社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

広域グリッドコンピューティングのための制御理論にもとづく動的資源管理方式

大崎　博之† 　渡部　創史† 　今瀬　真†

† 大阪大学 大学院情報科学研究科
〒 565-0871 大阪府吹田市山田丘 1-5
E-mail: †{oosaki,s-watanb,imase}@ist.osaka-u.ac.jp

あらまし　近年、地理的に分散した計算機資源を、ネットワークを介して統合的に利用するというグリッドコンピューティングが注目を浴びている。地理的に遠く離れた計算機資源を利用した、広域グリッドコンピューティングでは、従来のクラスタ計算機などを利用した並列計算とは異なり、以下のような問題が発生する。(1) 資源が複数の利用者によって共有されるため、サイト内の利用可能資源量が時間とともに変動する。(2) サイトが地理的に広い範囲に渡って分散しているため、サイト間の転送遅延が無視できない。計算機資源などの資源を有効に活用するためには、時間とともに変動する利用可能資源量にあわせて、サイトへ投入するジョブ量を動的に調整する必要がある。ただし、ネットワークの転送遅延が非常に大きいために、単純な資源割り当て制御では、サイト内の利用可能資源量の変化に追従することはできない。そこで、本稿では、グリッドにおける制御理論にもとづいた動的資源管理方式 DRM-DC (Dynamic Resource Management with Delay Compensator) を提案する。提案する DRM-DC の特徴は、遅延補償器 (スミス予測器) を用いることにより、転送遅延の大きな広域グリッドコンピューティングにおいて、高い定常特性と過渡特性を実現するという点にある。本稿では、Simgrid シミュレータを用いた離散時間シミュレーションを行い、提案する DRM-DC の有効性を示す。
キーワード　グリッドコンピューティング、動的資源管理、制御理論、転送遅延、遅延補償器、スミス予測器

On Dynamic Resource Management Mechanism using Control Theoretic Approach for Wide-Area Grid Computing

Hiroyuki OHSAKI†, Soushi WATANABE†, and Makoto IMASE†

† Graduate School of Information Science and Technology, Osaka University
1-5 Yamadaoka, Suita, Osaka, 565-0871 Japan
E-mail: †{oosaki,s-watanb,imase}@ist.osaka-u.ac.jp

Abstract　In recent years, Grid computing that integrates geographically distributed computing resources through communication networks captures the spotlight. Unlike parallel computing using conventional cluster computer systems, wide-area Grid computing must resolve the following issues for effectively utilizing geographically distributed computing resources. First, since computing resources are shared by multiple users, the amount of available resources in a site changes over time. Second, since sites are geographically distributed, the transfer delay between sites cannot be neglected for computing resource management. For utilizing computing resources effectively, the amount of jobs injected into a site must be dynamically controlled according to the dynamically changing amount of available resources in sites. However, the transfer delay of a network is significant, so that it is not trivial for a resource allocation controller to quickly and dynamically adopt to the change in the amount of available resources in sites. In this paper, a dynamic resources management mechanism for wide-area Grid computing called *DRM-DC (Dynamic Resource Management with Delay Compensator)* based on control theory is proposed. The main feature of our DRM-DC is that it realizes high steady state and transient state performance in wide-area Grid computing using a delay compensator called *Smith predictor*. Moreover, several discrete-time simulations using a Simgrid simulator are performed, and the effectiveness of our DRM-DC is demonstrated.
Key words　Grid Computing, Dynamic Resource Management, Control Theory, Transfer Delay, Delay Compensator, Smith Predictor

1　はじめに

近年、地理的に分散した計算機資源を、ネットワークを介して統合的に利用するというグリッドコンピューティングが注目を浴びている [1, 2]。グリッドコンピューティングは、従来、利用率がそれほど高くなかった、計算機の空き資源を利用しようという試みである。

地理的に遠く離れた計算機資源を利用したグリッドコンピューティングを、特に「広域グリッドコンピューティング」と呼ぶ。広域グリッドコンピューティングでは、従来のクラスタ計算機などを利用した並列計算とは異なり、さまざまな問題が発生する。

第一の問題は、サイト内の利用可能資源量が時間とともに変動することである。一般に、グリッドコンピューティングでは、資源が複数の利用者によって共有されるため、サイト内の利用可能資源量が時間とともに変動する。例えば、サイト内の計算

機資源は、同時に実行されているジョブ数に応じて変化する。また、ネットワーク資源は、他の利用者の通信状態に応じて変化し、ディスク資源は、他の利用者が情報を保存もしくは削除することにより変化する。

第二の問題は、サイトが地理的に広い範囲に渡って分散しているため、サイト間の転送遅延が無視できないという点である。これは、広域グリッドコンピューティング特有の問題である。例えば、広域ネットワークでは、サイト間のラウンドトリップ時間が 100 ms 以上のように、クラスタ計算機のような LAN 環境と比較して非常に大きな値となる。そのため、各サイトにおける資源の空き状況の変動等を、リアルタイムに把握することが困難である。

