RCSSディスカッションペーパーシリーズ
第23号　2004年9月

ISSN 1347-636X

Discussion Paper Series
No.23　September, 2004

グリッドミドルウェアCyberGRIPを用いた商用プロバイダ環境での政策グリッド実験

中庭 明子・小橋 博道・鵜飼 康東

RCSS

文部科学省私立大学学術フロンティア推進拠点
関西大学ソシオネットワーク戦略研究センター

Research Center of Socionetwork Strategies,
The Institute of Economic and Political Studies,
Kansai University
Suita, Osaka 564-8680 Japan
URL : http://www.rcss.kansai-u.ac.jp/
http://www.socionetwork.jp/
http://www.policygrid.jp/
e-mail : keiseiken@jm.kansai-u.ac.jp
tel. 06-6368-1177
fax. 06-6330-3304

グリッドミドルウェアCyberGRIPを用いた商用プロバイダ環境での政策グリッド実験

中庭 明子・小橋 博道・鵜飼 康東

RCSS

文部科学省私立大学学術フロンティア推進拠点
関西大学ソシオネットワーク戦略研究センター

Research Center of Socionetwork Strategies,
The Institute of Economic and Political Studies,
Kansai University
Suita, Osaka 564-8680 Japan
URL : http://www.rcss.kansai-u.ac.jp/
http://www.socionetwork.jp/
http://www.policygrid.jp/
e-mail : keiseiken@jm.kansai-u.ac.jp
tel. 06-6368-1177
fax. 06-6330-3304

グリッドミドルウェア CyberGRIP を用いた 商用プロバイダ環境での政策グリッド実験

中庭明子 [1)]、小橋博道 [2)]、鵜飼康東 [3)]
Akiko Nakaniwa, Hiromichi Kobashi, Yasuharu Ukai

1) 関西大学 ソシオネットワーク戦略研究センター
〒564-8680 吹田市山手町 3-3-35 akiko@rcss.kansai-u.ac.jp
2) 株式会社富士通研究所 IT コア研究所 グリッド&バイオ研究部
〒211-8588 川崎市中原区上小田中 4-1-1 h_kobashi@jp.fujitsu.com
3) 関西大学 ソシオネットワーク戦略研究センター
〒564-8680 吹田市山手町 3-3-35 ukai@rcss.kansai-u.ac.jp

概要

本稿では、株式会社富士通研究所が開発したグリッドミドルウェアである CyberGRIP を用いて、商用プロバイダネットワークを介したグリッド環境の構築を行う。商用プロバイダネットワークを介してグリッドを構築するにあたって重要な課題となるセキュリティ上の問題を解決するために、Virtual Private Network(VPN)の仕組みを用いて仮想的に専用ネットワークを構築することにより、セキュリティの高いシステムの実現を目指す。さらに、商用プロバイダネットワークを介したグリッド環境上で実際に計算処理を行い、そのシステム性能の評価を行う。適用事例としては、政策シミュレーション実験を行うこととする。同様の実験をシングルコンピュータにおいて行い、計算時間および仕事量の比較を行うことにより、本システムの性能を検討する。

キーワード

グリッド、CyberGRIP、商用プロバイダ、VPN、政策シミュレーション

Policy Grid Experiment using Grid Middleware CyberGRIP in Internet Service Provider Networks

Akiko Nakaniwa[1)], Hiromichi Kobashi[2)], Yasuharu Ukai[3)]

1) The Research Center of Socionetwork Strategies, Kansai University
Suita, Osaka, Japan
akiko@rcss.kansai-u.ac.jp
2) Fujitsu Laboratories Ltd.
Kawasaki, Kanagawa, Japan
h_kobashi@jp.fujitsu.com
3) The Research Center of Socionetwork Strategies, Kansai University
Suita, Osaka, Japan
ukai@rcss.kansai-u.ac.jp

Abstract

In this paper, we establish a Grid environment using Grid Middleware CyberGRIP which is developed by Fujitsu Laboratories Ltd. over business Internet Service Provider (ISP) networks. In order to overcome the security problem which is one of the important problems in building Grid systems over ISP networks, we apply the mechanism of Virtual Private Networks(VPN) and aim to realize the Grid system with high security. In addition, we evaluate the system performance of this Grid environment over the ISP networks. For this purpose, we carry out a policy simulation experiment over this environment. By carrying out the same experiment in a single computer, we make a comparison of the computing time and the workload and examine the performance of this system.

Keywords

Grid, CyberGRIP, Business Provider, VPN, Policy Simulation

1. はじめに

計算機科学におけるグリッドとは、ネットワークで結ばれ分散された複数の情報技術資源（コンピュータハードウェア・ソフトウェア・OS・ミドルウェア等。以下では物的な資源をリソースとする。）を複数の人で共有し、必要に応じて利用することを可能にする仕組みである。大規模な計算を行う際にも、従来のように高価なスーパーコンピュータを利用する必要はなく、グリッドの仕組みを利用してネットワーク上の空いているリソースを用いることにより、高速に計算を行うために必要な計算能力を確保することが可能である。

グリッドの実現に向け、その技術の研究開発と標準化策定が、現在も並行して進められている。本稿では、その詳細について述べるとともに、株式会社富士通研究所(以下、富士通研究所)で開発を行ったグリッドミドルウェアである CyberGRIP についてその機能を要約する。さらに本稿では、グリッド環境における計算時間と仕事量の分析を行なうことを目的とし、CyberGRIP を用いた、商用プロバイダネットワークを介したグリッド環境の構築を行う。商用プロバイダネットワークを介してグリッドを構築するにあたっては、セキュリティの確保が最も重要な課題となる。本稿では、Virtual Private Network(VPN)の仕組みを用いて仮想的に専用ネットワークを構築することにより、セキュリティの高いシステムの実現を目指す。

