5P－1

ルールコンパイラSaHowにおけるエージェント間協調プロトコルの表現について

水口卓也　新谷虎松
名古屋工業大学知能情報システム学科
E-mail : {mizuguti,tora}@ics.nitech.ac.jp

1．はじめに

マルチエージェントシステムにおいて，個々のエージェントのふるまいを，個々のゴールと共有のゴールを達成するために協調させることは重要な問題である[1]．本研究では，エージェント間で起こるインタラクションプロセスについての知識を明確に表現することによって個々のエージェントを協調させる．本研究では，エージェント間で起こる構造化されたメッセージの送受信について協調プロトコルを明記することによって，インタラクションプロセスについての知識を明確に表現した．本研究では，プロトコルの表現と実行にルールベースシステムを用いた．本論文では，本研究で実装したSaHowシステムで，どのようにルールを用いてエージェント間協調プロトコルを表現するか，また，本研究で構築したエージェントの構成について述べる．

2．SaHowシステムの概要

本研究では，エージェント間の協調プロトコルを記述するために，SaHow(作法)システムを構築した．SaHowシステムを用いてエージェントに協調作業を行わせるまでの流れを図１に示す．複数のエージェントがネットワーク上に存在している．これらのエージェントはJava言語によって実装されている．SaHowシステムでは，これらのエージェントが協調するためのプロトコルをルールとして表現する．プロトコルを表現したルールをルールコンパイルしてJava言語のプログラムに変換する．変換されたJava言語のプログラムは，ルール配送エージェントによって，ネットワーク上のエージェントに配送される．エージェントは，ルールの読み込みを完了すると，そのルールによって表現された協調プロトコルに従って協調作業を行う．SaHowシステムは，プロトコルを表現したルールからJava言語のプログラムを生成し，エージェントに配送するシステムである．

3．SaHowシステムにおけるエージェント

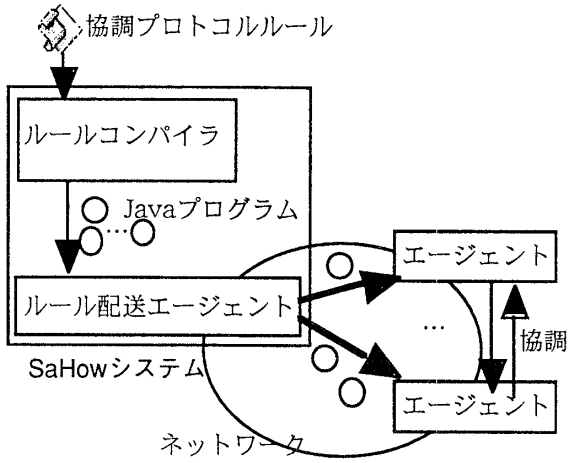

図１　SaHowシステムの流れ

SaHowシステムでは，すべてのエージェントのひな型となるAgentSkeltonが提供されている．Agent Skeltonは，Java言語のプログラムとして記述されている．エージェントは，複数のタスクを実行する必要があるので，それぞれのタスクに用意された複数の協調プロトコルを管理できなければならない．AgentSkeltonには，協調プロトコルを表現したルール集合を複数格納し，必要に応じて適切なルール集合を呼び出して認識-行動サイクルを行うための機能，ルール集合を用いて推論を行うためのルールインタプリタ，他のエージェントとメッセージを送受信するための機能，が用意されている．エージェントは，AgentSkeltonを継承し，それぞれの応用領域に特化した機能を追加することによって構築される．ここで重要なのは，エージェントを構築する時は，エージェントそれぞれの応用領域に特化した機能だけを実装すればよいという点である．

メッセージを受信したエージェントは，適切なルール集合のワーキングメモリに受信したメッセージを追加する．これによって適切なルール集合が呼び出され，ルールインタプリタが推論を行う．受信したメッセージと，ルール集合の状態遷移ルールの条件部との照合が成功した時，エージェントは外部情報源に対するアクションを行ったり，他のエージェントにメッセージを送信したり，他のルール集合を呼び出して推論を続ける．他のルール集合を呼び出すことは，エージェントが現行の協調作業を中断し，他の協調作業を行うことを意味している．

Representation of Cooperation protocols on Rule compiler SaHow,Takuya Mizuguchi, Toramatsu Shintani,Dept. of Intelligence and Computer Science,Nagoya Institute of Technology, Gokiso-cho,Showa-ku,Nagoya 466-8555,Japan

```
(1) agentInfo(name=MailAgent)
(2) rulesetname(name = PRegist)

    literalize(state(String statename))
(3) literalize(message(String sender,String type,String data))
    literalize(agent(String name))

(4) rule( datasend(
      if(
        state(statename=state2)
(5)     message(type=Mail,sender=?sender,data=?data)
        mail(maildata=?mail)
        agent(name=?name))
      then(
        sendMessage(receiver = ?sendername,
                    content = make(
                                message(sender=?name,
(6)                                     type= ?data,
                                        data= mail))))
                    remove(2))))
```

