グリッドミドルウェアGlobus の資源探索と通信に関するオーバヘッドの定量的評価(3.2 第2回情報シナジー研究会，3．研究活動)

グリッドミドルウェア Globus の資源探索と通信に関するオーバヘッドの定量的評価

村田善智 † 稲葉勉 ‡ 滝沢寛之 ◇ 小林広明 ◇

† 東北大学大学院情報科学研究科 ‡NTT 東日本 研究開発センタ ◇ 東北大学情報シナジーセンター

■あらまし

我々は，一般ユーザがグリッドの技術を利用し，様々な計算機資源を自由に利用可能なサービスの実現を目標として，そのグリッドミドルウェアの研究開発を行っている．そこで，現在グリッドミドルウェアとして標準的に用いられている Globus Toolkit についての調査と性能評価を行った．その結果 Globus Toolkit は，ユーザ認証や情報検索，そして通信に大きなオーバヘッドが生じることがわかった．Globus Toolkit は，これらのオーバヘッドの影響を受けにくい大規模計算やバッチジョブ処理に適したグリッドミドルウェアである．しかし多数の計算機資源を管理する場合や，頻繁に通信を行うジョブや短時間で処理が終了するジョブを実行する場合などでは，Globus Toolkit のオーバヘッドの影響が増大するため，これを改善する必要がある．

1 はじめに

近年の高速な広域ネットワークの整備により，グリッドと呼ばれる新たな計算技術が注目されている．グリッドとは，利用者が必要とする計算機資源をその時々にネットワーク上で動的に収集し，仮想計算機として提供する計算基盤を実現する技術である．

グリッドの応用として”ユビキタスコンピューティング”の実現が挙げられる．これが実現されれば，ユーザは何処にいようとも，ネットワークに接続可能でさえあれば，様々な計算機資源をいつでも利用することが出来る．

グリッドは図 1 に示す階層化によって実現される．グリッドミドルウェアは，下位の物理的な計算機資源の収集と仮想化を行い，上位階層のアプリケーションに仮想計算機としてサービスを提供する．

グリッドに関する技術はグリッドの標準化団体である Global Grid Forum[1] を中心に，様々な研究・開発が行われている．特に Global Grid Forum では，Globus Alliance[2] により開発が行われている Globus Toolkit(以下 Globus) をグリッドミドルウェアの標準実装として取り上げている．このためグリッドミドルウェアとして，Globus が標準的に利用されている．

現在，科学技術計算やサーバシステムにおけるバックエンドなどの比較的大規模な計算を行う分野では，Globus を利用したグリッド環境の導入が盛んに行われている．しかし今後は，ユビキタスコンピューティングのような個人利用者レベルでのグリッド環境の実現が必要となる．このときに，多数の計算機の中から如何に効率よく資源検索を行うか，そして即応性などの様々な処理性能の要求をどの様に実現するかが解決すべき問題である．

本稿では，一般のユーザが利用するユビキタスコンピューティングを実現するためのミドルウェアの開発にあたり，現在標準的に用いられている Globus バージョン 2 について，その調査と性能評価を行う．

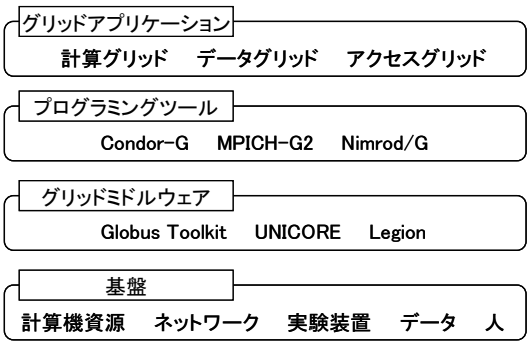

図 1　グリッドサービスレイヤ

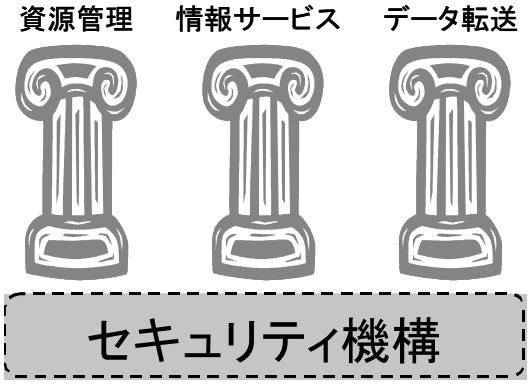

図 2　Globus の構成

2　Globus Toolkit

Globus は，複数の組織が管理運営している様々な計算機資源を利用してグリッド環境を構築し，組織間でお互いの資源を共有し合うことを目的として研究・開発が行われている．特に Globus は，グリッドに参加する計算機やユーザを一意に指定できる静的な環境を対象とし，その上で仮想計算機や仮想組織などを実現する．