さらに、商用プロバイダネットワークを介したグリッド環境上で実際に計算処理を行い、そのシステム性能の評価を行う。適用事例としては、政策シミュレーション実験を行うこととする。これは、世界初の政策グリッド実験となる。同様の実験をシングルコンピュータにおいて行い、計算時間および仕事量の比較を行うことにより、本システムの性能を検討する。

以下に、本稿の構成を示す。まず 2 章において、グリッドの現状として、グリッドの研究開発と標準化の背景について述べる。次に 3 章で、富士通研究所で開発を行ったグリッドミドルウェアである CyberGRIP の機能の要約を行う。4 章では、商用プロバイダネットワークを介して構築したグリッド環境について述べ、本環境における適用事例として行った政策シミュレーション実験とシステムの性能分析結果について 5 章で述べる。最後に、6 章でまとめと今後の課題を述べる。

2. グリッド

2.1. グリッドの概念

グリッドは今日の電力供給網と同じような環境を、コンピュータネットワーク上で実現するものである。今までは利用者が考えていた最適なマシン選択やマシン構成を、グリッドによって自動化しようというものである。自動化することによって、多種多様なリソースがあたかも単一の大きなリソースとして扱うことが可能になる(図 1)。

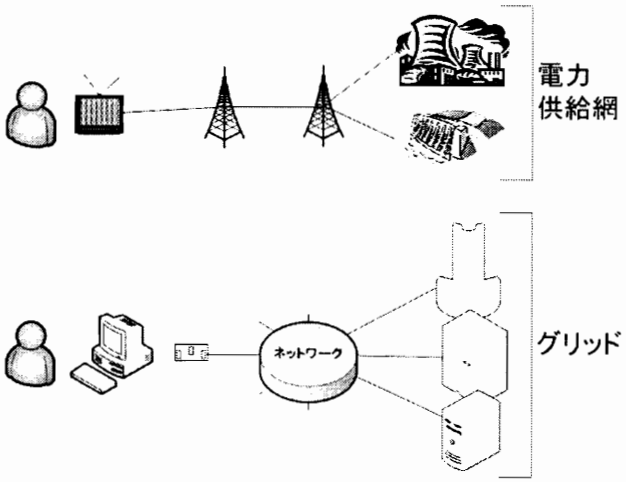

図 1 電力供給網とグリッド[i]

このようにグリッドとは、ネットワークにマシンを接続するだけで、その先にどのようなマシンが存在するかユーザは意識する必要がなく、必要なときに必要な分だけマシンを使えるネットワーク環境のことである。

グリッドの言語の起源はアメリカの Power Grid に由来している。一番初めに広域分散システムにグリッドという言葉を使ったのは、90 年代前半に NASA の傘下にある Ames、Langley、Glenn などの研究所にあったスーパーコンピュータをつなげた IPG:Information Power Grid であったと言われている。グリッドの概念がかなり前から存在していたにも関わらず、1990 年代後半から 2000 年にかけて騒がれ始めたのは、その当時リソースの処理能力やネットワーク帯域がグリッドの実用化には程遠いものであったことがあげられる。リソースは単一のアプリケーションを動かすだけでも中央処理装置(Central Processing Unit、以下 CPU)[1]を最大限利用する必要があったり、ネットワークも少数の文字しか送ることしか考えていなかった時代にグリッドは広まることはない。しかし今日では、ムーアの法則やギルダーの法則[2]が示すとおり、急激なスピードで処理能力やネットワーク帯域は向上している。その向上により、グリッドを実現できる環境が整ってきたことが、グリッドが盛んに研究されるようになった一つの要因である。

また注意したいのは、グリッドはいきなり現れた技術ではなく、今までスーパーコンピュータや PC クラスターで培われた技術を応用して誕生したものである。さらにグリッドがあればスーパーコンピュータや PC クラスターが不必要になるわけではない。グリッドにも得手不得手がある。断続的に通信が発生するようなシミュレーションにはグリッドは向かない。そのようなシミュレーションにはスーパーコンピュータや PC クラスターが必

[1] CPU：コンピュータの頭脳。コンピュータ上での処理を行うユニット。
[2] ムーアの法則とギルダーの法則：ムーアの法則とはインテルのゴードン・ムーア氏が唱えた「CPU の処理スピードは 18 ヵ月ごとに 2 倍になる」という説である。実際にムーア氏が言ったのは「24 ヶ月ごと」だが、最近では「18 ヶ月ごと」が定着している。ギルダーの法則とは、そのムーア法則のより速いスピードでネットワーク帯域が拡大していくという説である。現に、この二つの予言通りに処理能力・ネットワーク帯域は向上している。

要になる。ただ、最終的にすべてのリソースがグリッドに取り込まれることになれば、通信が大量に発生するジョブは自動的にスーパーコンピュータや PC クラスターに割り当てられることになるであろう。

2.2. グリッドの現状

グリッドは世界中でさまざまな団体、企業が研究開発を行い、実際に運用している例もある。

富士通株式会社(以下、富士通)では Systemwalker CyberGRIP と呼ばれるグリッド対応製品を開発し、2004 年 9 月に市場にリリースする。Sytemwalker CyberGRIP は後述する CyberGRIP の製品版である。CyberGRIP は社内適用されており、すでに業務効率の大幅改善に貢献している[ii]。具体的には社内の CAD 部門のシミュレーションに CyberGRIP は利用されている。その結果、シミュレーション期間を 75%、工数を 66%削減することに成功した。また、富士通は超高速コンピュータ網形成プロジェクト(NAREGI:National Research GRID Initiative)[iii]や VizGrid[iv]、ビジネスグリッドコンピューティング研究開発事業[v]などのプロジェクトにも積極的に参加し、大きな役割を果たしている。

富士通とならんで積極的にグリッドの研究開発を行っている International Business Machines Corporation(以下、IBM)では、グリッドミドルウェアのデファクトスタンダードとして認知されているオープンソースである Globus Toolkit[vi]の開発を Globus Alliance と共同で行っている。さらに業務のグリッド適用も積極的に行っている。例えば日本 IBM 株式会社は株式会社ニッセイ基礎研究所と共同研究を進め、「市場リスク」と「信用リスク」を統合的に計測し、金融機関のリスクを正確に評価するリスク統合管理システムを、グリッド技術に対応させた[vii]。

