ハイパフォーマンス
コンピューティング 83－9
（2000．10．6）

グローバルコンピューティングシステムにおける ネットワークGantt図を用いたジョブスケジューリング手法の提案

上田清詩† 本多弘樹† 弓場敏嗣†

現在，グローバルコンピューティングシステムが複数提案されており，その資源を適切に利用するためのスケジューリング手法に対する議論が続いている．特に，ネットワークのスループットが変動する広域ネットワーク環境においては，それらを考慮して適切に資源を割り当てるスケジューリング手法が必要である．本稿では，広域ネットワークの動作をモデル化し，ネットワークのスループットの変動予定を表現するネットワーク Gantt 図を提案した．そして，ns(Network Simulator) でのシミュレーションにより，その正当性を示した．また，ネットワーク Gantt 図を用いてスループットの変動予定を考慮したスケジューリング手法を提案し，その有効性を確認した．

A Job Scheduling Strategy based on Network Gantt-Chart for Global Computing Systems

KIYOSHI UEDA,† HIROKI HONDA† and TOSHITSUGU YUBA†

In recent years, some of the global computing systems have been proposed, and the problem of scheduling strategies which allocate resources appropriately is still open. In wide-area networks, the available throughput of network resources is change dynamically. Consequently, a scheduling strategy which considers these condition and allocates resources appropriately is necessary. In this paper, we modelled wide-area networks and proposed a Network Gantt-Chart which describe a schedule of the available throughput's fluctuation in the network resources. And we showed the validity of both model and Network Gantt-Chart by the ns(Network Simulator). Also, we proposed a scheduling strategy based on the Network Gantt-Chart that considers the schedule of the throughput's fluctuation, and verified its effectiveness.

1. はじめに

近年，広域ネットワーク上に分散した計算資源や情報資源を活用し，大規模計算要求に応えることを目的としたグローバルコンピューティングシステムが複数提案されている [1)~4)]．このようなシステムにおける計算サーバは，ネットワーク上に分散する多数のユーザにより共有される．そのため，各ユーザの一定の要求を満たすように適切なサーバとデータ転送路（以後パスと呼ぶ）を選択し，利用させるスケジューリング手法が必要である．

現在提案されているいくつかのシステムにおいては，ジョブが発行された時点のサーバの性能と負荷，及び，ネットワークのスループットを考慮したスケジューリング手法が用いられている．

しかし，広域環境におけるネットワークの性能は低く，スループットは常に変動する．従って，ジョブが発行された後にネットワークのスループットが変化するような場合には，ある時点のスループットのみを考慮するスケジューリング手法では，正確なデータ転送時間の予測ができないために適切なスケジューリングを行うことができず，ユーザの要求を十分には満たせなくなる．実際，広域ネットワーク上では，ある隣接する2つのルータ間のネットワーク資源（以後リンクと呼ぶ）に負荷が集中し，スループットが変動する状況が起こりやすい．そのため，リンク毎のスループットの変動を予測し，データ転送時間をより正確に算出することが要求される．

本稿では，サーバでの計算時間とサーバへのデータ転送時間の予定を考慮するジョブスケジューリング手法 (**NETG**) を提案する．データ転送時間の予定を立てるために，まず広域ネットワークをモデル化し，モデル上の各リンクの利用状況を Gantt 図を用いて表

† 電気通信大学 大学院情報システム学研究科
Graduate School of Information Systems, The University of Electro-Communications.

現する「ネットワーク Gantt 図」を提案する．ネットワーク Gantt 図はリンクの利用予定を表現できるため，各クライアントが将来どの程度のスループットでデータ転送を行える予定であるかを知ることができ，データ転送時間を算出することが可能となる．

次に，ns(Network Simulator)[5] を用いたデータ転送シミュレーションを行い，広域ネットワークモデル，及び，ネットワーク Gantt 図の正当性を示す．

また，発行されているジョブのデータ転送時間とサーバでの計算時間の和（以後処理時間と呼ぶ）の平均に関して，つまりジョブの平均応答時間に関して，NETG といくつかの基本的なスケジューリング手法を机上でのシミュレーションにより比較し，NETG の値が他のスケジューリング手法の値以下になることを示す．