このように、広域グリッドコンピューティングでは、利用可能資源量が時間とともに変動し、なおかつネットワークの転送遅延が無視できないという問題がある。計算機資源などの資源を有効に活用するためには、時間とともに変動する利用可能資源量にあわせて、サイトへ投入するジョブ量を動的に調整する必要がある [3, 4]。ただし、ネットワークの転送遅延が非常に大きいために、単純な資源割り当て制御では、サイト内の利用可能資源の変化に追従することはできない。

そこで、本稿では、制御理論にもとづいた動的資源管理方式 DRM-DC (Dynamic Resource Management with Delay Compensator) を提案する。DRM-DC では、サイトにおける利用可能資源量の変動に追従するため、フィードバック制御を使用する。つまりサイトの利用状況をフィードバックすることにより、サイト内の資源の有効利用を図る。提案する DRM-DC の特徴は、遅延補償器 (スミス予測器) を用いることにより、転送遅延の大きな広域グリッドコンピューティングにおいて、高い定常特性と過渡特性を実現するという点にある。遅延補償器とは、制御遅延の大きな状況であっても、制御対象の数学的モデルを利用することにより、制御遅延の影響を打ち消すことができる (制御遅延をゼロとみなすことができる) 手法である [5]。遅延補償器を用いることにより、転送遅延の大きな広域グリッドコンピューティングでも、資源が過負荷に陥ったり、空き状態になることを防ぎ、サイト内の利用可能資源の有効利用を図る。

本稿では、グリッドにおける資源管理を、連続時間系の制御システムとしてモデル化する。サイトは割り当てられた一連のタスクを順次処理することから、サイトを単一の待ち行列としてモデル化する。実際には、入力がジョブマネージャからのタスク到着レート、出力がサイトのローカル資源マネージャにおけるバッファ内タスク数であるような、積分器によってモデル化する。また、ジョブマネージャは、サイトからのフィードバック情報にもとづき、サイトへのタスク投入量を調整する。このため、ジョブマネージャを、入力がサイトからのフィードバック情報 (バッファ内タスク数)、出力がタスク投入レートである、PI (Proportional Integral) 制御器によってモデル化する。本稿では、単一の資源マネージャと複数のサイトが存在する環境を対象とし、各サイトのローカル資源マネージャの待ち行列長 (バッファ内タスク数) がジョブマネージャにフィードバック情報として通知することとする。さらに、Simgrid シミュレータ [6] を用いた離散時間シミュレーションを行い、提案する DRM-DC の有効性を検証する。具体的には、提案する DRM-DC と、遅延補償器を使用しない PI 制御器を比較することにより、提案する DRM-DC が、広域グリッドコンピューティングにおいて、良好な定常特性および過渡特性を実現することを示す。

本稿の構成は以下の通りである。まず、2 章では、関連研究を紹介する。3 章では、グリッドコンピューティングにおける資源管理を説明する。4 章では、グリッドコンピューティングにおける資源管理を、どのようにフィードバック制御としてモデル化するかを説明する。5 章では、我々が提案する、遅延補償器を利用した動的資源管理方式 DRM-DC のアルゴリズムを説明する。6 章では、シミュレーション実験により、提案する資源管理方式 DRM-DC の有効性を検証する。最後に 7 章において、本稿のまとめと今後の課題について述べる。

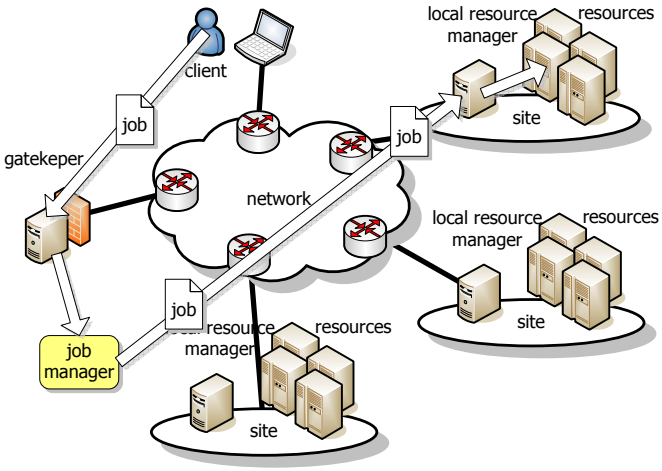

図 1　グリッドコンピューティングにおける資源管理
Fig. 1　Resource management in Grid computing

2　関 連 研 究

本稿では、広域グリッドコンピューティングにおいて、利用可能資源量が時間とともに変動し、なおかつネットワークの転送遅延が無視できないという問題を、制御理論における遅延補償器を用いることにによって解決する。転送遅延の大きいネットワークにおけるフィードバック制御を、遅延補償器によって解決しようという試みとして、文献 [7, 8] が挙げられる。例えば、文献 [7] では、ATM (Asynchronous Transfer Mode) 網におけるレート制御方式に、遅延補償器 (スミス予測器) を適用する手法を提案している。フィードバック情報として、ボトルネックルータのキュー長 (バッファ内パケット数) を使用し、単純な P (Proportional) 制御を用いて、ボトルネックルータのキュー長を制御している。その際、リンクの転送遅延による制御遅延を補償するためにスミス予測器を用いている。ただし、制御器が単純な P 制御器であり、定常偏差が発生する (定常状態において、ボトルネックルータのキュー長が、制御目標である目標キュー長に一致しない) という問題がある。一方、本稿で提案する DRM-DC では、定常偏差を解消するために、積分要素を持つ PI 制御器を利用することにより、広域グリッドコンピューティングにおいて、良好な定常特性および過渡特性を実現する。