富士通や IBM は基本的に閉じた社内のネットワーク環境でのグリッドを中心にビジネスを進めているが、NTT 西日本株式会社は国立遺伝学研究所と共同でフレッツユーザの PC をグリッドでつなげて DNA 解析を行う研究を開始している[viii]。海外では、地球外生命体を探すプロジェクトである SETI@Home[ix]や、たんぱく質のおりたたみ構造を解析するシミュレーションを行っている Folding@Home[x]がインターネットを介した一般消費者のパソコンを使ったグリッドでは有名である。

また、Global Grid Forum (GGF)[xi]では、グリッドの標準化作業を行っている。特にグリッドと Web サービスとの融合を目指した Open Grid Service Architecture (OGSA)について活発な議論がなされている。

この GGF と密接な関係にあるのが、最初のグリッドミドルウェアを開発した Globus Alliance である。Globus Alliance が開発した Globus Toolkit は世界中で利用されており、もっともユーザの多いグリッドミドルウェアであろう。また Globus Toolkit のバージョン 4 が OGSA 対応の最初のグリッドミドルウェアになる予定である。

Globus Toolkit とならんで有名な UNICORE[xii]は、欧州にある Fujitsu Laboratories of Europe が中心になって開発された。Graphical User Interface(GUI)や Windows 対応といった点で

Globus Toolkit よりも優れている。NAREGI でも標準のグリッドミドルウェアとして利用されている。

3. CyberGRIP の機能

3.1. CyberGRIP の概要

今日では、民間企業(仮)各事業部または各部ごとに Linux や Solaris などの UNIX マシンで構成されるローカルのバッチシステムをいくつか所有し、シミュレーションなどの計算を行いたいユーザのほとんどは自身が所属している部署のバッチシステムのみを利用している。しかし、それぞれの部署において、バッチシステムをフル稼働させている時期はまちまちであり、個々で見ると使われているときと使われていないときの差が大きく非効率であることが多い。また、たいていの場合、忙しい時期に隣の部署の空いているバッチシステムを利用したいと考えたとしても、そこには目には見えないが壁があることが多く、利用することは困難である。

CyberGRIP[xiii]は、それら既存のバッチシステムを仮想的に一つのバッチシステムとして利用できるようなフレームワークを提供するグリッドミドルウェアである。これにより、ユーザは組織の壁を越えて、遊休のマシンを利用することが可能になり、バッチシステムとしては、全体の利用効率をあげることができる。また、CyberGRIP ではオフィスの Windows マシンも取り込むことを可能にしている。これらは 1990 年代前半のスーパーコンピュータ並みの演算能力を持っているにも関わらず、既存のバッチシステム以上に利用率が低い。さらに、優先度の高いジョブが後から投入された場合、その緊急度によっては前に投入されていたジョブを追い越し、先に実行することも可能である。これにより、緊急時のために敢えて使わないで温存しておいたマシンも有効利用できる。

CyberGRIP は以下の 4 つの機能から構成されている。

a) OJC (Organic Job Controller)
b) GRM (Grid Resource Manager)
c) SRMD(Site Resource Manager Dispatcher)
d) SRM(Site Resource Manager)

全体図は図 2 の通りである。

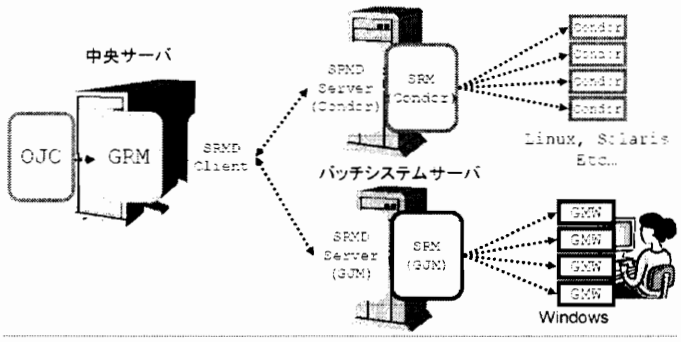

図 2 CyberGRIP

次に各モジュールについて説明をする。

3.2. SRM

CyberGRIP では既存のバッチシステムを総称して、SRM と呼ぶ。CyberGRIP では様々な部門や地域に散らばっている様々な種類のマシンが存在する環境を想定している(図 3)。

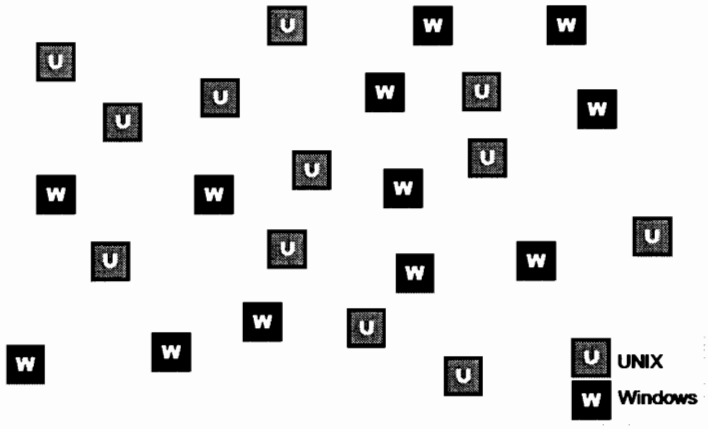

図 3 分散した様々なマシン

これらのマシンは物理的にはネットワーク回線で接続されているが、個々が独立して動作しており、グリッドのように協調して動作するようにはなっていない。

まず始めにこれら分散したマシンを大まかにまとめる(図 4)。このまとめるものを CyberGRIP では SRM と呼び、Linux、Solaris 用には Wisconsin 大学が開発した Condor[2]を、Windows 用のマシンには富士通研究所が開発した GJM(Grid Job Manager)を利用することが可能である。