更に，NETG のスケジューリングポリシを変更することにより，状況に応じた様々なスケジューリングを行うことができることに触れる．

以後，2 では広域ネットワークモデルについて説明し，3 ではネットワーク Gantt 図について述べる．また，4 では NETG について説明する．

2. 広域ネットワークのモデル化

広域ネットワークを，データ転送時のリンクの使用法の観点からモデル化し，そのモデル上でデータ転送量が既知の場合のデータ転送時間の算出式を定義する．なお，あるパス上のリンクの中で，スループットがそのパス上の他の全てのリンク以下となるリンクを，以後ボトルネックとなるリンクと言う．

2.1 リンクの使用法

あるクライアントがデータ転送を行っているパス上に複数のリンクが存在する場合，当該データ転送では，パス上の全リンクについてボトルネックとなるリンクの帯域幅分だけ使用すると仮定する．

また，複数のジョブが 1 つのリンクを同時に使用する場合，各ジョブはリンクの帯域幅を対等に分割して使用する．ただし，パス上の他のリンクがボトルネックとなり，等分した帯域幅を使いきれないジョブが存在する場合，より大きな帯域幅を要求する他のジョブが余った帯域幅を更に等分して使用すると仮定する．

2.2 データ転送時間算出式

広域ネットワークモデルでのデータ転送時間 t[sec] は，データ転送量 D[MB] と，ボトルネットとなるリンクの帯域幅 W_b[MB/s] を用い，次式で定義する．

$$t = D/W_b(0:t)$$

ただし，$W_b(0:t)$ は，時刻 0〜t[sec] の間のボトルネックとなるリンクの帯域幅である．また，データ転送開始から t_1[sec] 後にボトルネックとなるリンクの帯域幅が一度だけ変化する場合は，次式で定義する．

$$t = t_1 + (D - t_1 * W_b(0:t_1))/W_b(t_1:t)$$

つまり，帯域幅の変化前までに $W_b(0:t_1)$[MB/s] で転送されたデータ量を元のデータ転送量から差し引き，残りが $W_b(t_1:t)$[MB/s] で転送される．

