1-48

マルチエージェントシステムのための並列分散プラットホームの構築

日大生産工(院) ○ 藤代 孝男　　日大生産工　　西 恭一
日大生産工　　村田 守　　日大生産工　　星野 和義

1. 緒言

近年，ソフトウェアの設計・製作におけるオブジェクト指向に次ぐ新たなパラダイムとして，また解析・シミュレーションの手法としてもマルチエージェントが注目されており，その有用性も明らかにされている[1]．しかし，マルチエージェントを用いたシステムは，複雑な処理の必要性から大量のリソースを消費し，計算量も膨大となることから，結果を得るまでの計算時間が長くなる欠点を有する．このため，マルチエージェントシステムを大規模なシミュレーションへ適用するためには，マルチエージェントの特性に適した並列・分散処理環境が必須となる．以前は大規模な解析やシミュレーションに用いることができるライブラリが存在しなかったため，当研究室では独自の並列・分散マルチエージェントプラットフォーム(以後dmap)[2]を構築してきたが，dmapは大規模クラスタシステムにおいて，デッドロック現象によって途中で計算が止まってしまい，シミュレーション結果が得られない場合があることがわかっている．

そこで本研究ではdmapに代わり，容易にエージェントアプリケーションを作成でき，しかも，デッドロックが絶対に起こらない新たなマルチエージェントライブラリであるAgent Frame Work(以後 afw)の開発を目的とする．

2. afw(Agent FrameWork)

2.1. afw概要

afwはマシンやOSに非依存なシステムとするため，Sun Microsystems社が開発したオブジェクト指向言語であるJava言語を用いて構築している．また，afwはFIPA準拠のエージェントプラットフォームとしてデファクト・スタンダードであるJADE(Java Agent DEvelopment Framework)[3]をベースに作られているため， afwの起動の際にはJADEの起動が必須となっている．

2.2. JADEシステム概要

エージェントの実行環境をコンテナと呼び，複数のコンテナを格納する場をプラットフォームと呼ぶ．また，各プラットフォームには必ず一つのメインコンテナがあり，エージェントの管理が可能なAgent Management System(AMS)エージェントとエージェントの検索が可能なDirectory Facilitator(DF)エージェントが存在する（Fig.1).

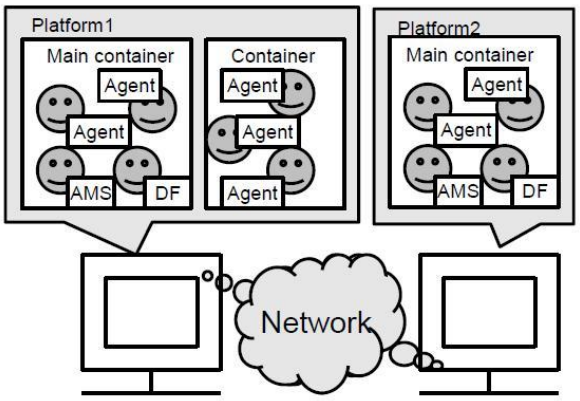

Fig.1 Containers and Platforms of JADE

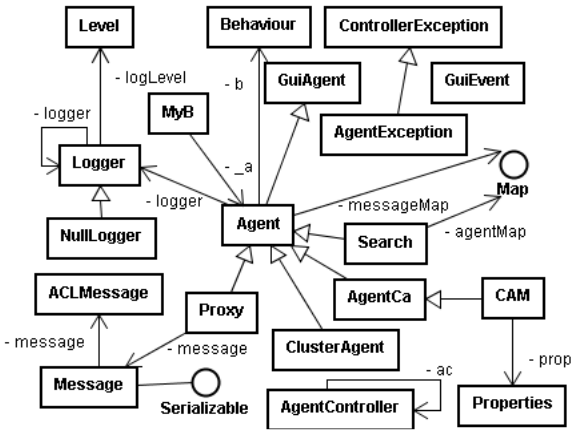

Fig.2 Class chart

2.3. afwのクラス

afwの各クラスの関係を示したクラス図をFig.2に示す．

afwの最もコアとなるクラスはAgentクラスであり，Agentクラスはエージェントの構築に必要な基本機能を提供しているため，Agentクラスを拡張し各計算式などのルールを追加することで，様々な計算を行うエージェントを容易に作成できる．なお，AgentクラスはJADEのjade.core.Agentクラスを拡張したjade.gui.GuiAgentクラスをさらに拡張したクラスである．

Development of Parallel and Distributed Multiagent Platform

Takao FUJISHIRO, Yasukazu NISHI, Mamoru MURATA and Kazuyoshi HOSHINO

3.　afwの特徴および性能評価

3.1 afw の特徴

afwはエージェント指向アプリケーションの構築しやすさを重視してJADEを拡張しており，その特徴を以下に挙げる．

- JADE ではエージェント作成の際，初期化のため setup メソッドから書き始める必要があるが afw はエージェントの行動から書き始めることができる．
- JADE はエージェント識別子を取得する必要があるが，afw は気にする必要がない．
- afw はメッセージ送受信の記述が容易である．
- プラットフォーム間のエージェント移動の際に，JADE は移動先のマシンを詳しく記述する必要があるが，afw はマシン名を指定するだけで移動できる．
- 自動負荷分散機能を有している(後述 3.2 節)．