3　グリッドコンピューティングにおける資源管理

まず、グリッドコンピューティングにおける資源管理の概要を説明する。ここでは、 Globus ツールキット [9] に含まれる、資源管理のコンポーネント GRAM (Grid Resource Allocation and Management) を例にとって説明する。

GRAM の構成要素には、クライアント、ゲートキーパ、ジョブマネージャ、ローカル資源マネージャ、サイトが存在する (図 1)。サイトとは、ある仮想組織に属する資源の集合 (例えば、クラスタ計算機) を意味する。GRAM におけるジョブ実行の流れを説明する。まず、クライアントは、ジョブを生成し、ゲートキーパを通じてジョブの実行を依頼する。ゲートキーパは、クライアントを認証し、実際にジョブ実行を管理するジョブマネージャを生成する。ジョブマネージャは、各サイトのローカル資源マネージャに対してジョブを割り当てるとともに、そのジョブの実行状態を監視する。ローカル資源マネージャは、割当てられたジョブをサイト内の計算機資源を用いて実行し、実行結果をジョブマネージャに返送する。

グリッドコンピューティングにおける資源管理の目的は、時間とともに変動する各サイトの利用可能資源量にあわせて、サイトへ投入するジョブ量を動的に調整することである [3, 4]。つまり、各サイトの利用可能資源量をもとに、ゲートキーパから生成されるジョブマネージャが、どのサイトのローカル資源

マネージャに、どれだけのジョブを割り当てるかを制御することである。本稿では、変数探索型アプリケーション [10] のような、多数の独立した処理 (タスクと呼ぶ) によって構成されるジョブを実行する場合を対象とする。つまり、クライアントから生成されるジョブは、粒度の小さな多数のタスクによって構成されている場合を対象とする。

次に、グリッドコンピューティングにおける資源管理では、どのような特性が重要となるかを議論する。まず、システムが安定した状態におけるふるまい (定常特性) が重要であろう。サイトの資源が、過負荷にもアイドル状態にもならず、高い利用率を維持できることが望ましい。また、サイトにジョブを割り当ててから、実際にジョブの実行が完了するまでの時間が短いことが望ましい。

グリッドコンピューティングでは、サイトの空き資源量が時間とともに変動する。このため、立ち上がり時間といった、系が安定するまでのふるまい (過渡特性) も重要である。さらに、遅延時間の大きな環境下において、サイトの利用可能資源量をもとに制御を行うため、ある程度の時間が経過すればシステムが安定するかどうか (安定性) も重要な性能指標となる。

さらに、グリッドコンピューティングにおいては、ネットワーク機器や計算機の障害などが発生することも考えられる。このため、障害発生が発生したとしても正常に動作するか (堅牢性) が重要である。例えば、ネットワーク機器の障害などにより、一時的にネットワーク遅延が大きくなった場合でも、システムが正常に動作し、安定性を維持できることが望ましい。

グリッドコンピューティングでは、一般に、サイトの処理能力やネットワークの遅延時間などが不均一である。このため、このような不均一な環境、つまり、さまざまなパラメータ条件下でも、安定性や堅牢性を実現できること (柔軟性) が望ましいと考えられる。

4 フィードバック制御としてのモデル化

以下では、グリッドコンピューティングにおける資源管理を、どのようにフィードバック制御としてモデル化するかを説明する

グリッドコンピューティングでは、サイトの資源が複数の利用者によって共有されるため、利用可能資源量が時間とともに変動する [1, 2]。グリッドコンピューティングの資源管理では、利用可能資源量の変動を予測したスケジューリング行うのは困難である。このため、利用可能資源量の変動にあわせて動的に制御を行うことが求められる。つまり、グリッドコンピューティングの資源管理では、単純な開ループ型の制御では不十分であり、サイトからフィードバック情報を利用した閉ループ型の制御が不可欠となる。

また、広域グリッドコンピューティングでは、地理的に広い範囲にサイトの資源が散在しているため転送遅延が大きく、資源管理において転送遅延を無視できない。転送遅延が大きいために、ジョブマネージャがサイトの資源情報を入手するために時間がかかったり、ジョブマネージャがサイトに割り当てたジョブがサイトへ到着するまでに時間がかかることがある。このため、単純なフィードバック制御では安定性や堅牢性を実現できない。広域グリッドコンピューティングの資源管理では、このような転送遅延に対処することが求められる、つまり、フィードバック遅延を考慮した制御が必要となる。

そこで以下では、グリッドコンピューティングにおける資源管理を、連続時間系の制御システムとしてモデル化する。

図 1 のような、複数のサイトから構成されるネットワークを考える。ここで、あるサイトに注目する。時刻 t におけるサイトの利用可能資源 (単位時間あたりに処理できるタスク数) を $\mu(t)$、ジョブマネージャ–サイト間の転送遅延を $\tau(t)$ と表記する。また、時刻 t における、ジョブマネージャからサイトへのタスク投入レート (単位時間あたりの割り当てタスク数) を $u(t)$ とする。時刻 t における、サイトのローカル資源マネージャのバッファ内に格納されているタスク数 (キュー長) を $x(t)$ とする。