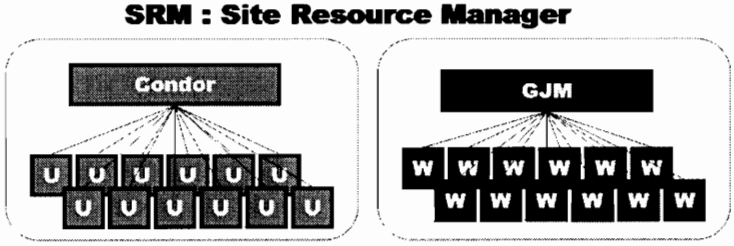

図 4 SRM でまとめる

ただ Windows 利用者はマシンをメールや文章作成などのツールとしてコンピュータを利用している場合が多く、複雑なソフトのインストールには不慣れな方が多い。そのため、GJM は Condor のような難しい仕組みを理解しなくても簡単に CyberGRIP で実現されたグリッドに参加できるように、また普段の業務の邪魔をしないような工夫がなされている。Windows ユーザは GMW(Grid Mediator for Windows)という小さなサービスを自分のマシンにインストールするだけで、CyberGRIP へ参加することができる。GMW は一般ユーザに使いやすい GUI(Graphical User Interface)を装備しており、その上でさまざまな設定を行う

ことができる。例えば、なんらかの理由でグリッドのジョブを実行したくないとき、完全に拒否をしたり時間指定で拒否をしたりすることができる。また、メモリ利用量やディスク使用量がある閾値を越えるとジョブを停止するように GUI 上から設定することも可能である(図 5)。停止されたジョブは他のマシンで実行されることになる。

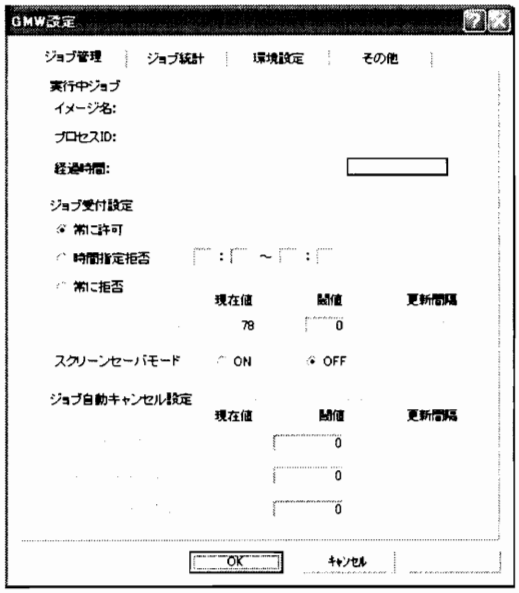

図 5 GMW の GUI

3.3. SRMD

SRMD は SRM を仮想化し、複数のバッチシステムをあたかも一つのバッチシステムのように見せる役割を果たすモジュールである。

SRMD では SRM の仮想化のために、以下のような 3 つの統一したインターフェイスを提供する。

a) 情報系
   - ホストの情報
   - ジョブの情報

b) 予約系(チケットの操作)
   - チケットの予約、開放、削除
   - 予約の停止、再開

c) ジョブ系(ジョブの操作)
   - ジョブの投入、停止、削除
   - 実行結果の回収

統一とはジョブの投入・削除・状態監視のような一般的なコマンド名、引数を同じにするだけではなく、動作や得られる情報も統一する。それぞれの SRM が提供する情報の差異をなくし、統一された単位、意味に変換する。これにより、SRMD を通して全ての SRM

が仮想的にあたかも一つの SRM かのように扱うことができる。

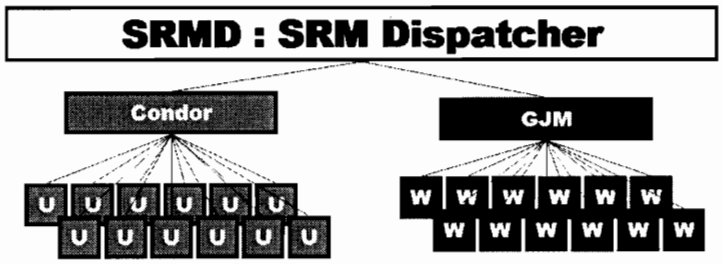

図 6 SRMD

また、SRMD は"予約"の仕組みを実現し各 SRM を管理している。SRMD での予約とは、"ジョブを実行する前に使用時間を決めて、自分専用にグリッド上にあるマシンを確保すること"である。利用したいマシンを予約することにより"チケット"を得る。その後、ジョブを実行する際、チケットをジョブとセットにして SRMD にジョブ実行を依頼する (図 7)。

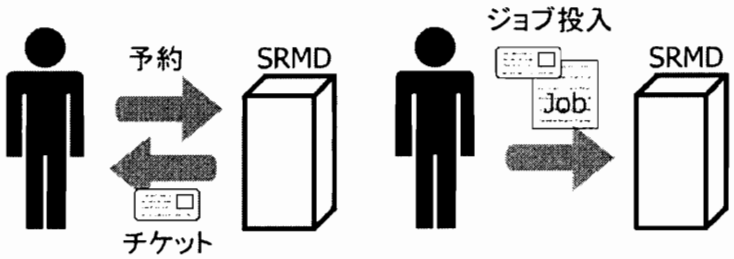

図 7 予約の動作

この"予約"は MPI(Message Passing Interface)のように、マシンが指定した台数分揃わないとジョブを実行できないようなプログラムに対して大きな力を発揮する。つまり、予約ジョブを実行するだけの台数が揃わなくても、確保できる分だけを予約しておくことで、横から来たジョブによって、いつまでも台数が揃わず待たされるといった症状を回避することができる。

