計算機アーキテクチャ 147－5
ハイパフォーマンス 89－5
コンピューティング
（2002．3．7）

Gridにおけるネットワーク負荷予測

秋岡明香† 村岡洋一†

本稿では，Gridにおけるネットワーク負荷を定義し，このネットワーク負荷を測定・予測する手法を提案する．提案するネットワーク負荷測定手法は，Gridでの本来の目的であるアプリケーションの実行を妨げないことを目的とし，従来と比較してネットワーク負荷測定のコストは1/64に抑えることができる．また提案予測手法は，複数予測手法の中からマルコフモデルによりその状況に応じた手法を適宜選択して予測値を求める手法で，評価実験により従来手法よりも正しい予測手法を選択する割合が7.5％上昇し，測定値に対する予測誤差の割合の平均が12.3％と良好な結果を得た．また，複数予測手法を組み合わせることの有効性も確認できた．

Predicting Networking load on computational Grid

SAYAKA AKIOKA† and YOICHI MURAOKA†

In this paper, we propose a technique to measure and predict networking load on the computational Grid. Our mesuring method costs lower than usual ones, and it does not disturb other applications' communications. Our predicting method consists of some independent predicting methods. Each one caliculates a predicting value, and the one that seems to be most exact is chosen finally based on Marcov Model. Some experiments showed our predicting method provides more precise predictions than usual methods did, and we confirmed that several methods in cooperation work better than a single method.

1. はじめに

世界各地に散在する計算機を広域ネットワークで接続し，ひとつの巨大な計算機クラスタと見なして運用する，Gridに関する研究が盛んに行なわれている．Gridの計算基盤に関する研究は既に多く行われている[1)~5)]．あるいは，昨今注目を集めているPeer-to-Peerは，広域ネットワークにより接続された対等な計算機群であり，Gridの一種であると言える．Gridを利用したアプリケーション例[6)~9)]は急速に増えているが，高性能計算アプリケーションの実行には多くの課題が残されている．

高性能計算アプリケーションをGridで実行する上での大きな課題はスケジューリングである．提案されているスケジューラ[10)~12)]の多くは，定期的な計測により取得した計算資源の負荷に基づいてスケジューリングを行なう．しかし，Gridの計算機群は広域ネットワークで接続されているため，計算資源の負荷は時々刻々と変動している．従って，頻繁に負荷計測を行ない最新の計測結果に基づいてスケジューリングを行なわなければ，効果的なスケジューリングは実現できない．しかし頻繁な負荷計測は，その方法によっては本来の目的であるアプリケーションの実行を妨げる恐れがある．

そこで本稿では，Gridの重要な計算資源の一つであるネットワークの負荷予測のために，これまでに提案されている手法に比べて2桁近くオーバヘッドを削減でき，またその予測精度も10％近く改善できる手法を提案する．提案手法の特徴は，時々刻々と変化するネットワークの負荷状況に対応するため，複数の予測手法の中からマルコフモデルにより，適宜適用手法を切り替えていく点にある．以降，第2章では従来研究の概要を述べる．次に第3章で提案システムについて述べ従来研究との相違点を明らかにする．第4章では提案システムに対する評価を行ない，その結果について考察を行なう．最後に，第5章で提案システムの特徴，評価についてのまとめを行ない，今後の課題を明確にする．

† 早稲田大学
Waseda University

2. 従来研究

2.1 ネットワーク負荷予測

Gridでのスケジューリング支援の目的でネットワーク負荷を予測する代表的な研究として，Network Weather Service (NWS)[13)]がある．NWSはCPUとネットワークそれぞれの負荷やメモリの空き容量を定期的に測定し，その測定値から1ステップ先の値を予測するシステムである．計算資源の負荷計測は各計算機上で行ない，計測データは負荷予測値計算サーバに集める．予測値の計算は負荷予測値計算サーバで行なう．NWSでは，最新の負荷計測値から複数の予測手法で複数の予測値を求めておき，前回最も誤差の小さい予測値を与えた手法により得られた値を予測値とする．個々の予測手法は，利用者が自由に差し替えることが可能で，そのためのAPIを整備している．

