序番号を用いて受信確認を行う場合の情報の転送に使用する。すべての情報転送形式フレームは、送信順序番号（以下Ｎ（Ｓ）と記述）を持ち、送信の確認に使用する。ＦＲビットは、データのフラグメント化の制御を行う。以下、情報転送形式フレームをＩフレームと記述する。

監視形式　監視形式のフレームは、受信の確認や再送の要求などを行う場合に使用し、情報部を持たない。監視機能ビットは監視機能の指定に用いる。以下、監視形式フレームをＳフレームと記述する。

番号制形式　非番号制形式のフレームは、モード設定などの制御機能に使用する。非番号制形式フレームは制御機能の種類により、情報

部を持つものと持たないものがある。修飾機能ビットは制御機能の指示に用いる。

(3) フレーム長

フレーム長はオクテット単位でフレームの情報部の長さを表す。フレーム長は、最大転送フレーム長（ＭＴＵ）の制約により 0から1495の範囲の数値でなければならない。

(4) 情報部

情報部もフレーム長と同様にＭＴＵの制約から、1495オクテット以下でなければならない。

4.1.2 ＥＨＤＬＣの伝送手順

ＥＨＤＬＣはＨＤＬＣにおける非同期平衡モードで動作する。ＨＤＬＣにおいてこのモードは、複合局に対する設定である。このモードにある局は相手複合局の許可無しに１つ以上のレスポンスフレームを送出することができる。

ＥＨＤＬＣにおけるデータリンクの設定はＳＡＢＭフレームを送出することで行う。ＳＡＢＭフレームを受信した局は接続を行う場合、ＵＡフレームを返すことでデータリンクを設定する。

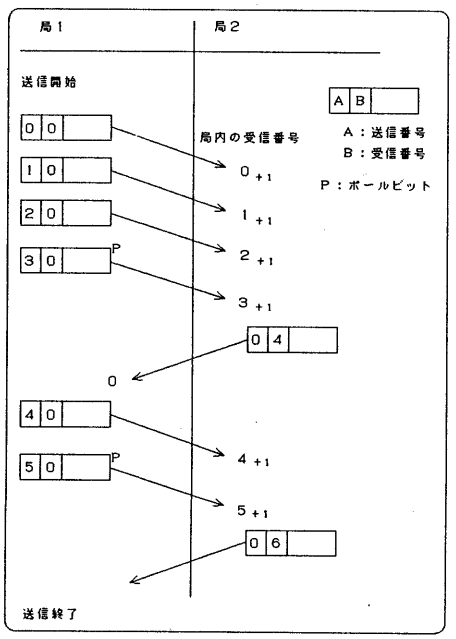


図4 ＥＨＤＬＣの通信動作

ＥＨＤＬＣにおいてデータ伝送はＩフレームで行う。また、各局は局内に８のモジュロである送信番号と受信番号を持つ。Ｉフレームの送出を行う場合、Ｉフレーム中に局内に持つ送信順序番号を挿入送出し、局内の送信順序番号をカウントアップする。Ｉフレームを受信した局はＩフレーム内の送信順序番号と局内の受信順序番号を比較し等しければ受信順序番号のカウントアップを行う。また、Ｓフレームの送出を行う場合には、局内に持つ受信順序番号を挿入する。基本的なデータ通信動作を図4 に示す。

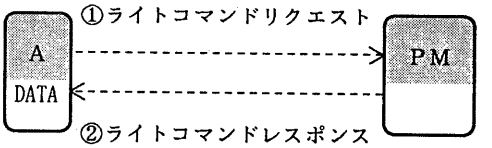
4.2. 上位レベルプロトコル

上位レベルプロトコルは、下位レベルプロトコルから提供されるさまざまな機能を用いてエージェント間の協調を支援する。

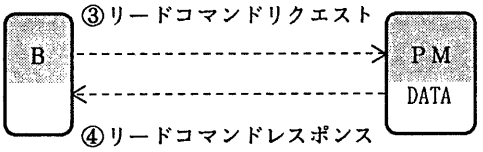
上位レベルプロトコルの実行部は上下層の二つの層に分割されている。下層はエージェントとプロトコル・マネージャの間の通信を制御するもので、実際にシステムでおこなわれる通信に即している。上層はエージェント間の通信を制御するものであり、現状ではプロトコル・マネージャを介しておこなわれている。