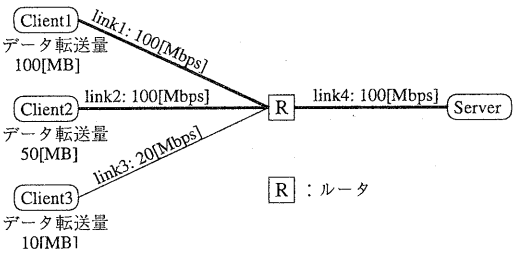


図 1 広域ネットワーク環境（1）

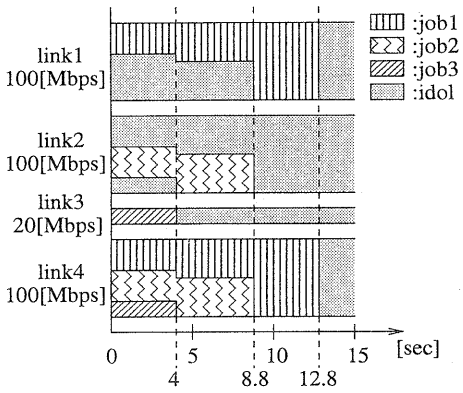


図 2 図 1 に対応するネットワーク Gantt 図

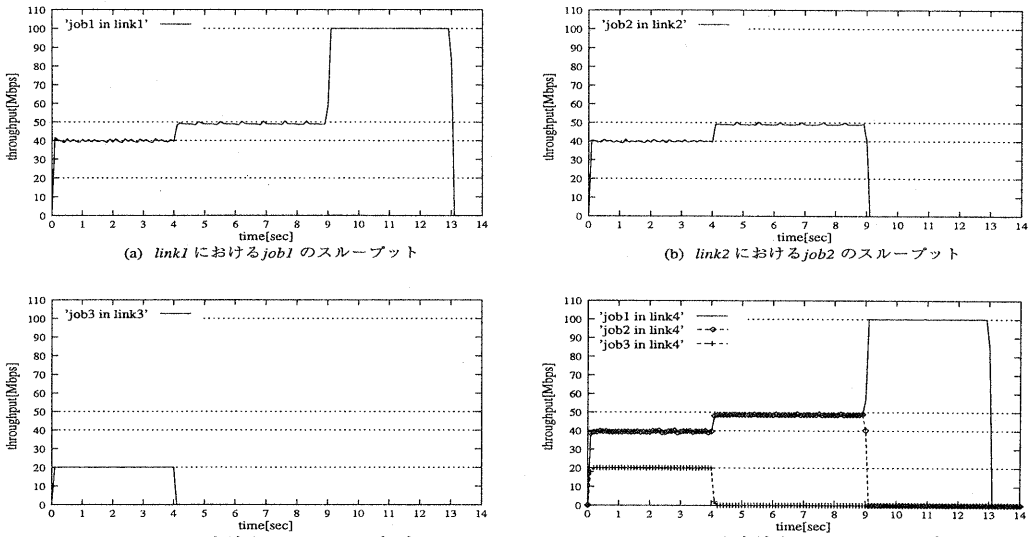
3. ネットワーク Gantt 図の提案

3.1 ネットワーク Gantt 図

ネットワーク Gantt 図は，広域ネットワークモデルのリンクの利用状況を Gantt 図を用いて表現する．以下，広域ネットワークモデル中のリンクの，ネットワーク Gantt 図を用いた表現方法を具体例を用いて説明する．

図 1 は，*Client*1, 2, 3 が *Server* に向けて，それぞれ 100[MB], 50[MB], 10[MB] のデータ転送を開始した状態を表している．この時刻を 0[sec] とし，*Client*1, 2, 3 のジョブをそれぞれ *job*1, 2, 3 とする．また，*link*1, 2, 3, 4 の帯域幅は，それぞれ 100[Mbps], 100[Mbps], 20[Mbps], 100[Mbps] である．

各クライアントのデータ転送開始から終了までの予



(a) *link1* における *job1* のスループット

(b) *link2* における *job2* のスループット

(c) *link3* における *job3* のスループット

(d) *link4* における *job1,2,3* のスループット

図 3 各ジョブのリンク毎のスループット

定を，ネットワーク Gantt 図で表現すると図 **2** となる．このような予定になる理由は以下の通りである．

(1) 時刻 0[sec] では，*link*4 は 3 つのジョブによって共有されるが，*job*3 は *link*3 がボトルネックとなり，*link*4 の帯域幅を 3 等分した 33.3[Mbps] を使いきれない．従って *job*3 は *link*3, 4 を 20[Mbps] だけ使用する．*job*3 が使いきれない帯域幅は *job*1, 2 によって更に等分して利用され，*job*1, 2 はそれぞれ *link*1, 4, *link*2, 4 を 40[Mbps] 使用する．

(2) この状態が 4[sec] 続くと，*job*3 は 20[MB] のデータ転送を終了する．すると，*link*4 を使用しているジョブ数が減少するため，*job*1, 2 は *link*4 を 50[Mbps] ずつ使用できるようになるが，このリンクが依然としてボトルネックとなる．また，時刻 0〜4[sec] までに *job*1, 2 は 40*4/8=20[MB] を転送し終っており，それぞれ残り 80[MB], 30[MB] を 50[Mbps] の帯域幅で転送する．

(3) 時刻 8.8[sec] になると，*job*2 は残り 30[MB] のデータ転送を終了する．すると，*link*4 を使用しているジョブ数が減少するため，*job*1 は *link*4 を 100[Mbps] 使用できるようになる．この時のボトルネックとなるリンクは *link*1, *link*4 のどちらでも良い．また，時刻 4〜8.8[sec] までに *job*1 は 50*(8.8−4)/8=30[MB] を転送し終っており，残り 50[MB] を 100[Mbps] の帯域幅で転送する．

