ハイパフォーマンス
コンピューティング 91−29
（２００２．８．２３）

Campus Grid における負荷状況を考慮したノードの選抜方法

蟻川 浩 †　　砂原 秀樹 ‡

† 奈良先端科学技術大学院大学 情報科学研究科
‡ 奈良先端科学技術大学院大学 情報科学センター

概 要

キャンパスネットワークに接続されている計算機を利用した仮想並列計算環境である Campus Grid が注目されている。本稿では、Campus Grid を構成する各ノードの人為的要因による負荷変動に着目し、平均・分散モデルによるノードの特徴抽出および Grid 全体を捉えるためのノード選抜のモデル化を提案する。その有用性について NAS Parallel Benchmarks(NPB) version 2.3 を用いて評価した。

A Selection Scheme of the Node to consider a Load Factor for Campus Grid

Hiroshi Arikawa†　　Hideki Sunahara‡

†Graduate School of Information Science, Nara Institute of Science and Technology
‡Information Technology Center, Nara Institute of Science and Technology

Abstract

Recently, Campus Grid which makes using many computers connected to the campus network is watched. Increasing the performance of Campus Grid requires changing the state of load average due to a human action by each node. In this paper, we propose the state of a node is defined using mean-variance model and the scheme of selection a node by using the method similar to the allocation problem, and confirmed the usefulness of the scheme using NAS Parallel Benchmarks(NPB) version 2.3.

1　はじめに

インターネットのような広域ネットワークに接続された計算資源を 1 つの計算基盤として利用できる、Grid と呼ばれる高性能広域分散システムの研究が盛んに行われている [1, 2, 3]。Grid は構成する計算機のアーキテクチャ、ネットワーク、利用する内容によって様々な形態がある。本稿では今後もっとも注目されると思われる Campus Grid に焦点を当てる。

Campus Grid は大学内に存在する複数の異なるアーキテクチャの計算機とキャンパスネットワークで構成された計算環境で、高性能の計算環境を安価で実現できる。最近では、1 人 1 台の計算機を割り当てる状況になりつつあるた

め、同一性能の計算機が大量に存在することも珍しいことではない。従って、キャンパスネットワークとそれに接続されている計算機を用いて、Beowulf 型クラスタ [4] を構築することが可能である。

ところで、Campus Grid を構成している各ノードには利用者が存在する。メールの読み書き、Web 閲覧、論文やレポートの作成といった日常的な処理、プログラム作成および実行、スクリプト言語を使ったデータの処理などというように様々な処理にノードは使われている。この状況下で Campus Grid を利用した並列実行プログラムを処理することを考えた場合、各々のノードの利用状況の違いからプログラムの処理時間にゆらぎが生じる。

本研究では Campus Grid において並列実行プログラム処理時間の結果にゆらぎが生じるのは Grid を構成する各ノードの人為的要因による負荷変動が原因であることに着目する。各ノードの負荷変動を過去の履歴から特徴抽出し、その情報をもとに予測した結果を利用できればプログラムの処理時間のゆらぎを低減させることが可能となる。そのためには Grid 全体を捉えるためのモデルの提案およびそれを踏まえたジョブ割り当てアルゴリズムが必要である。

本稿では、各ノードの負荷状況に対して平均値と分散値によって表すことで各ノードの特徴抽出を行い、その結果を利用した配分問題モデルを適用することで、負荷状況を考慮したノードの選抜方法を提案する。また、本稿で提案する方法の有用性を確認するために、NAS Parallel Benchmarks[5] version 2.3 の EP, CG ベンチーマークツールを用いた実験を奈良先端科学技術大学院大学 情報科学研究科の情報基盤設備の一部を用いて行い、その結果について報告する。

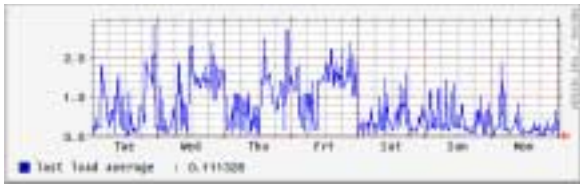

図 1: あるノードの 1 週間の load average の変化 (横軸:時間、縦軸:load average の値)

2 ノード選抜モデル

2.1 平均・分散モデルによるノードの特徴抽出

図 1 は大学内のあるノードにおける 1 週間の負荷の状態をグラフに示したものである。この図では全体的に平日午後に負荷が高く、休日は負荷が低いといったように、ノードの負荷の時間的推移は利用者の生活パターンといった人為的要因によって影響していることがわかる。つまり、プログラムの処理時間のゆらぎを小さくするためには、ノードを監視して現状における負荷の小さいノードを選択するだけではなく、過去の情報を利用してノードの負荷変動のパターンを考慮したノード選択のしくみが必要である。