エージェントがプロトコル・マネージャにコマンドを送出すると、プロトコル・マネージャがコマンドに関する処理をおこなう。それが終了するとなんらかのレスポンスをエージェントに返し、一連の通信処理が終了する。

コマンドには、他のエージェントからの情報を受け取る「リード」と、他のエージェントに推論結果を送る「ライト」の二つがある。エージェント間の通信は必ずプロトコル・マネージャを介しておこなわれるので、エージェントＡからエージェントＢに情報を送る場合の動作は、次のようになる。



① エージェントAがプロトコル・マネージャに対し、ライト・コマンド・リクエストを送信し、情報を書き込む。
② プロトコル・マネージャよりステータス情報を受け取り、正常終了を確認する



③ エージェントBがプロトコル・マネージャに対し、リード・コマンド・リクエストを送信する。
④ エージェントB宛ての情報がある場合は、プロトコル・マネージャから情報とステータス情報が送信され、ない場合にはステータス情報のみが送信される。

4.2.1 下層

エージェントとプロトコル・マネージャの間の通信に必要な情報は次のようになる。

・エージェント・ナンバー

その中間結果を送出したエージェントの識別子。プロトコル・マネージャが、どのエージェントにコマンドの結果を返すのかを判断するために使用する。

・パケット・ナンバー

エージェントが送出した、何番目のパケットであるかを表わす。エージェントが、送出したコマンドの結果を受け取るときに、コマンドの結果が受信バッファ中のパケットのどのパケットに含まれているかを判断するために使用する。

・リード／ライト・フラグ

そのパケットの内容がリード・コマンドなのか、ライト・コマンドなのかを表わす。上位レベル・プロトコルの上層が使用する。

4.2.2 上層

次に、エージェント間の仮想的な通信に必要な情報を示す。リードとライト、リクエストとレスポンスによって必要なものに違いがあるので、それぞれを識別するために以下の記号を用いる。

RQ：Read command reQuest
RS：Read command reSponse
WQ：Write command reQuest
WS：Write command reSponse

・存在フラグ（RS，WS）

エージェントが受け取るべき情報があるときにセットされる（RS）。また、ライトしようとした情報が既に存在するときにセットされる（WS）。

・PQAフラグ（RS，WQ）

提案（Proposal）、質問（Question）、応答（Answer）を識別する。

・再評価要求フラグ（RS，WQ）

中間結果が再評価すべきものであるときセットされる。

・受け取り先指定フラグ（RS，WQ）

中間結果を受け取るべきエージェントに対応する場所がセットされる。

・ステージ・ナンバー（RS，WQ）

中間結果がどのステージに属するのかを表わす。

・評価値（RS，WQ，WS）

中間結果の評価値。

・木における位置（RQ，RS，WQ，WS）

推論の進行状況を表わす木における、中間結果の位置。

・推論情報（ＲＳ，ＷＱ）

中間結果の推論情報を共通表現で表わしたもの。

プロトコル・マネージャの動作で説明したように、エージェントが送出したコマンドは、プロトコル・マネージャを介して処理され、エージェントがレスポンスを受け取ることで、その動作を終了する。つまり、エージェントはコマンドを送出した後、レスポンスが返されるまで待つ必要がある。これは、必然的にエージェント間の通信が、同期的におこなわれなければならないことを意味する。そのため、下位レベル・プロトコルが非同期通信をおこなっているにもかかわらず、上位レベル・プロトコルでは同期通信しかできないことになる。

4.3. 上位下位間のインターフェイス

下位レベル・プロトコルは、エージェントとプロトコル・マネージャとの円滑な通信を支援するために、以下の機能を上位レベル・プロトコルに提供している。

・初期化

下位レベル・プロトコルの初期化.
上位レベルが用いるエージェントのアドレス設定.