まず、サイトは割り当てられた一連のタスクを順次処理することから、サイトを単一の待ち行列としてモデル化できる。ジョブマネージャから投入されたタスクは、いったんローカル資源マネージャのバッファに格納される。ローカル資源マネージャは、サイトの利用可能資源量に応じて、バッファに格納されたタスクを順次実行する。このため、サイトは、処理速度が $\mu(t)$ であるような待ち行列としてモデル化することができる。このため、フィードバック制御という観点で考えると、サイトは制御対象 (プラント) に相当する。サイトを連続時間系の制御システムとしてモデル化する場合は、入力がジョブマネージャからサイトへのタスク到着レート $u(t)$、出力 (ジョブマネージャへのフィードバック情報) がサイトのローカル資源マネージャにおけるバッファ内タスク数 $x(t)$ である、以下のような積分器によってモデル化できる。

$$x(t) = \left[\int_0^t (u(v-\tau(v)) - \mu(v))dv)\right]^+ \quad (1)$$

ここで、$[x]^+ \equiv \max(0, x)$ と定義する。

一方、ジョブマネージャは、サイトからのフィードバック情報にもとづき、サイトへのタスク投入量を調整する。このため、ジョブマネージャを、入力がサイトからのフィードバック情報 (バッファ内タスク数) $x(t)$、出力がサイトへのタスク投入レート $u(t)$ である、制御器としてモデル化できる。このため、フィードバック制御という観点で考えると、ジョブマネージャは、サイトへのタスク投入量を調整する制御器 (コントローラ) に相当する。

つまり、広域グリッドコンピューティングにおける資源管理は、利用可能資源量が時間とともに変動し、なおかつネットワークの転送遅延が無視できないという状況下で、ジョブマネージャ (制御器) をどのように設計するかいう問題に帰着される。そこで次章では、遅延補償器 (スミス予測器) を用いることにより、転送遅延の大きな広域グリッドコンピューティングにおいて、高い定常特性と過渡特性を実現する制御器 DRM-DC を設計する。

5 動的資源管理方式 DRM-DC

本章では、我々が提案する、遅延補償器を利用した動的資源管理方式 DRM-DC (Dynamic Resource Management with Delay Compensator) のアルゴリズムを説明する。DRM-DC は、ジョブマネージャ上で動作する制御器 (コントローラ) であり、サイトの利用可能資源量にもとづき、ローカル資源マネージャへのタスク投入レートを制御する。一般には、ジョブマネージャが管理するサイトは複数存在する。そこで、DRM-DC では、それぞれのサイトごとに独立にフィードバック型の制御を行う。

DRM-DC ではサイトのローカル資源マネージャのバッファ内に格納されているタスク数 (キュー長) をフィードバック情報とした、閉ループ型のフィードバック制御を行う。制御目標は、ローカル資源マネージャのキュー長を一定値に保つことである。これにより、サイト内資源が過負荷になることを防ぐとともに、サイト内資源の利用率の向上を図る。また、ローカル資源マネージャのキュー長を低く抑えることにより、ローカル資源マネージャがタスクを投入してから、ジョブマネージャがタスクの実行結果を受け取るまでの待ち時間を小さくする。

グリッドコンピューティングにおける資源管理を、フィードバック制御としてとらえると、制御対象はサイト内の資源であり、操作量はジョブマネージャからサイトへのタスク投入レートとなる。DRM-DC では、利用可能資源量が時間とともに変動し、なおかつネットワークの転送遅延が無視できないという問題を、制御理論における遅延補償器を用いることににによって解決する。DRM-DC では、遅延補償器を用いた PI (Proportional Integral) 制御 [5] を行うことにより、定常特性の劣化を防ぐ。

本稿では、以下のような仮定を置いている。(1) タスクの粒度はジョブマネージャ– サイト間の転送遅延と比較して十分小

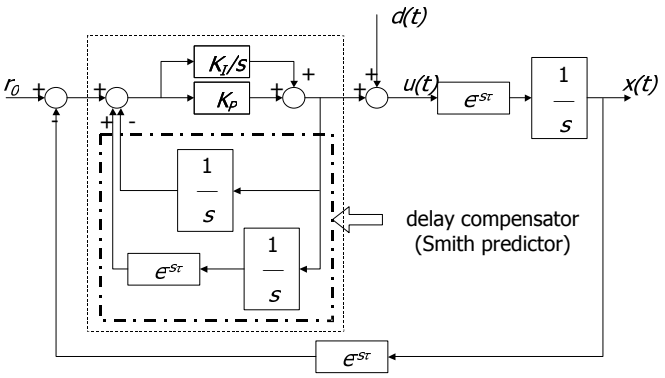

図 2　動的資源管理方式 DRM-DC のブロック図
Fig. 2　Block diagram of DRM-DC (Dynamic Resource Management with Delay Compensator)

表 1　記号の定義

r_0	ローカル資源マネージャの目標キュー長
$x(t)$	時刻 t におけるローカル資源マネージャのキュー長
$u(t)$	時刻 t におけるローカル資源マネージャへのタスク投入レート
$d(t)$	時刻 t における外乱
K_P	DRM-DC の制御パラメータ (比例要素のゲイン)
K_I	DRM-DC の制御パラメータ (積分要素のゲイン)
τ	ジョブマネージャ–サイト間の転送遅延