3.2 自動負荷分散機能

afwは自動的にクラスタシステムの各マシンの負荷を均一化することでシステム全体としての遊びを減少させ，リソースを効率良く活用する自動負荷分散機能(CAMエージェント)を実装している．

まず，CAMエージェントはpropertiesファイルから使用マシンを取得し，各マシンに負荷監視エージェントであるClusterAgentエージェントを送る．ClusterAgentエージェントは所属するマシンのCPU使用率およびメモリ使用率を取得し，メッセージとしてCAMエージェントへ負荷状態を送信する．CAMエージェントは受信メッセージを対応マシン情報と共にAgentCaとして定義した別クラス内に格納し，システムの停止までCAMエージェントとClusterAgentエージェントによるメッセージのやり取りが繰り返えされる．そのためAgentCaクラスは常に最新の全マシン負荷状態の把握ができており，計算中に各マシン間で一定以上の負荷差が生じる場合のみ，新規生成エージェントの配置対象を負荷評価値の低いマシンとすることで負荷分散を行う．

3.3 JADE との通信速度比較

本システムのマルチエージェントライブラリとしての性能評価のため，JADEとの通信速度比較を行う．なお，マシン環境はOS:Windows Vista Business，CPU:AMD Athlon(tm)64 3200++ 2GHz，Memory:3.5GBを用い，評価方法は２体のエージェントが5Byteの文字列をメッセージとして互いに送信しあう際の所要時間とする．Fig.3に示すようにafwはJADEに比べ通信時間が長くなる結果が得られるが，afwはプログラムのライン数がJADEより15%程度短くなる有意性を持つ．

3.4 並列環境における速度向上率

大規模データを扱うマルチエージェントシステム例としてFig.4に示すようなCT画像(567×657pixelsの人間顎部断面画像71枚)から骨部を自動検出しFEM解析ソフトに読み込めるファイルにコンバートするアプリケーションの実装を行う．また，本システムの並列環境における性能評価としてネットワーク通信を考慮したアプリケーションを作成し，上記のCT画像を10回取り込むことで複数マシンにおける速度向上率の測定を行う．なおマシン環境はOS:Red Hat Linux9, CPU:Pentium3 1.4GHz×2, Memory:512MBを用い，エージェント数を10とし，各エージェントにはCT画像を71枚ずつ計算させる．

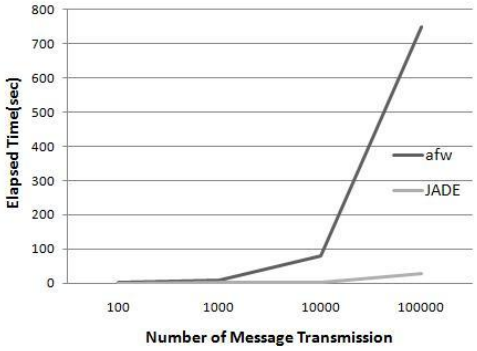

Fig.3 Transmision Time

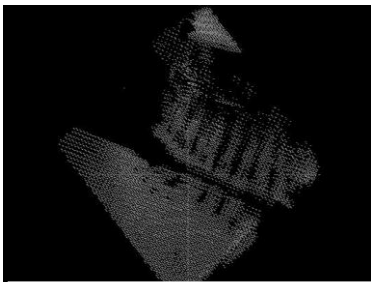

Fig.4 Bone parts of CT images

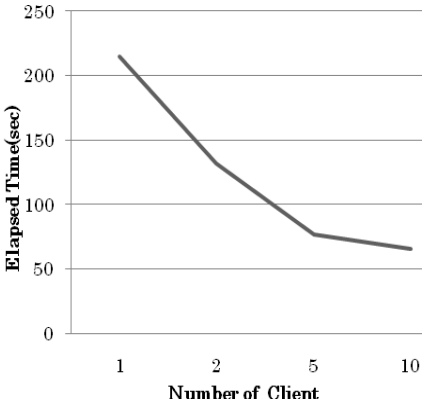

Fig.5 Speedup ratio

また，複数マシン環境における各マシンのエージェント数は均等になるものとする．

Fig.5に示すように計算時間はマシン数が多くなるに従い減少し，マシン数が10台のときは1台のときに比べ3割程度になっている．しかし，ネットワーク通信負荷やエージェント通信負荷によってマシン数が5台になると収束し始めており，この計算条件においてはこれ以上の速度向上は難しいと考えられる．

4.　結言

略

「参考文献」

1) Yasukazu NISHI, Mamoru Murata et al., Structural Analysis Method by Multiagent 2nd European Conference on Computational Mechanics, IACM, Jun. 2001.

2) 西恭一，大沼真也，ネットワーク環境における並列・分散マルチエージェントプラットホームの開発，計算工学会論文集　JSCES2002, 2002, No.20020016.

3) http://jade.cselt.it/