・終了

下位レベル・プロトコルの機能停止.
エージェント単体動作環境に復帰.

・接続

指定アドレスのエージェントと仮想回線を接続.

・状態取得

指定アドレスのエージェントの状態を取得.

・データ送信

指定アドレスのエージェントにデータを送信.

・ビジー状態設定

全エージェントにプロトコル・マネージャがビジー状態であることを告知.

・ビジー状態解除

全エージェントにプロトコル・マネージャのビジー状態解除を告知.

下位レベル・プロトコルは非同期通信を行い、上位レベル・プロトコルは同期通信を行うため、上述した機能には、データ受信の機能が含まれていない。

エージェントが非同期受信を行う場合には、現在の推論状態の保存および復元を行わなければならず、推論以上に負荷が大きいと考えたため、このような形態をとった。

実際のデータ受信は次のように行われている。

上位レベル・プロトコルの通信用バッファは、リング・バッファになっており、その先頭位置と末尾位置は、上位レベルと下位レベルの両方が参照可能である。下位レベルの処理機構は、非同期に他のエージェントからのデータを受け取ると、ただちにリング・バッファの末尾位置にデータを転送し、末尾位置を一つ先に進める。上位レベルの処理機構は、アプリケーションによって明示的に示されている処理の区切りごとに呼ばれると、リング・バッファの先頭位置の内容をアプリケーションが利用するバッファに転送し、先頭位置を一つ先に進める。

以上のような動作で、上位レベルの処理機構は同期的なデータの受信をおこなう。

5．問題点

今回のシステムの問題点として挙げられることは、共通表現形式を、アプリケーションに依存する部分と汎用的な協調動作に関連する部分とに、

明確に分割することができなかったことである。そのため、今回の分散協調プロトコルは、探索指向型の問題解決しか対応できない。異なったタイプの問題を扱う場合には、どのような機能や情報が必要であるかをはじめから検討し直す必要がある。

また、現段階では、エージェント間で交換し合っている情報には推論の状態に関する情報しか含まれていない点である。従って、エージェントは、現在行っている推論が全体問題のどの部分にあたるのかを判断できない。

以上の問題点を検討し、改良することが今後の課題である。さらに、現在独立しているプロトコル・マネージャを各エージェントに組み込むことにより、本来の意味である分散協調を実現することも必要である。

参考文献

[1] E.H. durfee, V.R.Lesser, D.D.Corkill : "Trends in Cooperative Distributed Problem Solving", IEEE transaction on knowledge and data engineering, Vol1, No1 (1989)

[2] Gregor von Bochmann : "Concepts for Distributed Systems Design", (1983)

[3] プロトコル・ハンドブック，朝日新聞社 (1985)

[4] PC/TCP Version 1.08 Packet Driver Specification:FTP Software Inc.

[5] Douglas Comer原著、村井純一・楠本博之訳:「ＴＣＰ／ＩＰによるネットワーク構築」、共立出版(1990).

[6] DEC, Intel, Xerox:"The Ethernet, A local Area Network Data Link and Physical Specifications, Version 2.0"(1982).

[7] ハイレベルデータリンク制御手順のフレーム構成:JIS X 5104-1987

[8] ハイレベルデータリンク制御手順の手順要素:JIS X 5104-1987

[9] ハイレベルデータリンク制御手順の手順クラス:JIS X 5104-1987

[10] Gregor von Bochmann原著、水野忠則・井手口哲夫訳:「分散処理システムデザイン」、工学社(1987).