さい、(2) ジョブマネージャ–サイト間の伝搬遅延は既知であり、時間とともに変動しない、(3) サイトの処理能力 (単位時間あたりに処理できるタスク数) はローカル資源マネージャのバッファ内タスク数によらず一定である、(4) ジョブマネージャは常に実行すべきジョブ (複数のタスクによって構成される) を持つ。

動的資源管理方式 DRM-DC のブロック図を、図 2 に示す。また、本稿で用いる記号の定義を、表 1 に示す。

次に、3 章で述べた、グリッドコンピューティングにおける資源管理で重要となる特性を、提案する DRM-DC がどの程度満たしているかを定性的に議論する。まず、DRM-DC はローカル資源マネージャのバッファ内タスク数を一定値に制御する。このため、サイトの資源の有効利用を図りながら、資源の過負荷を防ぐことができ、良好な定常特性が期待できる。また、制御目標 r_0 を適切に設定することにより、サイトにジョブを割り当ててから、実際にジョブの実行が完了するまでの時間を短かく抑えることができると考えられる。

提案する DRM-DC は、ローカル資源マネージャのバッファ内タスク数をフィードバック情報として用いることで、サイトの利用可能資源量の変化に追従するため、良好な過渡特性が期待できる。ただし、DRM-DC の有効性は、PI 制御器の制御パラメータ (K_P および K_I) に大きく依存すると考えられる。一方、DRM-DC は遅延補償器を用いた PI 制御器であるため、制御パラメータ (K_P および K_I) を適切に設定することにより、システムの安定性を高めることができると考えられる。

堅牢性および柔軟性は、遅延補償器で用いる制御対象 (サイト) のモデルおよび転送遅延などのパラメータの正確性に依存すると考えられる。現実には、サイトの厳密な数学的モデルを得ることは不可能であり、また、ジョブマネージャ– ローカル資源マネージャ間の転送遅延も、ネットワークの輻輳などが原因となり、時間とともに変動する。従って、制御対象のモデルや転送遅延に含まれる誤差が、DRM-DC の堅牢性および柔軟性にどのような影響を与えるかを評価する必要がある。そこで 5 章では、シミュレーション実験により、DRM-DC の有効性を検証する。

6　シミュレーションによる性能評価

本章では、シミュレーション実験により、提案する動的資源管理方式 DRM-DC の性能評価を行う。

表 2　シナリオ 1 のパラメータ設定
Table 2　Parameter configuration in Scenario 1

		LAN-S2	LAN-S20	WAN-S2	WAN-S20
タスクサイズ S	[MI]	2	20	2	20
転送遅延 τ	[ms]	1.0	1.0	100.0	100.0
制御間隔 T	[ms]	2.0	20.0	2.0	20.0
目標キュー長 r_0		200	20	200	20

シミュレーション実験には、グリッド用の離散時間シミュレータ Simgrid [6] を修正して使用した。Simgrid は、マスタ–ワーカ型アプリケーションのためのシミュレータである。DRM-DC のシミュレーションでは、ジョブマネージャがマスタとなり、サイト (ローカル資源マネージャおよび資源) がワーカとなる。DRM-DC は連続時間系の制御であるが、Simgrid は離散時間シミュレータである。そこで、Simgrid のマスタとして、一定の制御間隔 T ごとに制御を行うような、離散時間型の DRM-DC を実装した。また、Simgrid のワーカとして、バッファサイズが有限の FIFO キューを実装した。

ジョブマネージャが管理するサイト数を 10 とした。ネットワーク環境として、LAN 環境 ($\tau = 1$ [ms]) および WAN 環境 ($\tau = 100$ [ms]) の場合のシミュレーションを実行した。また、各タスクが必要とする資源量はすべて等しいものとした。パラメータ探索型のアプリケーションを想定し、ジョブマネージャには常に実行すべきタスクが存在するものとし、各タスクの実行に必要な浮動小数点演算数 (タスクサイズ) S を 2 [MI (Million Instructions)] または 20 [MI] とした。比較のため、遅延補償器を用いない PI 制御器のシミュレーションもあわせて実行した。

シミュレーション実験における、DRM-DC の性能指標として、ローカル資源マネージャのキュー長 (バッファ内タスク数) に着目する。特に、キュー長の時間的変動、立ち上がり時間 (rise time) 、行き過ぎ量 (overshoot) 、整定時間 (settling time) を性能指標として用いる [5]。具体的には、キュー長の立ち上がり時間として、シミュレーションの実行の開始直後に、キュー長がキュー長目標値の 10% から 90% に遷移するまでの時間を測定した。行き過ぎ量として、キュー長の最大値がキュー長の目標値をどれだけ上回ったかを測定した。整定時間として、キュー長がキュー長の目標値の 5% 以内に安定するまでの時間を測定した。15 秒間のシミュレーションを行い、上記の性能指標を測定した。