図 2 に Globus の構成を示す．Globus はグリッド環境の構築に必要となる資源管理機構と情報サービス，データ管理の機能と，これらのサービスの基盤となるセキュリティ機構によって構成される．

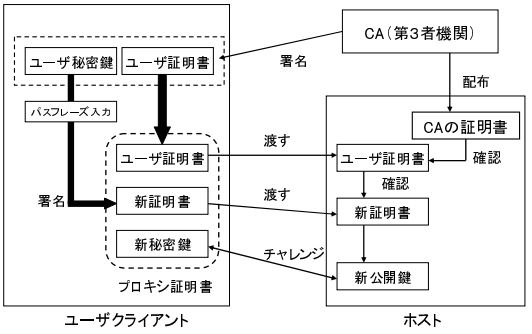

図 3　GSI による認証の仕組み

2.1　セキュリティ機構

Globus では，グリッド環境上での認証や認可，通信内容の保護になどのセキュリティ機構に，GSI(Grid Security Infrastructure) を使用する．GSI は Globus が提供するサービスの基礎となる重要な機構であり，シングルサインオンやジョブへのユーザ権限の委譲などの機能を提供する．

GSI による認証機構を図 3 に示す．GSI では計算機ごとのアカウントとパスワードによる認証ではなく，PKI(公開鍵基盤) による認証を行う．そのため Globus によるグリッド環境上では全てのユーザと計算機は，公開鍵と秘密鍵の組を持つ．特にユーザの秘密鍵は盗用を防ぐため，パスフレーズにより暗号化される．また，公開鍵の正当性を示すために，第 3 者機関である CA(認証局) により署名された電子証明書を持つ．この証明書をネットワークで流通させることで，集中型の認証サーバを持たなくとも任意の 2 者間での認証を実現する．GSI では証明書に加えて，パスフレーズの入力なしで秘密鍵を利用できるプロキシ証明書を使用する．プロキシ証明書とは，ユーザの秘密鍵で署名された一時的な証明書と秘密鍵の組である．プロキシ証明書に署名できるのはユーザ秘密鍵のパスフレーズを知る正規のユーザのみであり，このプロキシ証明書を用いてユーザ認証が行われる．ユーザはグリッド環境へのログインの際，一

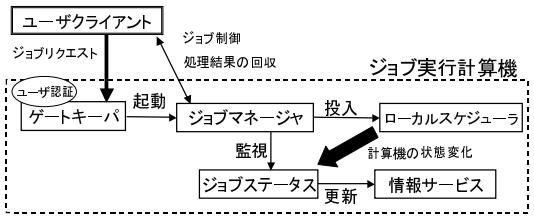

図 4　GRAM によるジョブ実行

度だけパスフレーズを入力してプロキシ証明書を作成すれば，パスフレーズ入力なしでグリッド環境上の資源を予め自分に与えられた権限内で自由に利用することが出来る．これが GSI の提供するシングルサインオンの機能となる．

2.2　資源管理機構

Globus における資源とジョブの管理は，GRAM(Globus Resource Allocation Manager) サービスにより行われる．GRAM は計算機ホストでのジョブ実行の仕組みと，資源を利用させるための標準化されたインターフェイスを提供する．

GRAM によるジョブ実行の仕組みを図 4 に示す．ユーザはまず利用したい計算機のゲートキーパ・デーモンに対してリクエストを出す．リクエストを受けたゲートキーパは，GSI によるユーザ認証を行う．認証に成功すると，ゲートキーパはユーザから RSL(Resource Specification Language) で記述されたジョブの処理内容を受け取るとともに，計算機ホストの grid-mapfile と呼ばれるマッピングファイルを参照して，リクエストを出したユーザを GRAM サーバのローカルユーザに関連付けして，ユーザの認可を行う．最後にゲートキーパはローカルユーザ権限でジョブマネージャと呼ばれるプログラムを起動し，RSL で記述されたジョブを渡す．ジョブマネージャは RSL ファイルに従ってジョブの実行・管理を行い，ユーザからのジョブ制御リクエストの受け付や，ジョブの実行結果の返答を行う．