3.4. GRM

GRM は SRMD で仮想化された複数のバッチシステムを統合しユーザのジョブを実行させる。このため、GRM は以下の 7 つの機能を持つ。

a) 複数 SRM の統合

SRM 配下のリソースをその名前と SRMD の名前の組で管理する事により、複数の SRM に属するリソースを一意に扱う事ができる。

b) 利用可能リソースの情報取得

定期的に SRMD 経由でリソースの情報を取得し、GRM 内部で一覧を作成する。

c) ユーザジョブのキューイング

ユーザが投入したジョブをキューイングしユーザ間の公平を保ちジョブを順番に実行す

る。

d) リソースとジョブのマッチング

ジョブに適したリソースを GRM 内部に保持している利用可能なリソース一覧から選択し、そこでジョブを実行させる。

e) ジョブ実行状態の監視

定期的にジョブの動作状況を監視し、終了している場合には、その実行結果を回収する。

f) ジョブ実行異常時のジョブ再実行

ジョブの動作状況の監視によって、ジョブが異常終了しているのが検出された場合ジョブをキューに戻し、他のリソースを割り当て、再実行させる。

g) SRM 毎の性能マッピング

SRM の利用パターンによって、ジョブの要求性能とリソースの性能の間のマッピングが変更可能。これにより、停止頻度の高い Windows マシンには比較的短時間のジョブを割り当て、UNIX マシンには長時間ジョブを積極的に割り当てる事が可能である。

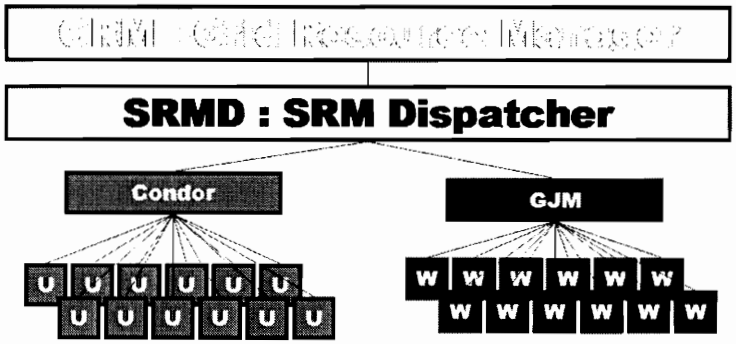

図 8 GRM

3.5. OJC

OJC は大量のジョブを CyberGRIP 上で投入管理するための機能を提供する。

グリッドを利用しないと、ユーザは限られた計算機資源のなかで納期などの締切りを守るため、必要最低限のシミュレーションしか試すことができなった。しかし、グリッドでは利用できるマシンが増えるため、ユーザは今まで以上のシミュレーションを行うことが可能になり、製品の品質向上を達成することができる。しかし、品質向上のためにより多くの計算をすることはより多くの計算結果を受け取り、解析することにほかならない。そのため従来は、計算結果を解析した後、満足いかない結果が出た部分があり、その部分だけを計算し直そうとしても、大量の計算結果に埋もれてしまい、どの結果がどれと関係しているかを把握することは困難であった。

OJC には、以下のような 3 つの主要な機能があるために大量のジョブを投入・管理に必要な工数を大幅に減らすことができる。

a) ジョブフローの記述

OJC が解釈するジョブの終了待ち合わせとジョブ投入を記述することができる。これにより、ジョブフローに基づいた柔軟なジョブ生成、終了後の纏め上げ、メールの送付、実行結果を用いたリアルタイムな更新などができる。また、外部の最適化エンジンと組み合わせ、効率的なジョブ生成によるスマートなグリッドを実現できる。

b) ジョブの一括削除、部分削除および再投入

OJC はジョブが投入済みの場合は何度実行してもジョブを投入しないように設計されている。このため、やり直したいジョブの状態を未投入状態にすることで、必要なジョブのみを簡単に再実行することができる。

c) ジョブの状態表示

OJC はジョブフローにしたがってジョブをツリー状に配置し、ジョブの進行状況を色分けして表示する。これによりジョブ同士のつながりを簡単に把握することができる。

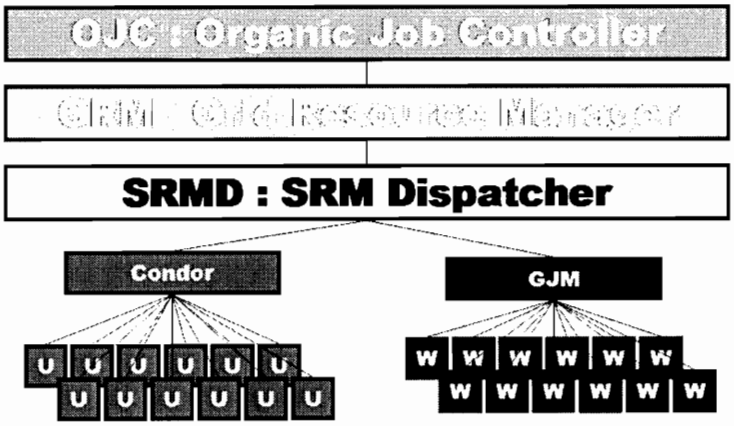

図 9 OJC

4. 商用プロバイダを介したグリッドシステムの構築

本稿では CyberGRIP を富士通研究所と関西大学ソシオネットワーク戦略研究センター（The Research Center of Socionetwork Strategies、以下 RCSS と省略する。）に配置し、商用プロバイダネットワークを介して接続し、グリッドシステムを構築した。その環境について述べる。この環境を"グリッド効率比較システム（商用プロバイダ）"とし、以下、本稿では、"商用プロバイダ実験環境"と呼ぶこととする。

4.1. セキュリティ上の課題

商用プロバイダを介した環境を利用するにあたり、一番気を付けなければならないのはセキュリティである。通常、グリッドの実験は Firewall の内側にある一定のセキュリティが保たれたネットワークや専用のネットワークで行われることが多い。なぜなら不特定多数の人間が利用するネットワークにおいて、情報のセキュリティを保つことが困難な場合が多いためである。