本稿では、以下の 2 つのシナリオに対してシミュレーションを実行した。

(1) シナリオ 1: サイトの利用可能資源量が時間とともに変動する場合

(2) シナリオ 2: ジョブマネージャ–サイト間の転送遅延が時間とともに変動する場合

まず、サイトの利用可能資源量 $\mu(t)$ が時間とともに変動する場合 (シナリオ 1) の結果を示す。シナリオ 1 で用いたパラメータ設定を表 2 に示す。シナリオ 1 では、サイトの利用可能資源量を時間とともに変動させて、シミュレーションを実行した。具体的には、サイトの利用可能資源量 $\mu(t)$ を以下のように変化させた (単位は MIPS)。

$$\mu(t) = \begin{cases} 1,000 & 0 \leqq t < 4 \\ 200 & 4 \leqq t < 10 \\ 1,500 & \text{otherwise} \end{cases} \quad (2)$$

シナリオ 1 のシミュレーション結果 (立ち上がり時間、行き過ぎ量、整定時間) を表 3 に示す。

LAN 環境における、ローカル資源マネージャのキュー長の変動を、図 3 (LAN-S2) および図 4 (LAN-S20) に示す。図 3 (LAN-S2) は、タスクの粒度が細かい場合 ($S = 2$ [MI]) の結果である。一方、図 4 (LAN-S20) は、タスクの粒度が荒い場合 ($S = 20$ [MI]) の結果である。

これらの図より、転送遅延の小さな LAN 環境においては、PI

表 3　シナリオ 1 のシミュレーション結果
Table 3　Simulation results in Scenario 1

		転送遅延 τ [ms]	タスクサイズ S [MI]	立ち上がり時間 [s]	行き過ぎ量 [%]	整定時間 [s]
LAN-S2	DRM-DC	1.0	2	0.016	0.5	0.012
LAN-S2	PI	1.0	2	0.016	0.5	0.008
LAN-S20	DRM-DC	1.0	20	0.4	5	0.8
LAN-S20	PI	1.0	20	0.4	5	0.8
WAN-S2	DRM-DC	100.0	2	0.20	46	0.11
WAN-S2	PI	100.0	2	1.6	79	1.7
WAN-S20	DRM-DC	100.0	20	0.16	55	0.3
WAN-S20	PI	100.0	20	1.62	85	1.6

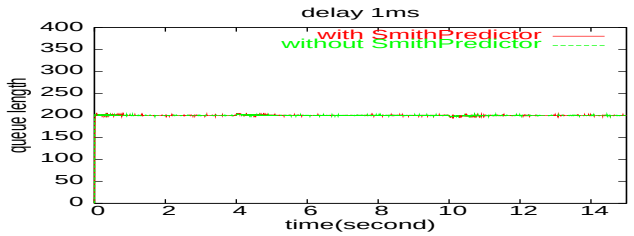

図 3　LAN-S2: LAN 環境で利用資源量が変動した時のキュー長 ($\tau = 1.0$ [ms], $S = 2$ [MI])
Fig. 3　Case LAN-S2: Queue dynamics for varying amoung of available resources in LAN environemnt ($\tau = 1.0$ [ms], $S = 2$ [MI])

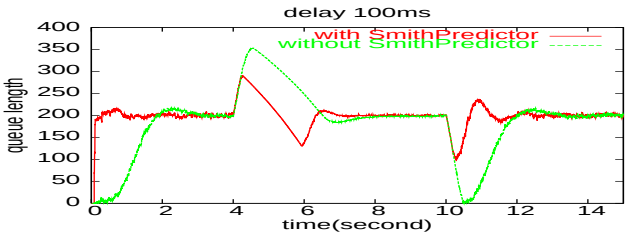

図 5　WAN-S2: WAN 環境で利用資源量が変動した時のキュー長 ($\tau = 100.0$ [ms], $S = 2$ [MI])
Fig. 5　Case WAN-S2: Queue dynamics for varying amoung of available resources in WAN environemnt ($\tau = 100.0$ [ms], $S = 2$ [MI])

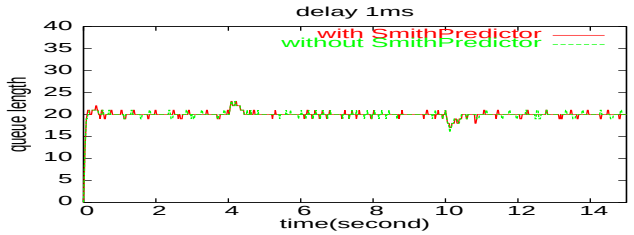

図 4　LAN-S20: LAN 環境で利用資源量が変動した時のキュー長 ($\tau = 1.0$ [ms], $S = 20$ [MI])
Fig. 4　Case LAN-S20: Queue dynamics for varying amoung of available resources in LAN environemnt ($\tau = 1.0$ [ms], $S = 20$ [MI])

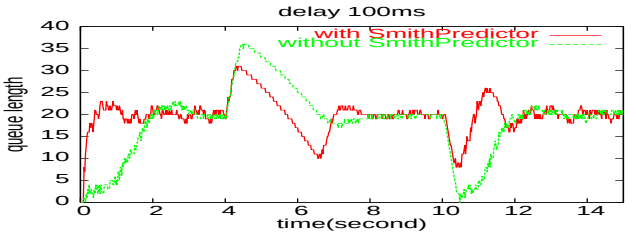

図 6　WAN-S20: WAN 環境で利用資源量が変動した時のキュー長 ($\tau = 100.0$ [ms], $S = 20$ [MI])
Fig. 6　Case WAN-S20: Queue dynamics for varying amoung of available resources in WAN environemnt ($\tau = 100.0$ [ms], $S = 20$ [MI])