本稿ではある期間において、どれだけの負荷を生じ、かつどの程度負荷の変動が大きいかを示すことでノードの負荷の特徴抽出を行う。これらを示すパラメータとして、本稿では前者は負荷の平均値、後者は負荷の分散値を用いる。

2.2 ノード選抜のモデル化

ここでは、2.1 節で説明した各ノードの特徴抽出の結果から、Grid 全体を捉えたノード選抜のモデルについて説明する。

2.2.1 ノード選抜における条件

Campus Grid には計算機アーキテクチャやネットワーク機器によって様々な形態が存在する。本稿では、ホモジニアスな Campus Grid

に焦点を当てることにする。

以後の話を明確にする上で、以下でホモジニアス Campus Grid の条件を定義する。

ホモジニアス Campus Grid の条件

- Grid を構成するノードの処理能力はすべて同一のものであること。言い替えると、ハードウェア仕様がすべて同一の計算機で構成されていること。
- ノードで動作するオペレーティングシステムはすべて同一であること。また、各ノードで使われる並列実行ライブラリも同一のものであること。
- ノード間同士のネットワーク帯域はすべて同じであること。

また、Campus Grid 上でプログラムを実行することを想定した場合、並列実行するプログラムによってはネットワークの帯域幅がボトルネックとなり、ノードの台数を増やしても処理時間が短縮されない場合がある。

そこで、本稿ではネットワーク帯域による制限を考慮し、実行時に必要とされるノードの数を予めわかっていることとする。

2.2.2 ノード選抜規準

数多くのノードの中からプログラム処理時間を短縮を行うためのノード選抜を行うためにはなるべく負荷の小さいノードから Grid を構成することが望ましい。また、プログラム処理時間のゆらぎを小さくするためのノード選抜を行うためにはなるべく負荷の変動が小さいノードで Grid を構成するのが望ましい。

そこで、本稿では以下の規準に従ってノード選抜を行うこととする。

1. 負荷の平均値が小さいものを優先する
2. 負荷の平均値が同じ場合には負荷の分散値が小さいものを優先する

2.2.3 ノード選抜のための評価式

2.2.2 節で示した規準を踏まえ、各ノードの過去の負荷変動の特徴抽出の結果および現在の負荷の値を用いたノード選抜の評価式を次式を示す。

$$\begin{bmatrix} L \\ E_1 \\ S_1 \\ \vdots \\ E_m \\ S_m \end{bmatrix} = \begin{bmatrix} l_1 & l_2 & \dots & l_n \\ e_{1,1} & e_{1,2} & \dots & e_{1,n} \\ s_{1,1} & s_{1,2} & \dots & s_{1,n} \\ \vdots & \vdots & \ddots & \vdots \\ e_{m,1} & e_{m,2} & \dots & e_{m,n} \\ s_{m,1} & s_{m,2} & \dots & s_{m,n} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \quad (1)$$

ここで、n, m はそれぞれノードの最大値および過去にさかのぼった時刻の最大値 (ex.$m = 24$ ならば、24 時間前のこと) を示す。x_i はノードに処理を依頼するための重み係数、l_i はノード i の現時刻における負荷の値、$e_{j,i}, s_{j,i}$ はそれぞれノード i の j 時刻前における 1 時間分の負荷の平均、および平均と標準偏差の和である。

また、左辺ベクトルを構成する L, E_j, S_j はそれぞれ現時刻におけるノード i の負荷と重み係数 x_i を掛けたものの和、j 時刻前におけるノード i の平均と重み係数 x_i を掛けたものの和、そして、j 時刻前におけるノード i の平均と標準偏差の和と重み係数 x_i を掛けたものの和である。

本稿における Grid ノードの選抜は、式 (1) における左辺ベクトルの大きさが最小となる右辺の X ベクトル ($X = (x_1, x_2, \dots, x_n)$) を求めることである。

3 実験

本節では、2.2 節で説明した内容の妥当性を示すために、次の実験を行う。

3.1 実験環境

実験環境は奈良先端科学技術大学院大学 情報科学研究科棟の情報基盤設備の一部 (同一性能の端末 40 台) をハードウェアおよびネットワークトポロジの変更をせずに用いた。表 1 に実験で用いたノードのスペックを示す。