(4) この状態が続くと，時刻 12.8[sec] で *job*1 は残り 50[MB] のデータ転送を終了する．

3.2 ネットワーク Gantt 図の正当性

広域ネットワークモデル，及び，ネットワーク Gantt 図の正当性を，ns(Network Simulator) を用いて検証する．図 1 の環境を ns でシミュレートし，各ジョブのデータ転送スループットの変化の様子を示したのが図 **3** である．シミュレーションでは，時刻 0[sec] に *Client*1, 2, 3 から *Server* に向けてそれぞれ 100[MB], 50[MB], 10[MB] の FTP によるデータ転送を同時に開始した．このジョブをそれぞれ *job*1, 2, 3 とする．

結果を見ると，図 2 のネットワーク Gantt 図とほぼ同様のスループットの変化をしていることが分かる．しかし，データ転送時間は *job*1, 2, 3 でそれぞれ 12.99[sec], 8.99[sec], 4.02[sec] であり，ネットワーク Gantt 図による予定より 1% 程度長くなっている．また，送受信したパケット数を計測した結果，*job*1, 2 ではそれぞれ 349[個], 343[個] のパケットロスが発生していた．データ転送時間が理論値より長くなったのは，広域ネットワークモデルで輻輳を考慮していないことが原因と考えられる．シミュレーションでは，リンクで輻輳が発生した時に TCP が輻輳制御を行うことによって，一時的にパケットを送信しない状態があった．そのため，スループットがネットワーク Gantt 図で予定されたものより僅かに低下している．しかし，輻輳が発生しない状況ではスループットの低下はない．

輻輳が発生する状態でのネットワーク Gantt 図とシミュレーションの時間差を測定するために，各ジョブのデータ転送量をそれぞれ 10 倍，50 倍，100 倍に

してシミュレーションを行ったところ，その差はデータ転送時間の 1% 程度であることが確認できた．広域ネットワーク環境において，データ転送時間を 1% 程度の誤差で算出できれば十分な精度と言える．これにより，広域ネットワークモデル，及び，ネットワーク Gantt 図の正当性が確認される．

3.3 ネットワーク Gantt 図作成の現実性

3.1 から分かるように，ネットワーク Gantt 図を作成するにはネットワーク全体のトポロジ，全リンクの帯域幅，及び，データ転送量が既知でなければならない．現実への適用を考えた場合に，トポロジ，リンクの帯域幅を知る方法として OSPF(Open Shortest Path First)[6] ルータからの情報の取得があげられる．OSPF は link-state 型のルーティングプロトコルであり，全ルータは同一のネットワークトポロジ情報を保持する．また，OSPF ではリンクの状態に変化がなければ，ある 2 点間のパスは複数存在したとしても，一意に決まる．従って，実装時にはそのパスのみのネットワーク Gantt 図を作成すれば良いことになる．

データ転送量に関して，Ninf[7] ではリモート実行の呼び出し時に問題サイズを引数として渡すことができる [1]．呼び出される手続きの種類と問題サイズが既知であれば，必要なデータ転送量は推定できる．従って，ルーティングプロトコルに OSPF を使用しているグローバルコンピューティングシステムであれば，ネットワーク Gantt 図を作成することが可能である．

4. ジョブスケジューリング手法の提案

4.1 対象とする問題

与えられたグローバルコンピューティングシステムにおいて，データ転送時間よりサーバでの計算時間が支配的な問題の場合には，サーバの性能と負荷のみを考慮したスケジューリング手法でも適切なスケジューリングを行えることが報告されている [8]．

ジョブスケジューリング手法 NETG では，計算時間とは別にネットワーク Gantt 図によるデータ転送時間の予定を考慮するため，サーバでの計算時間とデータ転送時間の比が同程度，あるいはデータ転送時間が支配的となる問題で，かつサーバへのデータ転送が完全に終了してから計算が開始される問題を対象とする．

しかし，データ転送時間がローカルでの計算時間を上回ってはリモート実行を行う意味がない．現在の広域ネットワーク環境においてこれらの条件を満たす問題として，Linpack ベンチマークがあげられる [1]．ただし，リモート実行により性能向上が得られるのは問題サイズがある程度大きな場合である．

従って，NETG では Linpack と同様に，問題サイズを n とした場合にデータ転送量が $O(n^2)$，計算量が $O(n^3)$ で，n がある程度大きな問題を対象とする．行列積や逆行列を用いる計算もこの種の問題としてあげられ，対象を限定しても NETG の適用範囲は広い．また，ネットワーク性能の向上に伴い，対象となる問題の範囲も広がることが期待される．