しかし、例えば SETI@Home や Folding@Home といった商用プロバイダネットワークを介してグリッドを実現しているプロジェクトも存在する。これらのプロジェクトはボランティアの方々の協力を得て、インターネット上にある大量のマシンをつなぎ合わせ、大規模計算を実現している。ピーク時には何億円もするスーパーコンピュータに匹敵、あるいは凌駕するような性能を発揮している。

ただ、これらのプロジェクトが成功したのには、いくつかの特別な事情がある。例えば SETI@Home で用いられているデータは、プエルトリコにあるアレシボ電波望遠鏡[xiv]で得られた公開されているデータである。さらに解析した結果もほとんどのケースが意味のないものであるため、データ管理に必要以上に神経質になることはない。Folding@Home の場合もよく似ており、一つのマシン得られる解析データは全体の中のほんの一部分だけであるため、それを見ても全く意味がないということがあげられる。

こういったことから、これらのプロジェクトは比較的容易にインターネット上のマシンにジョブをばらまくことができる。しかし、このようなインターネット上のマシンにジョブを投げることができるケースは珍しく、通常は情報漏えいなどの観点から不可能である場合が多い。

本稿では重要なデータが入ったマシンをリソースとして利用することが前提になっているため、通常以上のセキュリティ対策を施す必要がある。また、現段階の CyberGRIP では、マスターとリソースの間に Firewall がないことが前提になる。

4.2. セキュリティ上の問題の解決方法

この実験では

a) リソースをネットワーク外部のアタックから防御する

b) 富士通研究所と RCSS 間に Firewall をなくす

の 2 点をクリアーするために Virtual Private Network:VPN を使い、仮想的に専用ネットワークを構築する方法をとった。

これにより、富士通研究所と RCSS 間の通信は全て公開鍵暗号方式で暗号化され、情報は保護されている。また、富士通研究所と RCSS に置かれたそれぞれの VPN 装置は、それぞれの内側にあるマシン同士の通信しか許可しない。つまり、VPN の内部から外部に出ることが許されていない。これによって、不正にデータが持ち出させることを不可能にしている。もちろん、いかなる外部からのアクセスを許可しておらず、盗み見などはできない設定になっている。

このようにして、富士通研究所と RSCC 間は高いセキュリティ機構に守られた安全なネットワークを実現し、その中でグリッド実験を実施している。

4.3. システム構成

図 10 に商用プロバイダ実験環境のシステム構成図を示す。

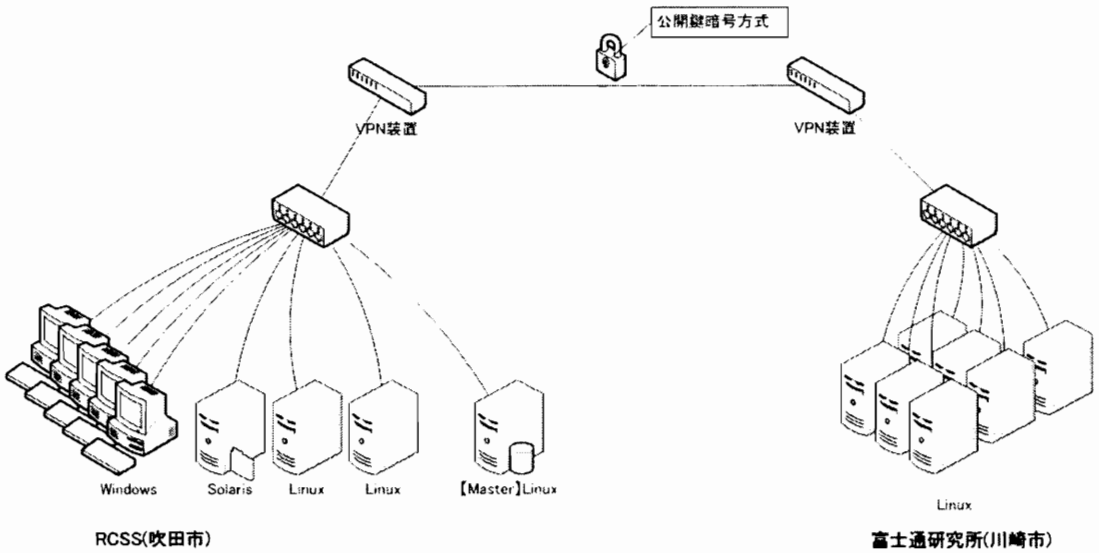

図 10 システム構成図

商用プロバイダ実験環境では、さまざまな機種や OS が混在する実際のネットワークと同様の環境を目指した。結果、OS は Linux、Solaris と Windows の 3 種類を、CPU アーキテクチャとしては Intel アーキテクチャの Pentium と Xeon ならびに SPARC アーキテクチャの UltraSPARC をグリッド環境に取り込んだ。
マシンリストを表 1 に示す。

表 1 マシンリスト

設置場所	OS	CPU	マシン数
富士通研究所	Linux	Xeon 3.2GHz x 2	3
	Linux	PentiumIII 1.4GHz x 2	4
RCSS	Linux	Pentium4 3Ghz x 1	1
	Linux	Pentium4 1.8GHz x 1	2
	Solaris	SPARCIII 966MHz x 2	1
	Windows	Pentium4 3Ghz x 1	5
総CPU数			24

5. 適用事例とシステム分析

5.1. 適用事例

本稿では、商用プロバイダ実験環境に対する第一の適用事例として、世界初となる政策グリッド実験を行った。具体的な実験対象は託児所設置シミュレーション[xv]である。

本シミュレーションでは、エージェント間の相互依存関係に着目したマルチエージェントモデル[3]を用いて、女性の労働市場への参加率を増加させるために託児所をどのように配

[3] マルチエージェント：本稿では，主体性をもち自律的に行動する構成要素をエージェントと定義し，エージェントが相互に依存・作用しあいながら多数共存する環境をマルチエ