制御および DRM-DC ともに良好な性能を示していることが分かる。つまり、キュー長の立ち上がり時間は十分小さく、行き過ぎ量も 5 % 以下と十分小さい。また、整定時間も 1 [s] 未満である (表 3)。サイトの利用可能資源量が $t = 4, 10$ [s] で変化しているが、これらの変化に即座に追従できている。タスクサイズが $S = 20$ [MI] と大きい時は、性能指標 (立ち上がり時間、行き過ぎ量、整定時間) が若干劣化しているが、これはタスクサイズが大きく、資源管理制御の粒度が荒いためと考えられる。

しかし、WAN 環境のように、転送遅延の大きいネットワークでは、単純な PI 制御では性能が大きく劣化してしまう。

WAN 環境における、ローカル資源マネージャのキュー長の変動を、図 5 (WAN-S2) および図 6 (WAN-S20) に示す。図 5 (WAN-S2) は、タスクの粒度が細かい場合 ($S = 2$ [MI]) の結果である。一方、図 6 (WAN-S20) は、タスクの粒度が荒い場合 ($S = 20$ [MI]) の結果である。

これらの図より、転送遅延の大きな WAN 環境では、PI 制御および DRM-DC の過渡特性に大きな差があることが分かる。特に、サイトの利用可能資源量が $t = 4, 10$ [s] で変化した直後に、その違いが顕著に表われている。PI 制御と DRM-DC の立ち上がり時間および整定時間を比較すると、DRM-DC の過渡特性が非常に優れていることがわかる (表 3)。また、DRM-DC の行き過ぎ量は、PI 制御の行き過ぎ量と比べて小さな値となっている。また、PI 制御では利用可能資源量が増加した直後 ($t = 11$ [s] 前後) でキュー長が 0 となっており、資源の利用率が低下している。

表 4　シナリオ 2 のパラメータ設定
Table 4　Parameter configuration in Scenario 2

		LAN-S20	WAN-S20
タスクサイズ S	[MI]	20	20
利用可能資源量 $\mu(t)$	[MI]	1000	1000
制御間隔 T	[ms]	20.0	20.0
目標キュー長 r_0		20	20

以上の考察から、遅延補償器を用いた DRM-DC は、特に転送遅延の大きな WAN 環境において、良好な定常特性および過渡特性を示すことが分かる。

次に、ジョブマネージャ–サイト間の転送遅延 τ が時間とともに変動する場合 (シナリオ 2) の結果を示す。シナリオ 2 で用いたパラメータ設定を表 4 に示す。シナリオ 2 では、ジョブマネージャ–サイト間の転送遅延を時間とともに変動させて、シミュレーションを実行した。具体的には、ジョブマネージャ–サイト間の転送遅延 τ を以下のように変化させた。

$$\tau \leftarrow \begin{cases} \tau & 0 \leqq t < 4 \\ 1.8\tau & 4 \leqq t < 10 \\ 0.8\tau & \text{otherwise} \end{cases} \tag{3}$$

シナリオ 2 のシミュレーション結果 (立ち上がり時間、行き過ぎ量、整定時間) を表 5 に示す。

LAN 環境における、ローカル資源マネージャのキュー長の変動を、図 7 (LAN-S20) に示す。また、WAN 環境における、ロー

表 5　シナリオ 2 のシミュレーション結果
Table 5　Simulation results in Scenario 2

		転送遅延 τ [ms]	タスクサイズ S [MI]	立ち上がり時間 [s]	行き過ぎ量 [%]	整定時間 [s]
LAN-S20	DRM-DC	1.0	20	0.06	5	0.08
LAN-S20	PI	1.0	20	0.06	5	0.08
WAN-S20	DRM-DC	100.0	20	0.1	100	0.24
WAN-S20	PI	100.0	20	0.52	100	0.7

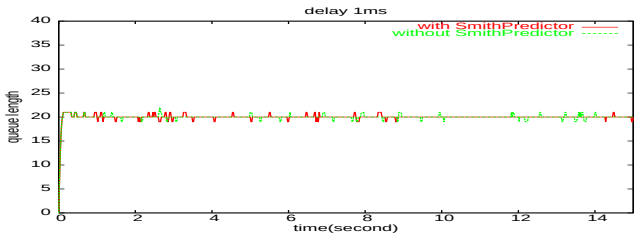

図 7　LAN-S20: LAN 環境で転送遅延が変動した時のキュー長 ($\tau = 1.0$ [ms], $S = 20$ [MI])
Fig. 7

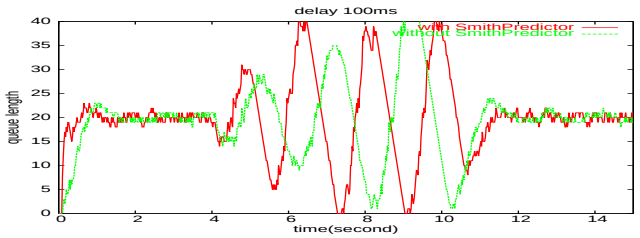

図 8　WAN-S20: WAN 環境で転送遅延が変動した時のキュー長 ($\tau = 100.0$ [ms], $S = 20$ [MI])
Fig. 8

カル資源マネージャのキュー長の変動を、図 8 (WAN-S20) に示す。これらのシミュレーションは、ともにタスクの粒度が荒い場合 ($S = 20$ [MI]) の結果である。