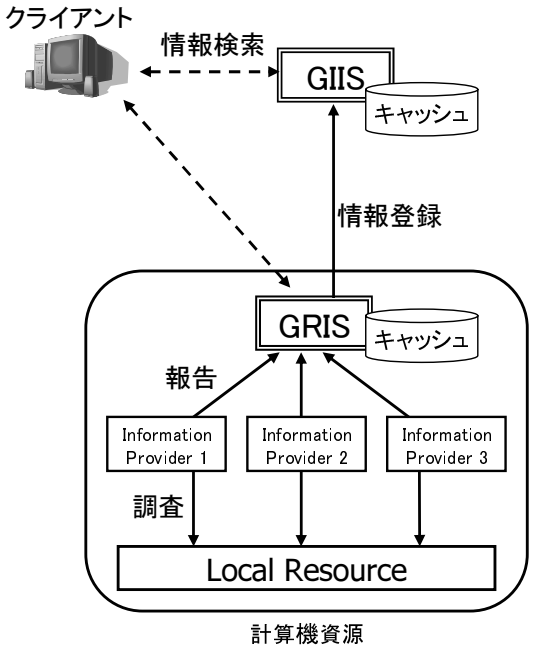

図 5　MDS による情報検索の仕組み

2.3　情報サービス

Globus における計算機の資源情報の収集は MDS(Monitoring and Discovery Service) により行われる．MDS は LDAP(Lightweight Directory Access Protocol) に基づいて実装されており，グリッド環境で利用される各種の資源情報の定義と，その収集のための標準的なインターフェイスを提供する．

図 5 に示すように MDS の機能はさらに，GRIS(Grid Resource Information Service) と GIIS(Grid Index Information Service) の 2 つのサービスにより構成される．GRIS は各計算機のローカルな資源情報のみを扱うサービスであり，Information Provider(IP) と呼ばれる機能を呼び出して計算機の各種情報の収集を行い，その結果をキャッシュに保持する．一方 GIIS は，ユーザクライアントに対して情報検索のためのディレクトリサービスを提供し，下位の GRIS または GIIS より報告された資源情報の管理を行う．

MDS による情報検索の流れを図 6 に示す．

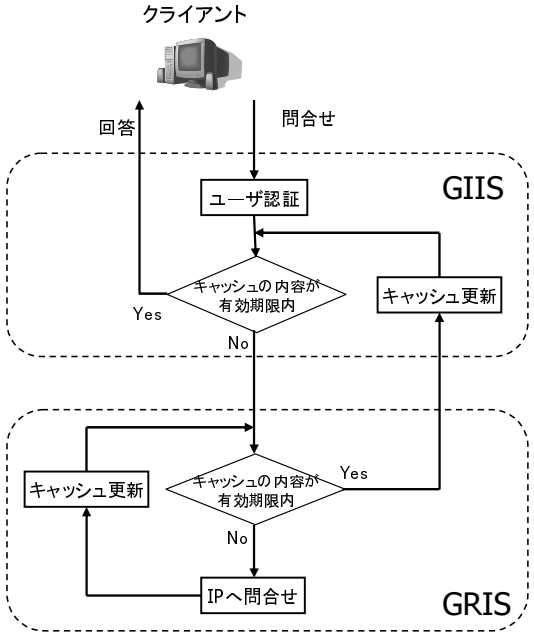

図 6　MDS による情報検索の流れ

ユーザクライアントはまず GIIS サーバに対して，条件を指定して検索のリクエストを出す．GIIS は GSI によるユーザ認証を行った後，自身のキャッシュ内容が有効期間内にあるか調べる．期間内であれば，キャッシュ内容から条件に一致するものをユーザに対し返答する．もし有効期間を過ぎていた場合には，下位の GRIS や GIIS に対して最新の資源情報を要求し，キャッシュの更新を行う．要求を受けた GRIS はまず自身のキャッシュ内容の有効期間を調べる．有効期間内であればその情報を上位の GIIS に対し返答し，期間外であれば IP を起動し，情報を収集してキャッシュの更新を行う．