2.2 ネットワーク負荷計測

実際の運用を考慮した場合，ネットワーク負荷の予測手法だけでなく，ネットワーク負荷の計測手法が重要となる．ネットワーク負荷測定に関する研究は既に多くあるが，これらの研究は大きく二つに分類できる．実際にデータを流してRTT，パケットロス，遅延，スループット等を測定するアクティブな手法と，主要地点のルータ等にソフトウエアモニタ或いはハードウエアモニタを設置して同様の測定を行うパッシブな手法である．本研究のように，任意の二計算機間のネットワーク負荷を測定する場合は，前者のアクティブな手法がより適している．前述のNWSでは，各計算資源の負荷を一定の時間間隔で測定するが，ネットワークの負荷を測定するために64k bytes程度のデータを定期的に転送し，転送に要した時間と遅延を測定

する．

アクティブなネットワーク負荷測定手法研究の代表例として，Active Measurement Project (AMP)[14]，NIMI[15]，Surveyor[16]，PPNCG[17] を挙げることができる．各研究の概要をまとめると，RTT やパケットロスは ICMP プロトコルで測定し，TCP スループットは一定の大きさのデータを転送することで測定している．また，現在使用されているネットワークプロトコルは測定を行うには不適切であるという観点から，AMP では IPMP[18] という新プロトコルを提案している．

3. 提案手法

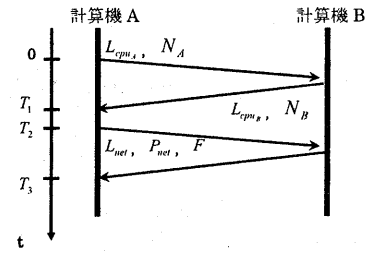
NWS のようなサーバ・クライアント型システムは，計算機間での情報の共有が容易である一方で，拡張性に優れているという Grid の大きな利点を損ないがちである．またスケジューラが負荷情報やその予測値を参照する場合には，その都度負荷情報サーバにアクセスして情報を取得する必要があり，通信が多発する．従って，実行時間が比較的短いタスクのように，頻繁なスケジューリングを必要とする類のタスクを扱うには不向きである．そこで本稿では，分散型のネットワーク負荷予測システムを提案する．分散型ネットワーク負荷予測システムとは，各計算機上でネットワーク負荷の測定・予測を行ない，サーバを必要としないシステムを指す．

分散型のネットワーク負荷予測システムは，サーバ・クライアント型システムと比較して Grid の持つ優れた拡張性を保持することが可能である．また，負荷予測システムの利用者であるスケジューラに，提案手法と同様な分散型のスケジューリングシステムを想定した場合，その計算機と他の計算機の間のネットワーク負荷やその予測値は勿論のこと，遠隔計算機の cpu 負荷等も手元の計算機に蓄積されているため，スケジューリングに必要な情報を収集するための通信は発生しない．従って，従来よりも実行時間が短いタスクに対するスケジューリングが可能となる．一方で，分散型システムでは最新の情報を共有することが難しいが，この問題点を解決し，可能な限り他のアプリケーションの実行を妨げずにネットワークの負荷を測定し，予測値を求めるシステムの構築を目指す．以降，本章ではこの提案システムについて具体的に述べる．

3.1 提案手法の概要

本稿では，広域ネットワークで接続された全ての計算機が対等な立場で接続された分散型のネットワーク負荷予測システムを提案する．つまり，各計算機が自律的に他計算機との間のネットワーク負荷情報を収集し，収集した情報から他計算機との間のネットワーク負荷の予測を行なう．提案システムは，NWS のようにサーバを必要とするシステムと比較して，Grid の持つ拡張性の高さを充分に活かすことが可能であるが，その一方で負荷の情報をシステム全体で共有することや，システム全体の状態を把握することが極めて困難である．しかし，現実の運用を考えると，個々の計算機の負荷をシステム全体で共有したり，個々の計算機がシステム全体の状況を把握しておくことは，以下の理由により必要条件ではない．