図２　ルールの記述例

４．協調プロトコルのルール表現

エージェント間の協調作業は，エージェントが受け取ったメッセージによって，状態から状態へ遷移をくり返すことによって行われる．状態遷移を行う時に，エージェントはメッセージを送信したり，特定の処理を実行する．SaHowシステムにおける協調プロトコルのルール表現では，エージェント間で交換するメッセージの形式の定義は literalizeの宣言に，協調プロトコルの状態遷移ルールの定義はルールの宣言に，それぞれ帰着される．協調プロトコルの状態遷移表現における現行の状態はワーキングメモリ要素として表現される．協調作業は，状態遷移ルールの条件部と，現行の状態や受信したメッセージとの照合が行われ，照合が成功した時に次の状態へと遷移することで行われる．

SaHowシステムにおけるルールの記述例の一部を図２に示す．図２の(1)は，記述した協調プロトコルが，nameスロットで指定した名前を持つエージェントによって使用されることを指定するものである．図２の(2)は，ルール集合の名前を指定するために用いられる．ルール集合の名前は，name スロットで指定する．ルール集合はこの名前でラベル付けされ，エージェントごとに管理される．図２の(3)は，literalize宣言である．協調プロトコルを表現する時には，literalize宣言は，現在の会話の状態を表すワーキングメモリ要素(WME)クラス，エージェント間で送受信されるメッセージを表すWMEクラスなどを宣言するのに用いられる．図２の(4)は，ルールの宣言である．図２の(5)はルールの条件要素，図２の(6)はRHS要素である．条件要素は，協調プロトコルを表現する時には，現在の会話の状態，受信したメッセージなどと照合を行うために用いられる．RHS要素は，協調プロトコルを表現する時には，現在の会話の状態の遷移，メッセージの送信，エージェントの応用領域に特化したアクションを行うために用いられる．

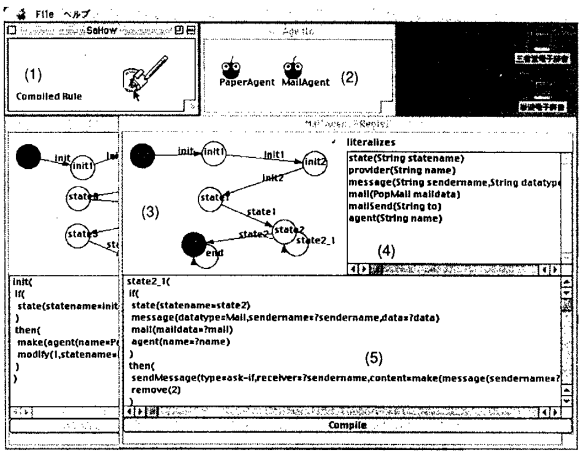

図３　SaHowシステムの実行図

図２のように記述されたルールはルールコンパイラ[2]によってJava言語のプログラムに変換される．この時に，ルールからLHSフィルタ[2]が生成される．LHSフィルタは，推論を高速に行うための照合フィルタの一種である．LHSフィルタはルール配送エージェントによって個々のエージェントに配送される．ルールを直接配送するのではなく，LHSフィルタを送信することによって高速なルールの処理を行うことができる．

SaHowシステムの実行例を図３に示す．図３の(1)はSaHowシステムの状態を示す窓である．図３の(2)はネットワーク上に存在するエージェントを示す窓である．図３の(3)は協調プロトコルの状態遷移グラフ，図３の(4)はliteralizeのリスト，図３の(5)は状態遷移ルールを示している．SaHowシステムでは，図３のようなインタフェースを用いて協調プロトコルを記述し，ルールコンパイルを行い，個々のエージェントにLHSフィルタを送信することができる．

５．おわりに

マルチエージェントシステムにおいて，個々のエージェントのふるまいを，個々のゴールと共有のゴールを達成するために協調させることは重要な問題である．本研究では，SaHowシステムを構築し，エージェント間の協調プロトコルを明記することで，この問題を解決した．本研究では，協調プロトコルを表現するのにルールを用いた．

参考文献

[1] Mihai Barbuceanu and Mark S. Fox,"The Design of a Coordination Language for Multi-Agent System", ECAI '96 Workshop (ATAL), Budapest,Hungary, August 12-13,1996.

[2] 水口卓也，新谷虎松，“Java言語に基づく照合フィルタのアプレット化について，”第54回情処全国,vol.2,pp. 2-141, 1996.