2.4　データ管理

Globus ではデータ管理機能を用いて，ファイルの管理と転送を行う．このデータ管理における主要コンポーネントが，GridFTP サービスとなる．GridFTP は，従来の FTP に対して GSI による認証などを拡張した実装となり，グリッド環境上でのセキュアなデータ転送サービスを提供する．

3　Globus Toolkit の評価実験

前節で示したとおり Globus はグリッドを実現するために必要な，柔軟で強力な機能を提供する．しかしその処理には大きなオーバヘッドが生じるものと予想される．本節では検証実験を行うことで，この Globus のオーバヘッドがグリッド環境全体に及ぼす影響の範囲や度合いの定量的な評価を行う．本稿では，1) MDS による情報検索時間，2) Globus を用いた遠隔ジョブ実行時間，3) Globus 上でのプロセス間の通信時間の計測を行うことにより，Globus の情報検索と通信に生じるオーバヘッドの評価を行った．

3.1　MDS による情報検索時間

MDS による計算機資源情報の検索に要する時間を計測し，性能の評価を行う．本実験を実施するため，処理能力の異なる 8 台の PC を用いたグリッド環境を構築した．各 PC は 10/100Mbps 混在の Ethernet によるネットワークで接続し，同一セグメント内に配置した．OS には Linux を用い，Globus のバージョンは 2.2 を利用した．MDS の階層構造による影響を調べるため，論理構成を 3 階層とした．MDS の論理構成を図 7 に示す．また，GIIS および GRIS のキャッシュの有効期間をそれぞれ 30 秒に設定し，キャッシュの影響を評価する．

実験は MDS 論理構成の最上位の GIIS に対し，管理下の計算機の全ての資源情報を収集するのに必要な時間の計測を行う．時間の計測には，Linux の time コマンドを用いる．また，GIIS および GRIS のキャッシュによる高速化の効果を調べるため，情報検索を行った時刻も合わせて記録し，検索要求と次の検索要求との間隔を明確にしている．本実験では MDS による検索時間を計測することが目的であるため，GSI による認証を行わない匿

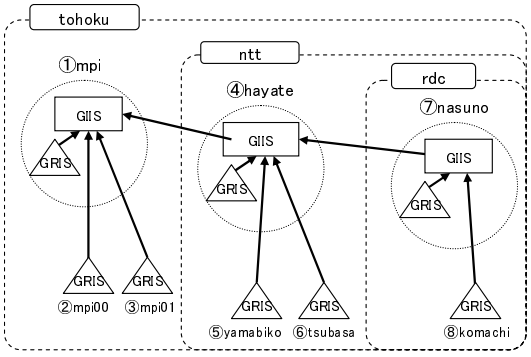

図 7　MDS 論理構成

表1　MDS問い合わせ時間

No	検索時間(s)	検索時刻
1	7.242	14:45:46
2	3.817	14:47:17
3	9.825	15:08:33
4	3.693	15:19:44
5	3.750	15:20:31
6	1.661	15:28:13
7	0.673	15:28:16
8	1.690	15:29:05
9	0.673	15:29:09
10	1.670	15:30:17
11	9.118	15:57:21
12	1.662	15:57:55
13	1.073	16:18:53
平均	3.581	–

名モード (anonymous mode) で計測を行う．計測結果を表 1 に示す．

実験結果より，MDS による情報検索に要する時間は，1 秒以内，1〜4 秒，7 秒以上の 3 通りが確認できた．特に表 1 において強調した 7，9 回目の検索では，前回の検索の直後に再検索を行っているため，GIIS のキャッシュに保持されている情報は全て有効期間内であり，検索時間は短時間となっている．また，前回の検索から時間が経つほど，キャッシュに保持される情報が古くなり，下位の GIIS や GRIS に対する問い合わせが生じる．そのため全体検索に要する時間が増加し，最大で 10 秒近く要した．

本実験では，キャッシュを用いた MDS の仕組みにより，最大で 15 倍の情報検索の高速化が確認できた．このキャッシュの存在により，MDS は高速に応答できるが，キャッシュの有効期間を長くすると，キャッシュ情報が古くなり計算機の現在の状態と一致しないものとなってしまう．しかし，キャッシュの有効期間を短くしてしまうと，問い合わせの度に全検索が行われる可能性が高くなり，結果として通信トラフィックを増大させることが問題となる．さらに本実験のような小規模でローカルなグリッド環境ですら，情報の検索に最大で 10 秒程度要するという結果を考えると，一般利用者が利用するような，インターネット環境上での多数の計算機情報の検索の場合，MDS による情報検索はオーバヘッドが非常に大きなものとなることが予想される．