実際の Grid 上でのアプリケーションの運用を考えた場合，あるアプリケーションの実行のために計算資源を提供できる計算機を募り，申し出のあった計算機上でのみそのアプリケーションを実行するのが通常である．従って，スケジューリングも参加の意思を表明している計算機群のみを対象にして行なえば十分である．この点を考慮すると，個々の計算機はシステム全体の状況や個々の他の計算機の負荷を把握しておく必要はなく，同じアプリケーションを実行する計算機群の中で負荷情報の共有を行なうことができれば十分であることが分かる．そこで本稿では，同じアプリケーションを実行したり，同じプロジェクトに参加する計算機群をグループ化することを前提とし，このグループ内においてネットワーク負荷の計測と予測を行ない，計算資源の負荷情報を共有するシステムを提案する．



L_{cpu_A}：計算機 A の cpu 負荷，N_A：計算機 A が先行で計測する数

L_{cpu_B}：計算機 B の cpu 負荷，N_B：計算機 B が先行で計測する数

L_{net}：ネットワーク負荷，P_{net}：ネットワーク負荷の予測値

F：次回計測時の先行計算機，$T_{trans} = T_1$，$\Delta T = T_3 - T_2$

図 1 ネットワーク負荷計測の手順

3.2 ネットワーク負荷測定手法と情報の共有

本節では，ネットワークの負荷を測定する手法について具体的に述べる．本稿では，ネットワークの負荷の指標として以下に述べる値を用いる．ある 2 つの計算機間でサイズ D_{trans} のデータの転送に要した時間が T_{trans}，ごく小さいデータの転送に要した時間が ΔT である場合，これらの計算機間の負荷 L_{net} は次式により定める．

$$B = \frac{D_{trans}}{T_{trans} - \Delta T}$$

$$T_{latency} = \frac{\Delta T}{2}$$

$$L_{net} = \frac{T_{latency}}{B}$$

この時の D_{trans} の適切なサイズについては後述する．各計算機は，この定義と図 1 に示す手順によりネットワーク負荷の計測を行なう．なお各計算機は，グループに初めて参加する時にそのグループに参加する計算機のリストを受け取り，最初の一回のみリストにある全ての計算機に対して図 1 の手法でネットワーク負荷の計測と負荷情報の交換を行なう．各計算機では，このリストにない計算機からネットワーク負荷計測用データを受け取った場合には，その計算機をリストに追加する．これにより，各計算機の持つリストを常に最新の状態に保つことができる．

また NWS では，ネットワークの負荷を測定するだけの目的で，計算機間で 64k bytes 程度のデータを転送していた．従って，計算資源の負荷情報を得るためには新たに通信を行なう必要があった．さらに，二つの計算機間で同時にネットワーク負

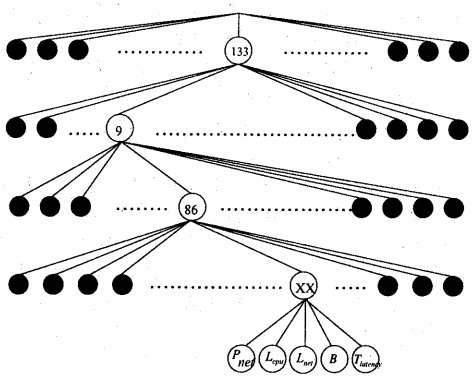


図 2　負荷情報データベース

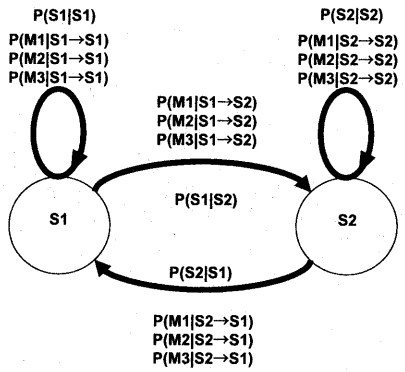


図 3　予測手法選択のためのマルコフモデルの例

荷の測定を始める可能性があるため，NWS では正しい計測を行なうためにトークンを用いているが，通信コスト自体は削減されない．またこの手法では，ネットワーク負荷の測定の衝突を完全には回避できない．そこで本稿では，これらの通信の無駄を減らし，衝突を防ぐために，ネットワーク負荷の測定手法を以下のように定めた．

- 2 つの計算機間では，ネットワーク負荷測定のために以下情報を含むデータを，約 1 分間隔で転送する．
  - ネットワーク負荷以外の計算資源の負荷情報
  - 現在自分が先行でネットワーク負荷の測定を行なっている相手計算機の数