表 1: 実験対象システムの諸元

CPU	Pentium III 800MHz
Memory	512MByte
OS	FreeBSD 4.2-RELEASE
Network	100Base-TX Ethernet
Compiler	gcc version 2.95.2
MPI	MPICH version 1.2.2.3

各ノードの負荷情報は、NET-SNMP Project[6] が配布している ucd-snmp 4.2.3 を用いて 1 分間の load average の値を定期的に収集した。

3.2 実験方法

並列コンピュータの実効性能を知る上で最も使われている NPB ver 2.3[5] を用いて、プログラムの実行に要した時間を

1. 負荷などを考慮せずに用いたノード
2. 本稿で提案するノード選抜モデル (後述するノード選抜決定アルゴリズムを含む) を用いたノード

とで比較を行う。測定回数は 3 回、プログラム実行時に使われるノードの台数は 8 台と 16 台とし、実行に要した時間の平均値と分散値で評価する。式 (1) の m の値は 1 日前における 1 日分のデータ ($m = 24$) を用いた。また、NPB の 8 種類のプログラムの中から EP と CG の 2 種類について比較を行う。EP と CG についての特徴を以下に示す。

EP 典型的なモンテカルロシミュレーションを行うベンチマーク。指定されたスキームに従い生成された、多数のガウス分布に従う擬似乱数を用いて、2 次元の統計情報を蓄積する。コードはほぼ完全な並列化がされており、データの転送を殆んど必要としない。よって、ノードの演算能力に注目して評価する上で適当なベンチマークである。

CG Conjugate Gradient 法を用いた大規模疎行列の最小固有値を求めるための評価を行うベンチマーク。ノードの台数が増えるにつれ、1 回の通信に要するデータ量は減るものの、通信回数が増える。よって、ネットワークの状態を考慮した評価をする上で適当なベンチマークである。

3.3 ノード選抜決定アルゴリズム

ここでは、式 (1) における左辺ベクトルの大きさが十分小さくするための X ベクトルを求める方法について説明する。

条件は 2.2.1 節のとおりとし、すべてのノードが利用できる環境とする。また、並列実行プログラムをノードに割り当てる際、必要とされるノードの数に対してすべて均等に割り当てることとする。(1 ノードに対して複数のジョブを割り当てることはしない。)

選抜決定アルゴリズム

ここでは、2 時間前の情報を知っている 5 台のノードで構成される Grid から 2 台選抜する例を使って説明をする。なお、式 (1) の右辺のマトリクスは以下の状態とする。

$$\begin{bmatrix} 1.85 & 0.12 & 2.21 & 0.50 & 0.55 \\ 2.01 & 0.07 & 1.50 & 0.08 & 0.08 \\ 2.06 & 0.09 & 1.55 & 0.09 & 0.10 \\ 0.50 & 0.10 & 0.08 & 1.00 & 0.40 \\ 0.55 & 0.11 & 0.10 & 1.01 & 0.47 \end{bmatrix}$$

1. 式 (1) の右辺のマトリクスと同じ大きさのマトリクスを準備し、すべての要素の値を"0" にする。このマトリクスを A マトリクスと呼ぶ。

$$A = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

2. 式(1)の右辺のマトリクスの各行について、値の小さい順から A マトリクスの要素の値を"1" にする。
   example.

$$A = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix}$$

3. A マトリクスの各列毎の和をとったベクトルを作成する。これを Y ベクトルとする。

   example.

$$Y = \begin{bmatrix} 0 & 5 & 2 & 3 & 1 \end{bmatrix}$$

4. Y ベクトルの要素の値が大きい順に X ベクトルの要素の値を"1" にする。なお、X ベクトルにおいて含まれる"1" の個数は並列実行に必要なノードの数分だけ存在し、残りはすべて"0" である。この X ベクトルがノードを選抜した結果である。
   example.

$$Y = \begin{bmatrix} 0 & 5 & 2 & 3 & 1 \end{bmatrix}$$

$$\Downarrow$$

$$X = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$

3.4 評価

表2および表3はそれぞれEP CLASS BおよびCG CLASS Bにおけるプログラム処理時間の結果を示したものである。以下で各々のベンチマークの結果について述べる。

3.4.1 EP

EPベンチマークはほぼ完全な並列化がなされているため、処理時間の短縮は各ノードのCPUの処理状況に大きく依存する。どれか1つのノードで何らかのジョブが実行されているとすれば、このノードが影響して処理時間の短縮が計れない。実験結果では提案手法を用いることで3倍弱の実行時間の短縮が可能となった。また処理時間の揺らぎも小さくなっている。ノードの処理状況を考慮した結果が明らかになった例である。