3.2　遠隔ジョブ実行に生じるオーバヘッド

ユーザがグリッドを利用して実際にジョブを実行させることを想定して，Globus によるジョブ実行に要する時間の計測を行い，Globus を用いたジョブ実行におけるオーバヘッドの評価を行う．このときのオーバヘッドは，GSI によるユーザ認証によって生じるものとみなすことが出来る．前述の実験環境において，printf による文字出力のみを記述した C 言語の実行ファイルを，Globus を用いて遠隔ホストで実行させ，処理が終了するまでの時間を Linux の time コマンドを利用して計測する．また比較対象として，リモートシェル (rsh) を利用して同じジョブを実行したときの処理時間の計測も行う．両者の比較により，GSI によるユーザ認証のオーバヘッドの評価を行う．計測結果を表 2 に示す．

表 2 の結果より，rsh ではジョブが常に 0.1 秒程度で終了したのに対し，Globus ではおよそ 2.5 秒で終了するパターンと 30 秒以上を要するパターンの 2 通りの結果が得られた．Globus では，ジョブを実行する度に GSI によるユーザ認証のオーバヘッドが生じる．こ

表2　遠隔ジョブ実行時間

No	Globus(s)	rsh(s)
1	36.988	0.102
2	2.429	0.099
3	2.367	0.097
4	2.740	0.102
5	2.642	0.101
6	32.715	0.101
7	32.860	0.101
8	32.950	0.098
9	3.118	0.101
10	32.747	0.103
11	2.588	-
12	2.712	-
13	32.662	-
14	2.720	-
15	2.437	-
16	2.521	-
17	2.498	-
18	2.701	-
19	2.594	-
20	2.502	-
21	2.464	-
22	33.543	-
23	2.449	-

のオーバヘッドが，Globus を介したジョブの実行時間が rsh と比較して 25〜300 倍以上に増加した要因と考えられる．

元々 Globus は，バッチジョブや大規模なジョブの実行を想定して開発されているため，このオーバヘッドは問題視されなかった．しかし，本実験条件のように処理時間が短いジョブの実行においては，ユーザ認証によるオーバヘッドが支配的になり，ジョブの実行効率を低下させる．また，プログラムの応答性が求めるような場合には，応答性の低下によりユーザの利便性を低下させる．このため，短いジョブの実行効率や即時性が求められる環境下では，Globus によって生じるオーバヘッドが性能に大きく影響する可能性があることが明らかになった．

本実験では，ジョブの実行時間が 2 通り計測される原因を解明することはできなかった．この原因について，今後さらに詳細に調査する予定である．

3.3 プロセス間通信に生じるオーバヘッド

実行中の並列プログラムのプロセス間通信に対して，Globus が与える影響を実験により評価する．東北大学情報シナジーセンターの並列計算機 NEC TX7/AzusA を用いて評価実験を行う．AzusA は 800Mz で動作する Itanium プロセッサを 1 ノードにつき 16 個搭載する．ノード内の各 CPU は 4.2Gbyte/s の高速スイッチで相互接続され，ノード間は 1Gbps のイーサネットにより接続されている．MPI[3] を利用した並列プログラムを用意し，2 つのプロセス間のメッセージ転送に要する時間を計測する．

本実験では，1) AzusA 用に最適化された NEC 製 MPI ライブラリによる共有メモリ間転送を用いた場合，2) MPICH[4] による共有メモリ間転送を用いた場合，3) 同一ノード内で MPICH による TCP 通信を利用した場合，4) 2 ノード間で MPICH による TCP 通信を利用した場合，5) 同一ノード内で MPICH-G2[4] を用いた場合のデータ転送時間を計測する．MPICH-G2 は Globus を利用するように MPICH を修正したライブラリであり，実験条件 3) と 5) の結果を比較することにより Globus 自体の処理のオーバヘッドを評価する．

評価には，式 (1) で表される通信時間のモデル [5] を用いる．

$$t(m) = t_0 + m/r_\infty \qquad (1)$$

ここで，m はメッセージサイズ，$t(m)$ はメッセージ転送に要する時間，t_0 はスタートアップ時間，r_∞ はメッセージサイズが無限大の時のバンド幅を示す．