- 2 つの計算機間では，ネットワーク負荷測定を先行して行なう計算機をどちらか一方に定めておき，双方の計算機からネットワーク負荷測定が開始することを避ける．ここで，1 により双方が先行して負荷計測を行なう相手計算機数を交換し，その数が少ない方が次回の計測を先行して行なうと決める．

以上の手順により収集された計算資源の負荷情報と，以下で述べる手法により求めたネットワーク負荷とその予測値，cpu 負荷は，それぞれ相手の計算機の IP アドレスと関連付けられたツリー状の負荷情報データベース（図 2）に格納され，必要に応じて参照できる．

3.3　予測値計算のアルゴリズム

以下では，3.2 で述べた手法により得たネットワーク負荷の計測値から，ネットワーク負荷の予測を行なう手法について述べる．

提案手法は，3.2 により得たネットワーク負荷から複数の予測アルゴリズムにより複数の予測値を計算し，次回最も誤差が小さい予測値を与えると期待される予測アルゴリズムから得られた値を，予測値として提供する．ネットワーク負荷計測から予測値算出までの流れを，以下に示す．

(1)　ネットワーク負荷を計測する．
(2)　前回の予測値と比較して正解を得て，確率分布を再計算する．
(3)　複数の予測手法により予測値を計算する．
(4)　次回の予測値をマルコフモデルにより決定する．

単一の手法による予測は，その予測手法が得意とする状況下では非常に良い精度で予測を行なうことが期待されるが，逆にその手法が不得意とする状況においては極端に誤差の大きい予測値を与えてしまう．しかし，異なる特徴を持つ複数の予測手法の中から，その時々の状況に応じた手法を選択して予測を行なうことで，あらゆる局面に柔軟に対応しながら，平均的に誤差の小さい予測を行なうことが可能になる．

次回最も誤差が小さい予測値を与えると思われる手法には，図 3 に示すマルコフモデルにより，次回最も誤差が小さい予測値を与える確率が大きい手法を選ぶ．図 3 において，$S1$ は前回と前々回で最適な予測値を与えた手法が同じである状態，$S2$ は前回と前々回で最適な予測値を与えた手法が異なる状態を表す．$P(m|a \rightarrow b)$ は状態が a から b へ遷移した際に m を出力する確率を表す．ここで m は予測手法を表す．このマルコフモデルは，ネットワーク負荷を計測する度に確率分布が再計算され，最適化される．具体的には，前回求めた予測値と実際の負荷を比較し，どの手法による予測値が最も誤差が小さかったかを求め，この正解に基づいて確率分布の再計算を行なう．このように，各計算機上でモデルを最適化していくことで，その計算機が置かれている環境に最適なモデルが得られることが期待できるため，時間経過と共に予測精度が向上することが見込まれる．

4.　評価と考察

以下では，ネットワーク負荷測定のために転送するデータの大きさについての考察と，提案予測手法に対する評価を行なう．

4.1　ネットワーク負荷測定のための転送データサイズの考察

本節では，ネットワーク負荷測定のために用いるデータサイズについて考察を行なう．従来手法のように，ネットワーク負荷測定のために定期的に数十 k bytes ものデータを転送することは，本来のアプリケーション実行を妨げる可能性が極めて高い．そこで，このネットワーク負荷測定のためのパケットサイズについて以下の実験を行なった．

- 図 4 に示す 2 計算機間のネットワーク負荷を測定した．
- ネットワーク負荷測定に用いる転送データのサイズは，1k bytes，2k bytes，4k bytes，8k bytes，16k bytes，32k

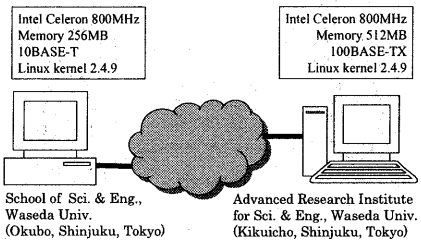


図 4　評価環境

bytes，64k bytes の 7 種類を用いた．

- ネットワーク負荷の測定は 3.2 で述べた手法により行ない，各データサイズについて約 7 分間隔で行なった．
- データの転送は tcp により行ない，実装には C 言語を用いた．
- 実験は 2002 年 1 月 23 日午前 2 時 30 分から，およそ 24 時間に渡って行なった．
- 各測定結果より，各データサイズについてネットワークスループットを計算して求めた．