置することがのぞましいかという、託児所配置問題を解くことを目的としている。託児所の配置を決定するために、本シミュレーションでは、エージェントの効用差関数や託児所選択関数などを含む非常に多くのパラメータの組合せをもったマルチエージェントモデルを構築する必要がある。このような大規模なマルチエージェントシミュレーションを実行するためには、高速な計算実行環境が必須となる。そこで、本稿では、商用プロバイダ実験環境において本マルチエージェントシミュレーション実験を行い、より高速に大規模な計算を行うことができることを検証する。

5.2. OJC スクリプト

ここでは本シミュレーションで実際に使用した OJC スクリプトを取り上げ、そのプログラム構造を説明する。OJC スクリプトはスクリプト言語として広く知れ渡っている Perl 言語やシェルスクリプトとほぼ同様の文法を用いているため、それらの言語を知っていれば、容易に OJC スクリプトは記述することが可能である。

OJC スクリプトを説明する前に、あるジョブを、そのジョブを実行するために必要なパラメータが書かれたファイルを一行ずつ読み込んで、逐次実行する Perl スクリプト[4]を考える。

```
#!/bin/perl
$parameterFileName = “parameter.txt”;  ←パラメータリスト

open(PARAM, $parameterFileName);  ←パラメータファイルを開く

while(<PARAM>){  ←リストの行数だけ繰り返す
    $masParam = $_;
     system( “mas_DCcenter $masParam” );  ←読み込んだパラメータをジョブに与えてジョブを実行する

}
close(PARAM);
```

図 11 パラメータファイルを読み込んでジョブを実行する Perl スクリプト

図 11 の Perl スクリプトはパラメータファイル”parameter.txt”を読み込み、一行ごとにジョブ”mas_DCcenter”に与えて実行するものである。この Perl スクリプトを実行すると、パラメータリストの分だけのジョブが一台のマシン上で”mas_DCcenter”が逐次実行される。

ージェ
ントと定義する。
[4] Perl：フリーのスクリプト言語。CGI など、世界中で広く利用されている。

次に、この Perl スクリプトを OJC スクリプトに書き換える。

```
config {{
    --sleep 3
    --abortcheckinterval 10
    --shelltype perl
}}

top {{
    $parameterFileName = “parameter.txt”;  ←パラメータリスト

    open(PARAM, $parameterFileName);  ←パラメータファイルを開く

    while(<PARAM>){  ←リストの行数だけ繰り返す
        $masParam = $_;
        do_job {{ case=$i}} {{
            header {{ deploy=mas_DCcenter OS=la_Gmw,la_Linux }}
            system("mas_DCcenter $masParam");
        }}
    }
    close(PARAM);
}}
```

図 12 パラメータファイルを読み込んでジョブを実行する OJC スクリプト

図 11 と図 12 の違いは BOLD 書体のところである。

既存の逐次処理を行う Perl スクリプトを OJC スクリプトに書き換えるには、”Config”と”Top”と呼ばれる設定ブロックが、Perl の”#!/usr/bin/perl”と入れ替わる。そして、ジョブを走らせる”system”の前後に”do_job”パートを書くだけである。これで、”mas_DCcenter”がグリッド上の様々なマシンに分散され、実行される。

このように、Perl に慣れ親しんだプログラマーであれば、誰でも簡単に OJC スクリプトを作成でき、容易にグリッド環境にジョブを投入することができる。

5.3. システム性能分析

今回の商用プロバイダ実験環境の性能を検討するために、シングルコンピュータで計算を行った場合の計算時間との比較を行った。

はじめに、商用プロバイダ実験環境およびシングルコンピュータで託児所配置シミュレ

ーションを行った場合の計算時間を図 13 に示す。ここで、比較対象となるシングルコンピュータのスペックは以下の通りである。

- CPU:：Pentium(R)4, 3.20GHz
- メモリ：1GB DDR-SDRAM メモリ
- OS：WindowsXP

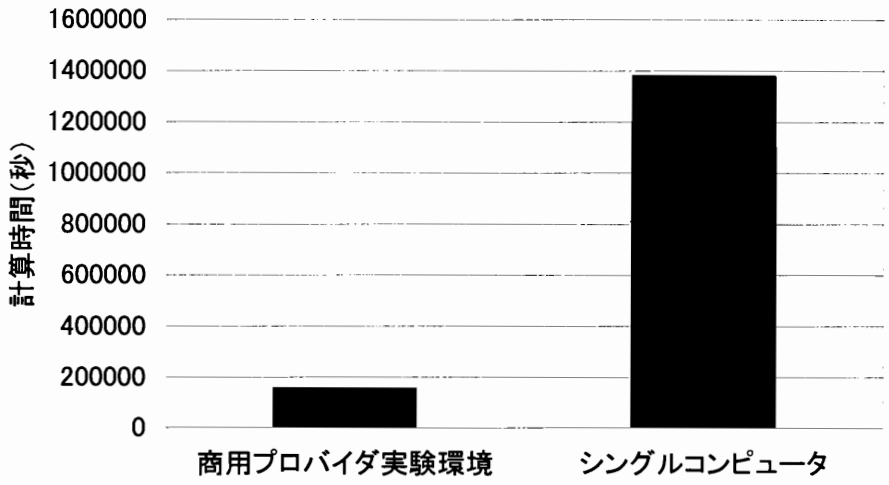

図 13 商用プロバイダ実験環境とシングルコンピュータの計算時間比較

図 13 から明らかなように、商用プロバイダ実験環境で計算を行った場合、シングルコンピュータで計算を行った場合と比較し、飛躍的に計算時間を削減できている。商用プロバイダネットワークを介して複数のリソースで分散させて計算を行うことにより、計算時間を大幅に抑えることができることがわかる。さらに多くのコンピュータをリソースとして用いることにより、さらなる計算時間の削減が期待できる。

