L-013

移動エージェントを用いた負荷分散システム
Load Balancing System for Mobile Agent Computing

古木 靖啓 * 菅谷 至寛 * 阿曽 弘具 *
Yasuhiro Furuki Yoshihiro Sugaya Hirotomo Aso

1. はじめに

分散計算機リソースの効率的な使用を促す技術は非常に重要であり、その 1 つとして移動エージェントシステムがある。移動エージェントは、プロセスを実行中に中断し他ホストへ移動後それを再開させるタスクマイグレーションが可能であるため、分散アプリケーションを構築するために役立つ。しかし、リソースを効率的に使用するようなポリシーを持っていない場合、計算機における計算負荷の不均衡が発生する可能性がある。また、エージェントで実行されるタスク同士で通信が発生する場合、移動をすることでタスク間の通信によるオーバヘッドが問題となることもある。本研究では IBM で開発された Java による移動エージェント実行環境 Aglets [3] を用いて負荷分散システムを構築し、その移動ポリシーとして計算負荷と通信コストの両方に注目した負荷分散アルゴリズムを提案することで効率的な負荷分散が可能であることを示す。

2. 移動エージェントにおけるタスク処理

移動エージェントを利用した本システムのタスク処理の様子を図 1 に示す。タスク処理はマスタスレーブ型の構造で行われる。ユーザがマスタエージェントへタスクを投入するとそのタスクを処理するスレーブを生成し、スレーブにおけるタスク処理終了後、最終的な結果を出力する。また、今回対象としているタスクは処理中に互いに通信を行うものとする。

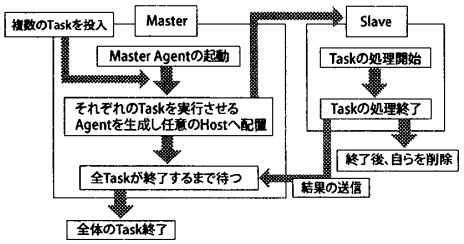

図 1: Task 処理の流れ

Aglets における移動エージェントはタスクマイグレーション時に静的オブジェクトの情報のみを保存するため、移動後タスクを中断状態から再開することができない。本研究ではタスクマイグレーション時にプロセスの中断状態を全て静的オブジェクトに保存することで中断状態からの再開を実現している。そのためコストは一時的に高くなるが全体としては高速な処理が可能である。

3. 負荷分散システム

本研究で構築する負荷分散システムは分散管理型システムである。分散管理型システムはそれぞれの計算機で独立して負荷情報の管理とタスクマイグレーションの決定を行うためネットワーク全体の情報を得ることはできないが一般的に集中管理型のシステムよりも大規模な環境にも対応できる。

3.1 使用するエージェントの分類

本システムの概要を図 2 に示す。本システムでは図 2 に示される 3 種類のエージェントを用いる。

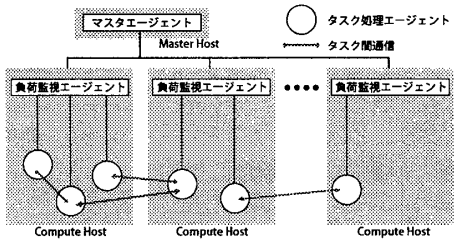

図 2: 負荷分散システムの概要

それぞれのエージェントについて簡単に述べる。

1. マスタエージェント
   システム起動時に生成される。ユーザは処理すべきタスクをこのエージェントに投入する。タスクを受け取ったエージェントはタスク処理エージェントを生成し、処理結果を待つ。マスタエージェントは異なるホストでの複数の起動も可能である。
2. 負荷監視エージェント
   システム起動時にマスタエージェントによって生成される。各ホストに 1 つ配置され、自ホストの負荷情報の管理、他ホストとの通信の監視を行い、それらの情報に基づいて移動するべきタスク処理エージェントとその移動先ホストを決定する。タスク処理エージェントの移動の判断とその移動先ホスト決定方法に関しては 4 章で述べる。
3. タスク処理エージェント
   ユーザからのタスク投入時にマスタエージェントによって生成される。ユーザが投入したタスクを実際に処理し、また、負荷監視エージェントの通知に基づいて移動する。

3.2 タスク処理エージェント間の通信の監視

本研究ではホスト間のレイテンシが未知である環境を想定している。そのためタスク処理エージェント間の通信を負荷監視エージェントが監視する。今回は監視を容易にするためタスク処理エージェント間の通信は直接ではなく負荷監視エージェントを経由して行われ、負荷監視エージェントはタスク間通信の際ホスト間のレイテンシを学習する。

4. 負荷分散アルゴリズム

本システムで用いる負荷分散アルゴリズムは各ホストの計算負荷情報と通信コストの両方を考慮し、さらに非

*東北大学大学院工学研究科

均質環境におけるホストの処理能力の差も考慮する。非均質環境への適応手段として負荷監視エージェント起動時に一定のサイズのプロセスを各ホストで実行する。このプロセスの処理時間の逆数を各ホストの処理能力として用いる。この値が大きい程ホストの処理能力は高い。また、本研究では局所的な情報のみで効率的な負荷分散の達成を目的としているため各ホストで随時取得可能な情報は自ホストの負荷情報のみとする。