結果を図 5 に示す．各データサイズについて同時に測定を行なうことは不可能であるためデータに多少のバラつきがあるものの，ネットワーク測定に用いたデータのサイズとスループットの間には一定の関係があることが分かる．このことより，データサイズを 1k bytes に設定することで，ネットワークの負荷を測定し，複数のネットワークを比較検討することが可能であると言える．なお，数箇所で負の値を測定しているが，これは 1 で述べた後半の測定でごく小さいサイズのデータを転送した際に，最初の転送よりも時間がかかったことが原因である．これにより，数 bytes のデータ転送では正しくネットワークの負荷を測定することが難しいことが分かると同時に，今後はネットワーク遅延の測定方法について更なる検討が必要であることが分かった．

4.2 予測精度

本節では提案予測手法の予測精度を評価する．3.2 に述べた手法により測定したネットワーク負荷の 1 ステップ先の値を 3.3 で述べた手法により予測し，その予測精度を調べ，従来手法との比較を行なった．評価の詳細を以下に示す．

- 図 4 に示す 2 計算機間でネットワーク負荷の計測，及び 1 ステップ先のネットワーク負荷の予測を行なった．
- ネットワーク負荷の測定に用いる転送データのサイズは，4.1 の結果に基づき，1k bytes とした．
- ネットワーク負荷の測定は 3.2 で述べた手法により行ない，約 1 分間隔で測定を行なった．
- データの転送は tcp により行ない，実装には C++言語を用いた．
- 実験は 2002 年 2 月 7 日午後 1 時から，およそ 24 時間に渡って行なった．ただし，最初の 12 時間はマルコフモデルの学習期間とした．

なお，本実験で用いた個々の予測手法を以下に示す．以下では，時刻 t におけるネットワーク負荷の計測値を $L_{net}(t)$，時刻 t におけるネットワーク負荷の予測値を $P_{net}(t)$ とする．

SG　統計的最急勾配法により予測値を求める手法．TCP/IP には，本手法によるエンドノード間のラウンドトリップタイムの予測が組み込まれているため，提案手法でも採用した．なお，今回は $g = 0.5$ とした．

$$P_{net}(t+1) = (1-g) * P_{net}(t) + g * L_{net}(t)$$
$$(0 < g < 1)$$

SW　過去の測定値の平均値を予測値とする手法．最近数件の計測値の平均値を予測値とすることで，最近の傾向を予測値に反映させることができる．また，直前値をそのまま予測値として出力するよりも，広い範囲に渡る傾向が反映できる．以下では i を $err_i(t)$ を最小にする値とする．今回は $1 \leq K(t) \leq 10$ とした．

$$P_{net}(t+1) = SW(t, K(t))$$

$$SW(t, K(t)) = \frac{1}{K(t)+1} \sum_{i=t-K(t)}^{t} v(i)$$

$$K(t+1) = K(t) + i$$

$$err_i(t) = (L_{net}(t) - SW(t, K(t)+i))^2$$

MF　メディアンフィルタにより予測値を求める手法．本手法により最近の測定値の中間値を選ぶことで，一時的な急変に引きずられて大きく狂った予測値を出力することを防ぐことができる．以下では，S_K は最近 K 個の測定値をソートした数列を表わし，$S_K(l)$ は数列 S_K の l 番目の要素を表わす．ただし i は $err_i(t)$ を最小にする値とする．

$$P_{net}(t+1) = MF(t, K(t))$$

$$MF(t,K) = \begin{cases} S_K((K+1)/2) \\ \quad (K : odd) \\ \frac{S_K(K/2)+S_K(K/2+1)}{2} \\ \quad (K : even) \end{cases}$$

$$K(t+1) = K(t) + i$$

$$err_i(t) = (L_{net}(t) - MF(t, K(t)+i))^2$$

AR　自己回帰モデルにより予測値を求める手法．近年の研究により，ネットワークの状況は ARIMA モデルにより表現できることが知られている[19)20)]．しかし，ARIMA による予測には非線型方程式を解く必要があるため，今回は解くのが容易な自己回帰モデルを採用した．時刻 t における M 次自己回帰モデルは次式により表現できる．今回は最大次数を 10 とした．