次に、図 14 に商用プロバイダ実験環境およびシングルコンピュータで実際に行った総仕事量（計算時間×総クロック数）の比較結果を示す。 単純に計算時間とクロック数を掛けることが正確な仕事量となるとは言いがたいが、ひとつの目安になるので、本稿では、これを尺度として仕事量の比較を行うこととする。

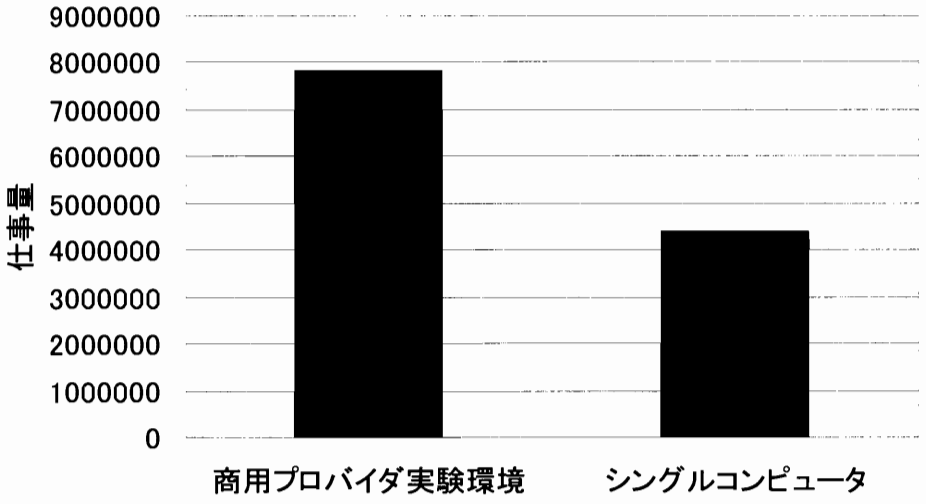

図 14 商用プロバイダ実験環境とシングルコンピュータの仕事量比較

図 14 から、商用プロバイダ実験環境で計算を行った場合には、シングルコンピュータで計算を行った場合と比較して、総仕事量が大きくなってしまっていることがわかる。これは、CyberGRIP を導入してグリッドシステムとして計算を行なっていることによるオーバーヘッドによるものと考えることができる。

以上の結果から、グリッド環境においては、計算時間と仕事量のトレードオフが存在することが推測できる。したがって、計算時間の短縮を最優先とする政策課題に対しては、本稿のグリッド環境は有効に機能するであろう。しかしながら、計算時間の短縮よりも計算に要する費用の削減が優先される政策課題に対しては、本稿のグリッド環境はかならずしも有効ではない。

6. 結論

本稿では、グリッド環境における計算時間と仕事量の分析を行なうことを目的とし、商用プロバイダネットワークを介したグリッドシステムの構築を行った。富士通研究所が開発したグリッドミドルウェアである CyberGRIP を神奈川県川崎市の富士通研究所と大阪府吹田市の RCSS に配置し、商用プロバイダネットワークを介して接続し、グリッド構築した。商用プロバイダネットワークを介することで懸念されるセキュリティの課題を克服するために、ここでは、VPN を用いて仮想的に専用ネットワークを構築する方法をとった。

商用プロバイダネットワークを介したグリッド環境に対する第一の適用事例として、本稿では、世界初の政策グリッド実験となる、託児所配置シミュレーション実験を本システム上で行った。本システムの性能を検討するために、さらに同様の実験をシングルコンピュータにおいて行い、計算時間および仕事量の比較を行った。その結果、商用プロバイダ実験環境で計算を行った場合、シングルコンピュータで計算を行った場合と比較して、飛躍的に計算時間を抑えることができることを示した。一方、仕事量においては、商用プロ

バイダ実験環境で計算を行った場合、シングルコンピュータで計算を行った場合よりも、仕事量が大きくなることが示された。これは、CyberGRIP を導入してグリッドシステムとして計算を行っていることによるオーバーヘッドによるものと考えられる。このように、グリッド環境においては、計算時間と仕事量のトレードオフが存在することが推測される。したがって、どのような政策課題のシミュレーションが、グリッド環境に適合しているかの検討が、今後の重要な研究課題である。

なお、本稿においては、システム性能に関する検討のみを行ったが、今後、さらに経済的な尺度を導入し、グリッドの構築における費用と性能の関係を調査することによって、経済的観点からみて最適なグリッド環境をモデル化することが可能になるであろう。

参考文献

[i] 「今日から始めるグリッドコンピューティング –Globus Toolkit 入門-」
小橋博道、UNIX USER 2003 年 10 月号、pp.50-73
[ii] 「グリッド環境「CAD-Grid」構築と移動通信シミュレーションへの適用」
山下智規，中村武雄，上田晴康，今村信貴，岩松隆則，斉藤直之
第 6 回問題解決ワークショップ論文集，pp.31-36、2003 年 8 月
[iii] http://www.naregi.org/
[iv] http://www.vizgrid.org/
[v] http://www.ipa.go.jp/software/bgrid/index.html
[vi] http://www.globus.org/
[vii] http://www.ibm.com/news/jp/2004/06/06281.html
[viii] http://www.ntt-west.co.jp/news/0402/040203.html
[ix] http://setiathome.ssl.berkeley.edu/
[x] http://folding.stanford.edu/
[xi] http://www.gridforum.org/
[xii] http://www.unicore.org/
[xiii] 「グリッドミドルウェア CyberGRIP による組織を横断した計算機利用」
小橋博道、清水智弘、今村信貴、上田晴康、野口弘、山下智規、門岡良昌、宮澤君夫
第 7 回問題解決環境ワークショップ、2004 年 10 月発表予定
[xiv] http://www.naic.edu/
[xv] “Political Multi-Agent Simulation with Grid Computing”, Tadahiko Murata, Hiroko Kitano, Yoshimasa Kadooka and Ysuharu Ukai, RCSS Discussion Paper Series No.17 (August, 2004)