表 2: EP CLASS B におけるプログラム処理時間の比較

ノード数	負荷考慮なし		提案手法	
	平均	分散	平均	分散
8	430.71s	1.72	144.03s	0.25
16	215.12s	1.12	73.32s	0.89

表 3: CG CLASS B におけるプログラム処理時間の比較

ノード数	負荷考慮なし		提案手法	
	平均	分散	平均	分散
8	750.16s	5.63	559.24s	13.81
16	582.96s	5.42	497.28s	4.11

3.4.2 CG

CGベンチマークはノードの台数が増えるにつれ通信回数が増えるので、ノードの処理能力のみならず、ネットワークの状態も処理時間の結果に影響を及ぼす。

本稿ではネットワークの入出力の状態については考慮していないが、提案手法を用いることで1.2倍前後の実行時間の短縮が可能となった。ノードが8台のときの提案手法の分散の値が大きいが、ベンチマークプログラムの実行中にノード間を接続するネットワーク上で何らかの遅延が発生するようなトラフィックが起きたためではないかと思われる。今後の課題として、ネットワークトラフィックの状況も考慮に入れる必要がある。

4 関連研究との比較

小出らは、COMPACS に対応した資源情報サーバ (Resource Information Server;RIS) [7] において、過去の資源情報の変動パターンと類似したパターンを資源情報サーバから検索し、将来の資源情報の予測を行っている。各ノードの状態の変動予測を高精度で得られるが、類似したパターンの検索をすべてのノードに対して行う必要がある。ノードの数が多い Campus Grid において、すべてのノードの変動予測を低コストで行うことが望ましいが、この方法では類似パターン検索のコストが大きくなるため望ましくない。

Dinda らは、線形時系列モデルをノードの負荷の予測に適用し [8]、過去のノードの振舞いから実用的な精度で予測できることを指摘している。過去のノードの振舞いの情報から、未来のノードの振舞いや Grid の性能を予測できるという点では我々の考え方と合致している。しかし、各ノードの負荷変動はそのノードの利用者の作業用途や生活パターンといった人為的要因によるものとの観点から考えているため、突発的事象が考慮されていない線形時系列モデルは相応しくない。

5 まとめ

本稿では学内のネットワークに接続された様々な用途に利用されている端末を用いて Grid 環境を構築する Campus Grid に焦点を当て、各ノードにおける過去の負荷変動の履歴に対して平均・分散モデルによる特徴抽出を行い、抽出結果を用いたノード選抜モデルを提案した。また、大学の情報基盤設備を用いた NPB version 2.3 の EP, CG ベンチマークの評価を行った。

その結果、ノードの過去の負荷変動の履歴を考慮したノード選抜を行うことで処理時間の短縮および処理時間のゆらぎを低減できることがわかった。

本稿では、ホモジニアスな Campus Grid 環境においてノードの負荷のみに着目したノード選抜モデルとなっている。キャンパス内に存在するすべての計算機を利用することを想定した場合、ヘテロジニアスな環境である。また、ネットワーク帯域やメモリやハードディスクといった記憶装置の空き容量といった、CPU 以外の計算機資源も考慮に入れる必要がある。今後はヘテロジニアスな環境に対応したモデル、および CPU 以外の計算機資源を視野に入れたモデルといったものに対して本稿で提案したノード選抜モデルの拡張を行う予定である。

参考文献

[1] Foster,I. and Kesselman,C.(Eds.): The Grid: Blueprint for a New Computing Infrastructure, Morgan Kaufmann, 1999

[2] Foster,I. and Kesselman, C.: Globus: A Metacomputing Infrastracture Toolkit, International Journal of Supercomputer Applications, Vol.11, No.2, pp.115-128, 1997

[3] Asia-Pacific Grid: http://www.apgrid.org/

[4] The Beowulf Project: http://www.beowulf.org

[5] NAS Parallel Benchmarks: http://www.nas.nasa.gov/Software/NPB/

[6] NET-SNMP: http://net-snmp.sourceforge.net/

[7] 小出 洋, 山岸 信寛, 武宮 博, 笠原 博徳: 資源情報サーバにおける資源情報予測の評価, 情報処理学会論文誌 プログラミング, Vol.42, No.SIG3(PRO 10), pp.65-73, 2001

[8] Dinda, P. and O'Hallaron, D: An Evaluation of Linear Models for Host Load Prediction, Proc. 8th IEEE International Symposium on High Performance Distributed Computing, http://www.computer.org/proceedings/hpdc/0287/02870010abs.htm