各実験条件について，メッセージサイズを変化させてデータ転送時間を測定した結果を図 8 に示す．また，図 8 の結果から式 (1) 中のスタートアップ時間およびバンド幅を求め

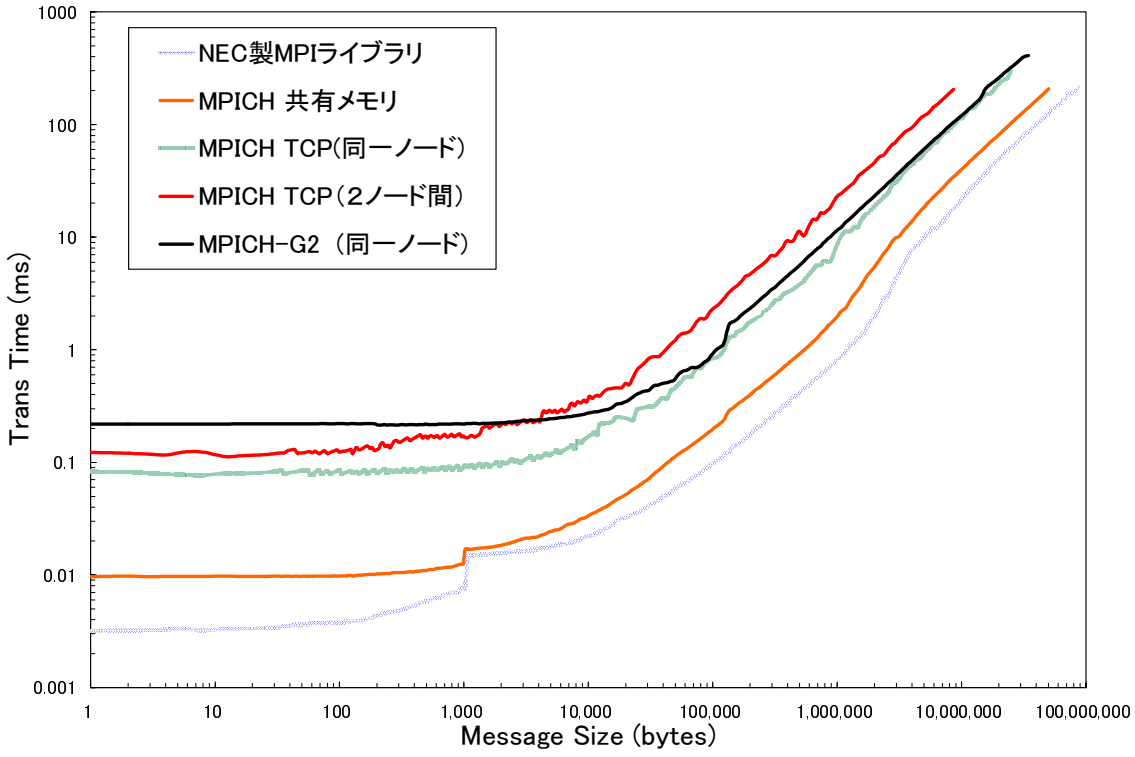

図 8　MPI の通信性能の変化

表3　各条件でのスタートアップ時間およびバンド幅

	スタートアップ時間(μs)	バンド幅(Mbytes/sec)
NEC製MPIライブラリ	3	405
MPICH 共有メモリ	9	245
MPICH TCP(同一ノード)	80	88
MPICH TCP(2ノード間)	120	42
MPICH-G2(同一ノード)	210	80

た結果を表 3 に示す．図 8 および表 3 より，NEC 製 MPI ライブラリを用いたときスタートアップ時間が最小となり，バンド幅は最大を示すことが分かる．MPICH を用いた場合，NEC 製 MPI ライブラリを用いた場合に比べて性能が低下し，MPICH の共有メモリ間転送の場合でもスタートアップ時間は 3 倍程度，バンド幅は 60 %程度になる．同一ノード内で TCP 通信を用いる実験条件 3) ではさらに性能が低下し，NEC 製 MPI ライブラリを利用する実験条件 1) と比較すると，スタートアップ時間は約 25 倍まで増加し，バンド幅は 20 %まで減少する．同様に，実験条件 3) を MPICH の共有メモリ間転送を用いる実験条件 2) と比較すると，スタートアップ時間は 9 倍に増加し，バンド幅は 33 %程度まで低下することが確認できる．この結果から，NEC 製 MPI ライブラリの代わりに MPICH の共有メモリ間転送を利用することによるオーバヘッドよりも，通信に TCP を利用することによるオーバヘッドの方が非常に大きいことが分かる．同様に，実験条件 3) と 5) との比較から，Globus 自体の処理のオーバヘッドによりスタートアップ時間は約 2.5 倍程度に増加するが，バンド幅は 90 %程度までしか減少しないことが確認できる．