4.2 ジョブスケジューリング手法：NETG

NETG では，ネットワーク Gantt 図，サーバの使用状況の予定を表現する Gantt 図，ルーティングプロトコルの OSPF を用いる．サーバが単純なラウンドロビンによって CPU 割り当てジョブを選択している場合，複数のジョブが CPU のクロック数を等分して使用するとモデル化する．これにより，ネットワーク Gantt 図におけるリンクの帯域幅をサーバの計算性能に置き換えることで，ネットワーク Gantt 図と同様にサーバの Gantt 図を描くことができる．

また，OSPF から情報を取得することで，クライアント・サーバ間のパスが一意に求まる．

以上のことを前提とし，ジョブの平均応答時間を最小にするというポリシに沿った，以下のジョブスケジューリング手法 (NETG) を提案する．

(1) クライアントが発行したジョブを処理できるサーバを全て抽出する．
(2) (1) で抽出したサーバとクライアントの全組み合わせに対し，OSPF からパスを抽出する．
(3) (1) のサーバそれぞれに対し，発行されたジョブを割り当てた場合のネットワーク Gantt 図，及び，サーバの Gantt 図を作成し，全ジョブの処理時間の予定を算出する．
(4) (3) で算出した全ジョブの処理時間の平均が最小となるサーバを割り当てる．

4.3 有効性の検証実験

文献 9) であげられている 6 つの基本的なジョブスケジューリング手法のうち，シミュレーションと実環境における実験において最も良い結果を残した **LOTH**，**LOTH+LT** について，机上でのシミュレーションにより NETG との比較を行う．

4.3.1 使用する問題

対象とする問題の 1 つである，LU 分解を用いた n 次の連立一次方程式の求解問題を用いる．これを表 1 の実験環境で実行した結果と，リモートで実行する場合に最低限必要とされるデータ転送量を表 2 に示す．

リモートで n 次の連立一次方程式を LU 分解を用いて解くためには，n 次の正方行列と n 次のベクトルを転送する必要がある．また，結果として n 次の

表 1　計算に用いた実験環境

	CPU[MHz]	メモリ[MB]
Ultra10	UltraSPARC-II 440	256
PC	PentiumIII 600	256

表 2　n 次の連立一次方程式の求解問題の実行時間とデータ転送量

n	実行時間 [sec]		データ転送量 [MB]
	Ultra10	PC	
1024	22.8	27.5	2.4
2048	199.5	240.3	8.8

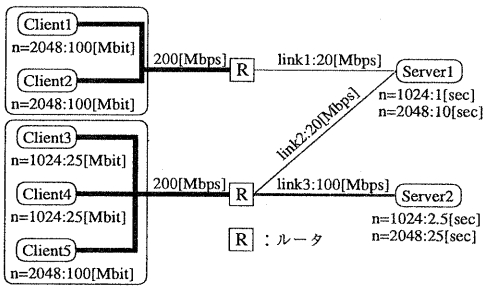


図 4　広域ネットワーク環境 (2)

表 3　*Server*1, 2 の計算性能とデータ転送量の仮定

n	実行時間 [sec]		データ転送量 [Mbit]
	Server1	Server2	
1024	1.0	2.5	25
2048	10.0	25.0	100

ベクトルを転送する必要がある．データ転送には 1 文字が 1[byte] の文字型が使われ，文字と文字の区切りには空白が用いられると仮定する．表 2 のデータ転送量は，転送する行列とベクトルの要素が全て 1 桁の整数とした場合のものである．

4.3.2 実験環境

図 4 の環境を想定する．グローバルコンピューティングシステムを利用することで，処理時間がローカルの約 1/10 になることを想定し，*Server*1, 2 の計算性能はそれぞれ Ultra10 の 20 倍，8 倍とする．また，CPU 割り当ては単純なラウンドロビンを用いて行うと仮定する．n=1024, 2048 の連立一次方程式を *Server*1, 2 で解くのに要する時間，及び，その際のデータ転送量を表 3 に示す．