$$L_{net}(t) = \sum_{m=1}^{M} a(m)v(s-m) + err(t)$$

$$1 - a(z) - a(z^2) - ...a(M)z^M = 0$$

実験結果を，図 6，図 7，表 1，表 2 に示す．これらの結果より，以下の結論を得た．

- 図 7 より，複数予測手法を組み合わせる提案手法が有効であり，単一の手法では状況の変化に対応できないことが分かった．
- 表 1 より，従来手法と比較して提案手法が正しい予測手法を選択する割合が 7.5 %上昇した．
- 表 2 より，予測手法の組み合わせを工夫することで，更なる予測精度の向上が見込まれる．
- 実験全体を通じて，従来手法と比較して提案手法の方が，大きい変化に対して柔軟に対応する傾向が見られた．これ

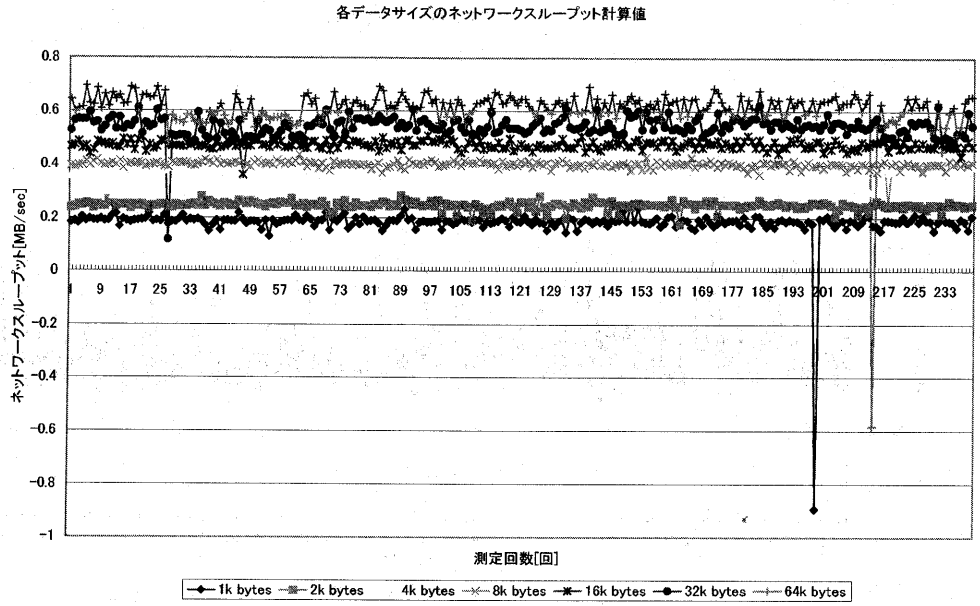


図 5　実験結果 1

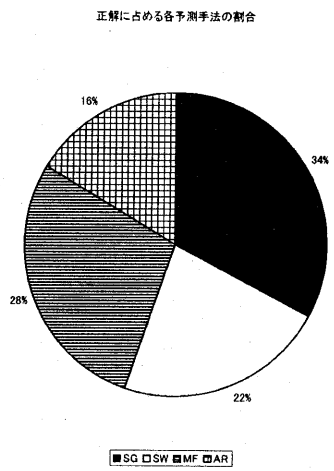


図 7　実験結果 3

表 1　実験結果 4

	正しい予測手法を選択した割合
従来手法	25.9 %
提案手法	33.4 %

は，直前に良い結果を出した手法をそのまま継続して利用する従来手法では，一定の変化が続いている場合には良い結果を出すが，激しい変化を繰り返す場合等には弱いことを示す結果である

