社団法人 情報処理学会　研究報告
IPSJ SIG Technical Report

2004－ARC－157　(4)
2004－HPC－ 97　(4)
2004／3／1

Ninf-G version2の実装および性能評価

武宮 博†，田中 良夫†，中田　秀基†‡，関口 智嗣†

Ninf-G version2は，Gridプログラミングモデルの一つであるGridRPCの参照実装であり，広域に分散した複数台のクラスタから構成される大規模Grid環境上でアプリケーションを効率よく実行することを目的とする．関数ハンドル同時生成機能やリモートオブジェクトを実装することで，遠隔手続き呼び出しに伴う起動コストや通信コストの低減を図るとともに，ハートビート機能，関数ハンドル作成タイムアウト機能，サーバ属性の個別設定機能を提供することで，非均質，不安定で動的に変化するGrid環境への対応を図っている．4台のクラスタ計300プロセッサから構成されるGridテストベッド上でNinf-G version2の実行性能を測定した．その結果，関数ハンドル同時生成機能がプログラム起動コストの低減に有効であること，およびNinf化されたシミュレーションが大規模グリッド環境で効率的に実行可能であることがわかった．

Implementation and Evaluation of Ninf-G version 2

HIROSHI TAKEMIYA†, YOSHIO TANAKA†, HIDEMOTO NAKADA†‡ and SATOSHI SEKIGUCHI†

A high performance GridRPC system called Ninf-G version 2 has been developed and its performance was evaluated. Ninf-G2 aims to enable applications to run efficiently on a large scale Grid environment which consists of clusters widely distributed over a network. It tries to reduce costs for start-up and communication by simultaneous function handles creation function and remote object mechanism. In addition,it tries to cope with heterogeneous, unstable, and dynamically varying grid environment by heart-beat monitoring function,timeout mechanism in creating function handles, and methods to specify server-dependent attributes. Performance of Ninf-G version 2 was evaluated on a grid testbed which consists of 4 distributed clusters with totally 300 processors. It was found that simultaneous function handle creation function is effective to reduce starting-up cost and "Ninfied" simulation program could be executed efficiently on such an large scale grid environment.

1. はじめに

Grid環境における遠隔手続き呼び出しを支援するプログラミングモデルであるGridRPCは，パラメータサーベイ等多くの科学，工学分野において適用されているタスク並列型の処理を容易に記述できるという特徴を持ち，Grid環境における有効なプログラミングモデルの一つとして期待されている[1)2)]．Grid技術の標準化を進めるGlobal Grid Forum(GGF)[3)]でも，GridRPC Working GroupにおいてGridPRC APIの標準化に関する議論が進められている[4)]．

GridRPCに関する研究は，広域分散コンピューティングの実現を目的としたNinfプロジェクト[5)]，NetSolveプロジェクト[6)]等に端を発し，これまでにNinf-G[7)]，NetSolve[8)]，DIET[9)]，OmniRPC[10)]等，いくつかのGridRPCシステムが実装されてきた．それらを用いた性能評価や実用性評価[11)12)]により，数十プロセッサ規模のタスク並列処理を効率的に実行できることは明らかになってきた．しかし，より大規模な処理，すなわち数百～数千プロセッサ規模を用いた処理を安定して実行できるのか，あるいは実行効率がスケールするのかといった問題はまだ論じられていない．

Ninf-G version 2(以下，Ninf-G2)は，数サイト～十数サイトに分散配置された数十～数百プロセッサ規模のクラスタにより構成されるGrid環境におけるアプリケーションの効率的な実行を目的として開発されたシステムである．我々が対象としている環境は，計算機，ネットワーク等が非均質であり，その稼動状況は不安定で，複数の利用者によって共有されているという特徴を持つ．したがって，その環境上で動作するシステムには，それらの特徴に柔軟に適応できる機能が求められる．さらに，その環境上でアプリケーションを効率的に実行するためには，処理の高性能性，スケーラビリティが求められる．我々が開発を進めているNinf-G2には，これらの要件に対処するために種々の機能が実装されている．本稿では，それらの機能について紹介する．

また，Ninf-G2を用いた大規模タスク並列処理の可能性を探るために，広域に分散配置された4クラスタ計300プロセッサから構成されるGridテストベッドを利用してNinf-G2の基本性能を測定するとともに，典型的なタスク並列型アプリケーションである気象シミュレーションプログラム[13)]を用いてNinf-G2の実行性能を評価した．

†産業技術総合研究所
National Institute of Advanced Industrial Science and Technology
‡東京工業大学
Tokyo Institute of Technology

本稿では，次章でNinf-G2の概要を紹介した後，3章でNinf-G2の実装方針およびその方針に基づいて実装された機能について述べる．4章では，Ninf-G2の通信性能，プログラム起動コスト等基本性能実験結果について報告するとともに，気象シミュレーションを用いて行った性能評価結果について述べる．最後に現状と今後の課題についてまとめる．

2. Ninf-G2概要

Ninf-G2はGridプログラミングモデルの一つであるGridRPCの参照実装であり，Ninf-Gの開発経験およびNinf-Gを用いたアプリケーションによる性能評価[14)15)]から得られた知見に基づいて設計[16)]，開発されたものである．