この環境において，*Client*1, 2, 3, 4 がそれぞれ n=2048, 2048, 1024, 1024 のジョブを発行し，それらは既に *Server*1 へ割り当てられ，時刻 0[sec] で図 5 のように処理中であると仮定する．このジョブをそれぞれ *job*1, 2, 3, 4 とする．ただし，*Server*2, *link*3 の Gantt 図は使用しないため省略した．この状態における *job*1, 2, 3, 4 の処理時間とその平均の予定を表 4 に示す．

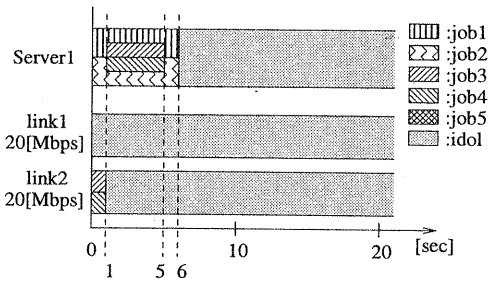


図 5　図 4 に対応する時刻 0[sec] のネットワーク Gantt 図

表 4　時刻 0[sec] での各ジョブの処理時間とその平均の予定

job	1	2	3	4	平均
処理時間 [sec]	22	22	6.5	6.5	14.25

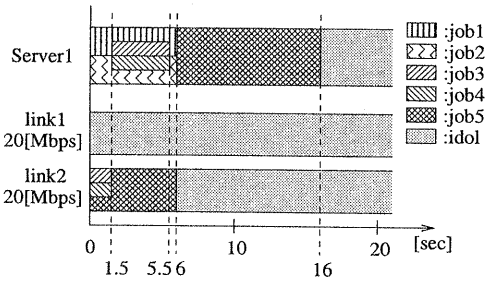


図 6　*Server*1 に割り当てた場合のネットワーク Gantt 図

表 5　*Server*1, 2 に割り当てた場合の各ジョブの処理時間とその平均の予定

job	1	2	3	4	5	平均
*Server*1[sec]	22	22	7	7	16	14.8
*Server*2[sec]	22	22	6.5	6.5	26	16.6

4.3.3 評　価

4.3.2 の状態で，時刻 0[sec] に *Client*5 から n=2048 のジョブ (*job*5) が発行された場合を考える．

今後，十分長い時間どのクライアントからもジョブが発行されないと仮定すると，*Server*1 に割り当てた場合は図 6 のようになり，*job*5 の処理時間は 16[sec] となる．ただし，*Server*2, *link*3 の Gantt 図は使用しないため省略した．*Server*2 と *link*3 はどのジョブにも使用される予定がないので，*Server*2 に割り当てた場合の *job*5 の処理時間は 26[sec] となる．*Server*1, *Server*2 に割り当てた場合の各ジョブの処理時間とその平均の予定は表 5 となる．

各ジョブの平均応答時間を最小にするには，*Server*1 に割り当てるのが妥当である．以下，各スケジューリング手法がどちらのサーバを割り当てるかを比較する．ただし，LOTH, LOTH+LT は以下の方法でジョブを割り当てるサーバを選択する．

表 6　時刻 0[sec] における $Server1,2$ の各パラメータ

$Server$	t	L	T	N	$W_b(0:0)$
1	10[sec]	2	10[Mbps]	2	6.6[Mbps]
2	25[sec]	0	100[Mbps]	0	100[Mbps]

LOTH：ジョブが発行された時点において，サーバで実行中のジョブ数 L とクライアント・サーバ間の通信スループット T，及び，そのサーバでのそのジョブの実行時間 t を求め，

$$t*(L+1)+[\text{データ転送量}]/T$$

が最小となるサーバを選択する．

LOTH+LT：LOTH の L,T に対して，ジョブ割り当て時の負荷を考慮する．具体的には，ジョブが発行された時点において，サーバに割り当てが行われているが，データ転送中で実行が開始されていないジョブ数 N とボトルネックとなるリンクの帯域幅 $W_b(0:0)$ を求め，

$$t*(L+N+1)+[\text{データ転送量}]/W_b(0:0)$$

が最小となるサーバを選択する．

時刻 0[sec] における 2048 次の連立一次方程式の求解問題に対する $Server1,2$ の $t,L,T,N,W_b(0:0)$ は表 **6** の通りである．これらのパラメータから，LOTH, LOTH+LT は $Server2$ を選択する．NETG は $Server1,2$ に割り当てた場合の，それぞれの Gantt 図を作成して全ジョブの処理時間の平均を算出し，その値が最小である $Server1$ を選択する．