4.1 計算負荷値の定義

一般的に Linux 環境では容易に得られる計算負荷値の指標として load average を用いることが多い。この時ホスト i において得られる負荷値 $load(i)$ はカーネルの実行待ちキューに並んでいるプロセスの一定時間の平均を表している。しかし、Aglets において生成される移動エージェントにおけるタスクはカーネルでは複数のプロセスとして認識されない。そのため $load(i)$ を負荷値として定義することは不適切であるといえる。また、1 台のホストが P 個のプロセッサを搭載しているような場合、その計算能力の優位性も考慮するべきである。そこでユーザレベルの負荷値 $u\_load(i)$ を $load(i)$ と、ホスト i に滞在するエージェント数 $agent(i)$ を用いて以下のように定義した。

if $agent(i) = 0$

$$u\_load(i) = \begin{cases} 0 & (0 \leq load(i) \leq (P-1)) \\ load(i) - 1 & ((P-1) < load(i) \leq P) \\ load(i)/P & (P \leq load(i)) \end{cases} \quad (1)$$

if $agent(i) \geq 1$

$$u\_load(i) = \begin{cases} agent(i) & (load(i) < P) \\ load(i) \cdot agent(i)/P & (P \leq load(i)) \end{cases} \quad (2)$$

この $u\_load(i)$ と各ホストの処理能力を用いてエージェントの移動の判断を行う。今回は $u\_load(i)$ にホストの処理能力の逆数を掛けた値を各ホストの相対的な移動判断値とし、適当な閾値によって移動の決定を行う。

4.2 移動先ホストの決定

4.1 節の方法で移動が決定したエージェントの移動先のホストを決定する。この時ホスト間のレイテンシを考慮するために 2 段階の移動要求の生成とその応答によって移動先ホストを決定する。負荷監視エージェントはまず、移動が決定したエージェントと通信を行っているタスク処理エージェントが存在するホストの負荷監視エージェントへ移動要求を発生させる。移動要求を受け取った負荷監視エージェントは自ホストの負荷状況に応じて移動受入可否の応答をするとともにレイテンシが既知となっているホストの中で自ホストとのレイテンシが小さいホストへ更に移動要求を発生させる。このレイテンシの大小は閾値によって決定する (4.3 節)。新たに移動要求を受け取ったホストも負荷状況に応じた応答をし、最終的に複数の移動受入可能応答を返したホストを移動先ホストとして決定する。

4.3 レイテンシに関する閾値の更新

本システムでは負荷監視エージェントが通信の監視を行っているためにホスト間のレイテンシに関する情報が随時更新されていく。そのため閾値も最新の情報に基づいて更新してく必要がある。移動要求発生時に通信コストが既知のものからその平均を求め、その平均を閾値とした。

5. 実験

本研究で構築した分散システムの性能評価を行う。実験環境を表 1 に示す。

表 1: 実験環境

	CPU	MEMORY	台数
Master Host	Xeon 2.4GHz×2	1GB	1
Compute Host	Xeon 2.4GHz×2	1GB	2
	Pentium3 1GHz×2	512MB	2
	Pentium3 800MHz×2	512MB	2
	Pentium3 500MHz×2	512MB	2

本環境は LAN により構成されているため実際の通信によるコストが 0 に近い。今回は大規模なネットワークを想定し、乱数を用いてホスト間の通信時間を人工的に決定した。また、各エージェントへ投入されるタスクの条件 (タスクの大きさ、通信パターン等) も全て乱数によって決定した。以上の条件において以下の 3 つのシステムについて比較を行った。

システム **A** タスク投入時のみの分散システム (タスク処理中のエージェントの移動はなし)

システム **B** 集中管理型システム (ネットワーク全体の計算負荷情報から移動先ホストを決定)

システム **C** 分散管理型システム (本研究で構築したシステム)

評価指標は全タスク終了までの時間と、その時要した通信時間の合計である。結果を表 2 に示す。表 2 よりタスク間の通信時間を考慮することの有効性を確認できる。

表 2: 実験結果

	全タスク処理時間 (s)	通信に要した時間 (s)
システム A	756.84	192.13
システム B	625.10	311.21
システム C	416.85	159.02

6. 結論

本研究では移動エージェント環境 Aglets を用いて負荷分散システムを構築した。計算負荷だけではなくタスク間の通信コストも考慮した移動ポリシーを提案し、その有効性を実験によって確認した。

参考文献

[1] Ka-Po Chow et al.: "On Load Balancing for Distributed Multiagent Computing",IEEE Trans. on Parallel and Distributed Systems, Vol.13, No.8, pp.787-801,Aug.2002.

[2] 沢野泰淳 他 : "移動エージェントを用いたネットワーク負荷分散システムの構築とその評価", 信学論,Vol.J84-D-I,No.9,pp.1450-1453,Sep.2000.

[3] DNNY B.LANGE et al.: "PROGRAMMING AND DEPLOYING JAVA MOBILE AGENTS WITH AGLETS",Addison-Wesley.