まず、転送遅延の小さな LAN 環境におけるシミュレーション結果 (図 7 (LAN-S20)) に着目すると、PI 制御、DRM-DC ともに良好な性能を示していることが分かる。表 5 の結果を見ても、キュー長の立ち上がり時間、行き過ぎ量、整定時間に関して、PI 制御と DRM-DC では性能がまったく同じである。

一方、転送遅延の大きな WAN 環境では、DRM-DC が優れた定常特性および過渡特性を示す。図 8 (WAN-S20) より、PI 制御に比べて、DRM-DC はより素早く転送遅延の変動に対応していることが分かる。また表 5 より、立ち上がり時間および整定時間に関して、DRM-DC の性能が PI 制御よりも大幅に優れていることがわかる。ただし、行き過ぎ量は、PI 制御、DRM-DC ともに 100% となっている。図 8 (WAN-S20) から分かるように、転送遅延が増加した時 ($4 \leq t < 10$) に、キュー長が振動的に変化し、安定性が実現できていない。その結果、行き過ぎ量が非常に大きな値となっている。この問題を回避するためには、制御パラメータ (K_P および K_I) をより小さな値に設定すれば良いと考えられる。

以上の結果から、遅延補償器を用いた DRM-DC は、特に転送遅延の大きな WAN 環境において非常に優れた定常特性および過渡特性を示すことがわかる。

7　まとめと今後の課題

本稿では、広域グリッドコンピューティングにおける、動的資源管理方式 DRM-DC を提案した。広域グリッドコンピューティングでは、利用可能資源が時間とともに変動し、なおかつネットワークの転送遅延が無視できないという問題がある。そこで、フィードバック制御における遅延補償器 (スミス予測器) を用いることにより、高い安定性と過渡特性を実現する資源管理方式を提案した。さらに、 Simgrid シミュレータを用いた離散時間シミュレーションを行い、提案する DRM-DC の有効性を検証した。その結果、サイトの処理量の変動に対して、良好な結果を残し提案方式が有効であることが明らかになった。

本稿で提案した DRM-DC では、3 章で説明した資源管理方式の性能指標のうち、特に、定常特性および過渡特性を向上させることを目的としていた。しかし、実際の広域グリッドコンピューティングでは、安定性、堅牢性、柔軟性といった性能指標も重要である。我々が提案する DRM-DC は、転送遅延の影響を打ち消すために、フィードバック制御における遅延補償器を用いている。しかし、遅延補償器の有効性は、制御対象 (サイト) のモデルの正確性に大きく依存している [5]。現実には、サイトの正確な数学的なモデルを構築することは容易ではなく、ある程度のモデル化誤差を許容しなければならない。そこで今後は、提案する DRM-DC の安定性および堅牢性を向上させる手法について検討を行ってゆく予定である。

謝　　辞

本稿で提案した解析手法に対し、有意義な議論をしていただいた、大阪大学大学院情報科学研究科の村田正幸氏、多田知正氏に感謝いたします。

文　　献

[1] I. Foster and C. Kesselman, *The Grid: Blueprint for a New Computing Infrastructure*. San Francisco: Morgan Kauffman, 1999.

[2] I. Foster, C. Kesselman, J. Nick, and S. Tuecke, "The physiology of the Grid: An open grid services architecture for distributed systems integration," Jan. 2002. available at `http://www.globus.org/research/papers/ogsa.pdf`.

[3] I. Foster, A. Roy, and V. Sander, "A quality of service architecture that combines resource reservation and application adaptation," in *Proceedings of the Eight International Workshop on Quality of Service (IWQOS 2000)*, pp. 181–188, June 2000.

[4] F. Berman, G. Fox, and T. Hey, eds., *Grid Computing: Making the Global Infrastructure a Reality*, ch. Condor and the Grid, pp. 945–962. 2003.

[5] W. S. Levine, ed., *The Control Handbook*. CRC Press, 1996.

[6] H. Casanova, "Simgrid: A toolkit for the simulation of application scheduling," in *Proceedings of the First IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid 2001)*, May 2001.

[7] S. Mascolo, "Congestion control in high-speed communication networks using the Smith principle," *Automatica*, vol. 35, pp. 1921–1935, May 1999.

[8] 久保 聖治, 潮 俊光, 村田 正幸, 大崎 博之, "伝搬遅延時間を考慮した ATM の PID 型ふくそう制 御," 電子情報通信学会論文誌, vol. J85-B, pp. 371–380, Mar. 2002.

[9] Globus Project Team, "The Globus Alliance." `http://www.globus.org/`.

[10] H. Casanova, G. Obertelli, F. Berman, and R. Wolski, "The AppLeS parameter sweep template: User-level middleware for the Grid," in *Proceedings of Super Computing 2000*, Nov. 2000.