このように，NETG だけが適切なスケジューリングを行える場合が存在する．これは，他のスケジューリング手法がジョブ発行時の資源情報のみしか考慮しておらず，予め予定されている資源の変動に対応できていないのに対し，NETG はその予定を考慮しているためである．従って，予定がそのまま確定すれば NETG は特定のスケジューリングポリシに関して他のスケジューリング手法に劣ることはない．本稿ではジョブの平均応答時間を最小にするポリシの例だけを取り上げたが，システムのスループットを最大にするなどといったポリシにも容易に変更できるため，状況に応じたスケジューリングポリシの切り替えが可能である．以上のことより，NETG の有効性が確認される．

5. まとめと今後の課題

本稿では，モデル化した広域ネットワーク上の各リンクの利用状況を表現するネットワーク Gantt 図を提案し，その正当性を示した．また，グローバルコンピューティングシステムにおけるジョブスケジューリング手法「NETG」を提案し，その有効性を確認した．

しかしながら，NETG は予定が食い違った場合に適切なスケジューリングを行うことができなくなる．輻輳が発生した場合のネットワーク Gantt 図の誤差は予定したデータ転送時間の 1% 程度であるが，輻輳が 1000[sec] 続くと誤差は 10[sec] になる．サーバでの 10[sec] の誤差は相当大きく，広域ネットワークと計算機に要求される精度の違いが問題となる．この問題に対する解決法としては，ネットワーク Gantt 図とサーバの Gantt 図を定期的に作り直すということが考えられる．しかし，スケジューリング自体のオーバヘッドとともに，作り直しのオーバヘッドは実装時に相当大きくなると予想される．

今後は，広域ネットワークモデルに輻輳の発生を取り込むことでネットワーク Gantt 図の精度を上げるとともに，より現実的な広域ネットワーク環境に対し，ns などを用いてシミュレーションによる評価を行う．

また，本稿では広域ネットワークの外乱については考慮に入れていないので，実環境での実験に向けて外乱の考慮も行っていく．

参 考 文 献

1) 竹房あつ子, 小川宏高, 松岡聡, 中田秀基, 佐藤三久, 関口智嗣, 長島雲兵: マルチクライアントによるネットワーク数値情報システム Ninf の性能, 並列処理シンポジウム JSPP'97 論文集, pp. 273–280 (1997).
2) Foster, I. and Kesselman, C.: Globus: A Metacomputing Infrastructure Toolkit., *International Journal of Supercomputer Applications* (1997).
3) Casanova, H. and Dongarra, J.: NetSolve: A Network Server for Solving Computational Science Problems, *Proceedings of Super Computing '96* (1996).
4) Legion. http://legion.virginia.edu/.
5) UCB/LBNL/VINT Network Simulator - ns (version2). http://www.isi.edu/nsnam/ns/.
6) J., Moy: OSPF Version 2, RFC 2328 (1998).
7) Ninf: Network Infrastructure for Global Computing. http://ninf.etl.go.jp/.
8) 竹房あつ子, 合田憲人, 小川宏高, 中田秀基, 松岡聡, 佐藤三久, 関口智嗣, 長嶋雲兵: 広域計算システムのシミュレーションによる評価 -Ninf システムの広域分散環境でのジョブスケジューリング実現に向けて-, 並列処理シンポジウム JSPP'98 論文集, pp. 127–134 (1998).
9) 竹房あつ子, 中田秀基, 合田憲人, 小川宏高, 松岡聡, 長島雲兵: Ninf システムにおけるジョブスケジューラの実装と予備的評価, 情報処理学会研究報告, Vol. 98, No. 72, pp. 73–78 (1998).