表 2　実験結果 5

	予測誤差の測定値に対する割合
全て正しい手法を選んだ場合	7.9 %
提案手法	12.3 %

5. おわりに

本稿では，Grid においてネットワーク負荷を測定，予測する手法を提案した．提案測定手法は，本来のアプリケーションの実行を妨げることのないよう工夫を凝らしたもので，従来の 1/64 のコストでネットワーク負荷を測定することができる．提案手法予測手法は，マルコフモデルより複数予測手法からそれぞれの状況に適した予測手法を選択して予測値を求めるものである．これらの提案手法を実験により評価したところ，提案予測手法は従来手法と比較して，正しい予測手法を選択する割合が 7.5 %上昇し，予測誤差の測定値に対する割合は 12.3 %と良好な結果を得た．一方で，ネットワーク遅延測定方法や予測手法の選択・組み合わせに工夫の余地を残すことも分かった．今後はこれらの問題点に取り組み予測精度を上げていくと共に，利用者が好みの予測アルゴリズムを採用できるように API を整備して，本システムを公開する予定である．

参考文献

1) Condor. http://www.cs.wisc.edu/condor/.
2) Globus. http://www.globus.org/.
3) Legion. http://www.cs.virginia.edu/~legion/.
4) Netsolve. http://icl.cs.utk.edu/netsolve/.
5) Ninf. http://ninf.apgrid.org/.
6) climateprediction.com. http://www.climateprediction.com/.

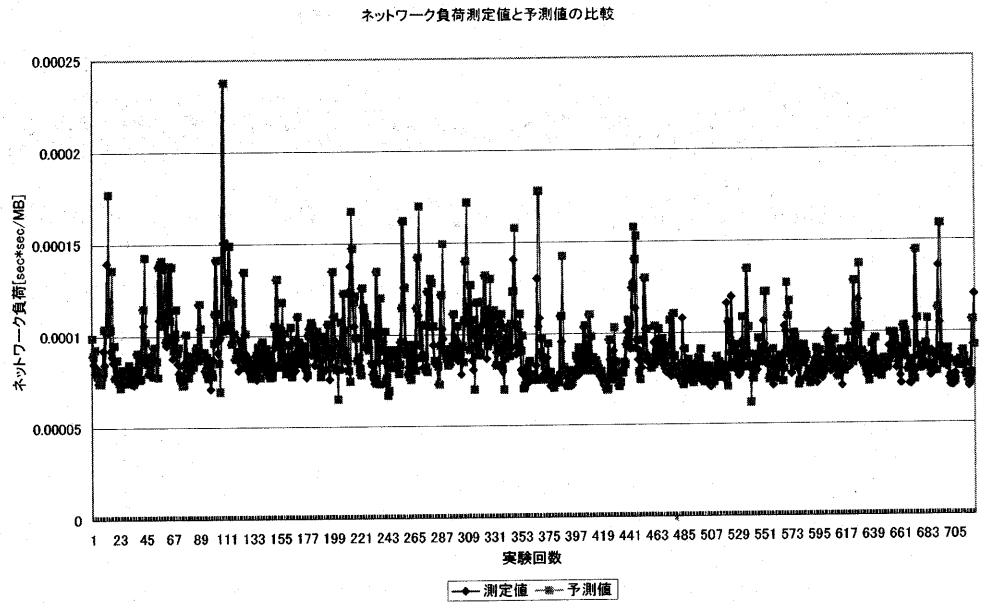


図 6 実験結果 2

7) fightAIDS@home. http://www.fightaidsathome.org/.
8) HyperBee. http://www.hyperbee.org/.
9) SETI@home. http://setiathome.berkeley.edu/.
10) AppLeS. http://apples.ucsd.edu/.
11) PC^2-CCS. http://www.uni-paderborn.de/pc2/projects/ccs/.
12) Prophet. http://ringer.cs.utsa.edu/faculty/jon/het.html.
13) Network Weather Service. http://nws.cs.utk.edu/.
14) Active Measurement Project. http://amp.nlanr.net/.
15) NIMI. http://www.ncne.nlanr.net/nimi/.
16) Surveyor. http://www.advanced.org/surveyor/.
17) PPNCG. http://icfamon.rl.ac.uk/ppncg/title.html.
18) Ipmp. http://watt.nlanr.net/AMP/IPMP/.
19) A. Mukherjee S. Basu and S. Kilvansky. Time series models for internet traffic. Technical Report GIT-CC-95-27, Georgia Institute of Technology, 1996.
20) N. Groschwits and G. Polyzos. A time series model fo long-term traffic on the nsfnet backbone. In *Proceedings of the IEEE International Conference on Communications (ICC 794)*, May 1994.