本実験より，Globus にはスタートアップ時間のみを一定時間増大させるオーバヘッドが存在することが判明した．このオーバヘッドによるスタートアップ時間の増加はデータ転送時間全体の 60 %以上を占めている．このため，プロセス間通信を頻繁に要求する並列計算では Globus の処理時間が支配的になり，処理効率が大きく低下する．一方 Globus のオーバヘッドはバンド幅にはほとんど影響を与えない．このような条件下で通信を行う場合，転送するメッセージが小さなサイズで

あるほど，全体の転送時間に占めるスタートアップ時間の割合が大きくなる．以上の結果から，Globus を利用してデータ転送を行う場合には，メッセージサイズを出来るだけ大きくして，スタートアップ時間の影響を小さくして転送効率を高くする必要があることが明らかになった．

一方，実験条件 3) と 4) との比較から，ネットワークの物理層の通信性能がボトルネックとなり，スタートアップ時間が 1.5 倍に増大し，バンド幅は約 50 %まで低下することが確認できる．そのため，2 ノード間での MPICH-G2 を利用したデータ転送では，同一ノード内で MPICH-G2 を利用することによるオーバヘッドとネットワークの物理層の通信性能の限界から，大幅なセットアップ時間の増加とバンド幅の低下を引き起こすと考えられる．特に，Globus を用いて広域に分散した計算機資源を利用した並列計算を行う場合，ネットワークの性能のばらつきや負荷変動による通信性能の低下により，プロセス間のデータ転送時間がさらに増大すると考えられる．

4 まとめ

本稿では，現在標準的に用いられているグリッドミドルウェアである Globus のバージョン 2 について調査，および評価を行った．その結果，Globus は GSI という強力なセキュリティ機構の上で，グリッド環境を構築するための様々なサービスが提供されていることがわかった．しかし実験結果より，Globus は情報検索やジョブ実行に大きなオーバヘッドが生じるため，多数の計算機資源からの資源情報の検索や，ユーザへの即応性が求められるようなプログラムを実行するような場合には，Globus は適さないことが判明した．また，Globus は大きな通信遅延を生じさせるため，通信を頻繁に行うようなアプリケーションの実行にも適さないことがわかった．

現在ではグリッド環境構築には Globus の存在は欠かせないものとなっている．その利用分野としては，ハイパフォーマンスコンピューティングやバッチジョブの実行など，ユーザとの対話性が必要とならない大規模計算が中心となっている．しかし動的に変化する多数の計算機を管理する場合や，対話性や即応性などの様々なユーザの要求に対応する場合，Globus を用いたグリッド環境では，その大きなオーバヘッドが問題になることが予想される．

今後は，Globus が上記の様な，より一般性を持つ計算機環境に拡張可能であるか，さらに詳しい検証を行う必要がある．このとき問題になるのは，処理のオーバヘッドが増大することであり，そのオーバヘッドが生じる原因を解明し，改善を行う必要がある．もし，オーバヘッド自体を改善することが出来ない場合には，そのオーバヘッドの影響を最小にするような仕組みを考案する必要がある．例えば，ジョブのプロセス間通信のオーバヘッドが問題であれば，スケジューリングによりジョブを最適な組み合わせの計算機に割り当てることで，オーバヘッドの影響を最小にすることが期待できる．一方，計算機資源を静的にしか管理できない Globus の情報検索機構では，動的に変化する計算機資源を効率よく管理することが難しい．従って，現在の仕組みとは異なる新たな情報検索機構を考案する必要がある．

参考文献

[1] Global Grid Forum, http://www.gridforum.org/

[2] The Globus Alliance, http://www.globus.org/

[3] Message Passing Interface Forum, http://www.mpi-forum.org/

[4] MPICH, http://www-unix.mcs.anl.gov/mpi/mpich/

[5] Kai Hwang, Zhiwei Xu,
"Scalable Parallel Computing,"
WCB/McGraw-Hill, 1998