多目的最適化を用いたグリッドスケジューラシステム

高畑 泰祐

1　はじめに

近年，ネットワークの高速化にともない，大規模計算環境・大規模ストレージ環境として Grid が注目されている．Grid とは地理的に分散したリソースを，利用時にそれらの可用性・能力・性能・コスト・サービス品質などに従って動的に共有・選択・集約することのできる並列・分散処理システムである．このように Grid 環境では，サービスの連携から生まれる相乗効果を最大限に利用することを目指しており，そのためにユーザは分散したデータや計算資源にアクセスし，これらを透過的に利用する能力が必要となる．

また今後 Grid の普及が進むと，ビジネスとしての Grid 利用者が増え，利用コストが問題になってくると思われる．

そこで，こういったヘテロジーニアスな環境における透過的なアクセスを可能とするために開発されたリソースブローカーを利用し，さらにコストとリソース利用時間を最適化するスケジューラを提案する．

2　既存の Grid 環境

Grid 上の資源には，計算サーバからデータベース，科学機器，アプリケーションなどまでが含まれている．このような環境では，リソースの管理ポリシーや所有者が複雑に入り組んでおり，また用いられているミドルウェアも様々である．当然こういったリソースへのアクセスやリソース管理は簡単にはいかない．

そして Grid 環境が整備され普及が進むと，当然リソースに対する課金といった制度が登場してくる．現在でも Sun Microsystems や NTT などが一部課金制度を導入してきており，こういった動きは今後も増えていくものと思われる．しかし，未だ料金プランやリソース能力などのバリエーションは少なく，ユーザにとって選択の余地はあまりない．今後さらにリソースを提供する企業などが増えてくれば，料金プランや CPU パワーにもバリエーションが出てくると予想される．ところがそうなると Grid を利用するユーザにとって問題となるのは，時間と予算の関係である．選択の余地が広がると各ユーザに適したリソースを選ぶことが難しくなってしまう．料金は高くても短時間でジョブを完了したい場合，予算を抑えようと時間を長くかける場合など，既存のスケジューラにはそのようなことを考慮した機能はない．

3　Gridbus Broker を利用したスケジューリングシステムの提案

2 章のような問題に対し，Grid 上のリソースへのアクセスを単純化する目的でリソースブローカーが開発されている．リソースブローカーとはユーザに抽象レイヤを提供することにより，リソースの可用性やアクセス方法，セキュリティやその他のポリシーの心配をすることなくユーザに透過的なアクセス手段を提供するソフトウェアである．そのうちのひとつに Gridbus Broker がある．

Gridbus Broker はジョブスケジューリング機能も備えており，この部分に対してユーザが独自のスケジューリングアルゴリズムを導入することが可能である．ここにジョブ完了の期限と料金コストを考慮したスケジューリングアルゴリズムを実装することで，ユーザの期限と予算を考慮したシステムを実現することができる．

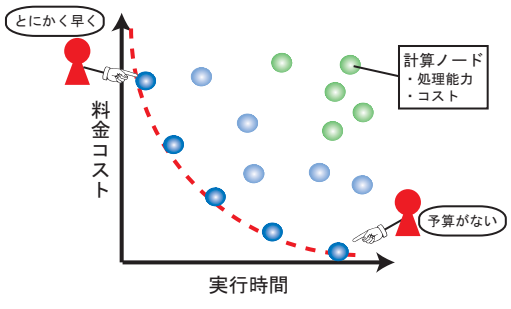

Fig. 1　実行時間とコストの関係

一般的にジョブの完了時間とコストは反比例の関係にあり，たくさんのリモートノードの中から Fig. 1 のようなパレート解となるノードを示し，最終的にはユーザにどのプランで実行させるかを選んでもらうといったシステムを目指す．

4　Gridbus Broker

Gridbus Broker はオーストラリアのメルボルン大学 Grid Computing and Distributed Systems (GRIDS) Laboratory で開発されたリソースブローカーである．Computational Grid 及び Data Grid アプリケーションの両方をサポートするよう設計されており，主な特徴として以下のようなものがある．

4.1　特徴

- 様々なミドルウェアやスケジューラが起動しているリソースへの透過的なアクセスの提供．

  - Globus 2.4，3.2，4.0
  - Alchemi (1.0)
  - Unicore 4.1（試験的）
  - XGrid v.1.0
- ジョブスケジューリング
  あらかじめ数種類のスケジューリングアルゴリズムが用意されている．
- ジョブモニタリング
  ジョブの実行状態を逐次監視することができる．
- 完了したジョブの出力を収集し，ユーザが指定したディレクトリへ保管．
- XPML（eXtensible Parametric Modelling Language）
  XML をベースとしたアプリケーション記述フォーマットとリソース記述フォーマットを用いてジョブを記述．
- クラスタ上のキューシステムのサポート
  PBS や Condor を利用可能．
- Java 実行環境
  コマンドラインや Tomcat によるポータルやポートレットのような Web 利用環境，そしてまたブローカーの提供するサービスから構成された他プログラム内からも利用可能．

4.2 入力ファイル

Gridbus Broker では，以下 3 つのファイルを用いて制御を行っている．

- Broker.properties
  スケジューラの種類指定や，期限，コストなどの設定情報ファイル．
- The Application Description
  コマンドや変数，ファイル転送などの指示が XPML で記述されたファイル．
- The Resource Description
  使用するリソースの情報が XPML で記述されたファイル．Fig. 2 にその例を示す．

```
<?xml version="1.0" encoding="UTF-8"?>
<xgrl xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="../xml/ResourceDescriptionSchema.xsd">

 <credentials id="proxy" type="proxyCertificate">
  <proxyCertificate source="local">
   <local password="*****" />
  </proxyCertificate>
 </credentials>

 <!-- haru.isdl.doshisha.ac.jp -->
 <resource type="compute" credential="proxy" cost="12">
  <compute domain="remote">
   <remote middleware="globus">
    <globus hostname="haru.isdl.doshisha.ac.jp" version="3.2" />
   </remote>
  </compute>
 </resource>

</xgrl>
```

Fig. 2　リソース記述ファイルの例

Fig. 2 にはリソースのミドルウェアやコスト情報などが記述されている．

5　システム構築

今回構築したシステムは，リソース記述ファイルに記述した 4 つのリソースに対してランダムなスケジューリングを実現するものである．

システムを構築した実験環境は Table 1 の通りである．Gridbus Broker をインストールするのは Local のみで，2006 年 5 月現在最新のバージョン 2.4 を使用している．

Table 1　システム環境

Local Side		
CPU	Intel Pentium4	2.8EGHz
Memory		1GB
HDD		160GB
OS	Debian GNU/Linux 3.1 (kernel 2.6.15)	
middleware	Globus Toolkit 4.0 + Gridbus Broker 2.4	
network		
Remote Side		
nodes		4
CPU	Intel Pentium	800MHz
Memory		256MB
HDD		20GB
OS	Debian GNU/Linux 3.1 (kernel 2.6.8-2-386)	
middleware	Globus Toolkit 3.2.1	
network		

6　まとめ

本報告では，現在の Grid 環境ではまだあまり考慮されていない，ジョブの完了期限とコストの問題をスケジューリングするシステムの構築を目指し，まずは Gridbus Broker を導入した．そして Gridbus Broker への期限とコストを最適化する独自スケジューリングアルゴリズムの実装の前段階として，ランダムスケジューリングの実装を行った．

今後は Gridbus Broker API の理解を進め，独自アルゴリズムの実装とリモートサーバを増加させてのシステム構築を目指す．

参考文献

1) The GRIDS Lab and the Gridbus Project
   http://www.gridbus.org/

2) The Globus Alliance
   http://www.globus.org/

3) The Grid Economy Projetc
   http://www.buyya.com/ecogrid/