Ninf-G2の目的は，数サイト～十数サイトに分散配置された数十～数百プロセッサ規模のクラスタにより構成されるGrid環境上でアプリケーションの効率的な実行を可能とすることである．近年の急速なクラスタ構築技術の進展，普及により，安価かつ高性能なクラスタを利用した並列計算が一般的になってきている．Ninf-G2は各サイトに散在するこれらのクラスタを連携させ．一つの巨大な仮想計算機として利用し大規模なタスク並列型プログラムを実行可能とすることを狙う．

Ninf-G2はローカルな環境にある計算機（クライアント計算機）から，リモートに分散配置された計算機（バックエンドサーバ）上に実装されたプログラムをRPCの形式で呼び出し，実行する機能を提供する．クライアント計算機上で実行され，遠隔手続きを呼び出すプログラムをクライアントコンポーネントと呼ぶ．クライアントコンポーネントから呼び出されるバックエンドサーバ上のプログラムを遠隔実行プログラムと呼ぶ．遠隔実行プログラムは関数ハンドルという形で抽象化され，クライアントコンポーネントからの呼び出しは関数ハンドルに対する呼び出しという形で実現される．

Ninf-G2は，Grid環境構築のためのインフラとして最も普及しているGlobus Toolkitのコンポーネントを基盤として利用している(図1)．具体的には，(1) MDSを用いて遠隔実行プログラムの呼び出し情報(プログラムのパス情報や引数情報)を管理する．これによりクライアントコンポーネントが個々に呼び出し情報を管理しなくても情報を取得できる．(2) GRAMを介して遠隔実行プログラムの起動を行う．これにより，リモートに配置されたクラスタのバッチキューシステムをセキュアに利用できる．(3) Globus IOを用いて計算機間のデータ送受信を行う．これによりGSIに基づくセキュアな通信が実現される．

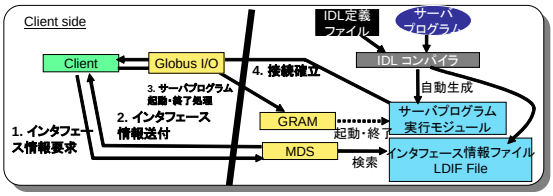

図1.Ninf-G2ソフトウェアアーキテクチャ

Globus Toolkit上に実装されたNinf-G2のAPIにより，アプリケーションプログラマはGlobus Toolkitの低レベルなAPIによる複雑なプログラミングを行うことなく標準技術に基づくGridアプリケーションを開発でき，多くのサイトで動作させることができる．

3. Ninf-G2の実装

3.1 実装方針

前節で述べた目的を達成するためには，資源の非均質性，動的な変化，不安定性といった特質を持つGrid環境への柔軟な適応性と，処理の高効率性，スケーラビリティの実現が求められる．以下に，Ninf-G2開発に際して考慮した項目を挙げる．

(1) Grid環境への柔軟な適応性

Grid環境上のクラスタやそれらを接続するネットワークは，多数の利用者によって共有されており，その負荷は動的に変動する．また，Grid環境が大規模，広域になればなるほど，構成要素であるクラスタやネットワークの一部がダウンし利用不可能になることが頻繁になる．このような動的な資源の変化，不安定性に対処できる必要がある．

また，Grid環境上に存在するクラスタは，機種や規模，運用されているバッチシステムが異なるだけでなく，デフォルトで設定される環境変数，遠隔実行プログラムの実行パスといった実行環境も異なっている．さらに，サイトによってセキュリティポリシーが異なるため，ネットワーク接続を許可されるポート番号，利用可能なプロトコル，認証方法なども異なる可能性がある．このようなGrid環境の非均質性にも適応できなければならない．

(2)高性能性，スケーラビリティの実現

我々のターゲットとするアプリケーションは，数百～数千個の遠隔実行プログラムを起動し，データの授受を行う．したがって，遠隔実行プログラムの起動/終了コストや通信コストの低減等を図って効率的な処理を実現し，大規模な処理を可能にする機能の実現が求められる．

3.2 Ninf-G2の実現機能

3.1で述べた実装方針に基づき，Ninf-G2では以下のような機能を実装，提供している．

(1) Grid 環境への柔軟な適応性

関数ハンドル作成タイムアウト機能

複数の利用者が計算機にアクセスしている状態では，多数のジョブがバッチシステムに投入され，ジョブを投入しても長時間にわたって実行が開始されない可能性がある．このような状況に対応するため，Ninf-G2では，タイムアウト値をコンフィグレーションファイルに設定することで，自動的に遠隔実行プログラムの起動がキャンセルされる．

ハートビート機能

不安定なGrid 環境で長時間にわたってアプリケーションを実行する場合，定期的に遠隔実行プログラムの実行状況をチェックし，正常に動作していることを確認できることが望ましい．Ninf-G2では，遠隔実行プログラムから定期的にクライアントコンポーネントに対しハートビートを発行することにより，遠隔実行プログラムの実行状態を確認できる機能を提供している．

コールバック機能の提供

ハートビート機能は遠隔実行プログラムの動作確認を可能にしているが，処理によってはより細かな制御，例えば計算の途中経過を出力させ，その結果を保存しておいたり，計算機の負荷の増大等の原因により進行が思わしくない場合は実行を他の計算機資源に振りかえる等，プログラムの実行制御を行うことが必要な場合もある．しかし，GridRPC の枠組みでは遠隔実行プログラムが実行を終了するまでクライアントコンポーネントにアクセスする手段がないため，途中経過の出力は困難である．

Ninf-G2 では，クライアントコンポーネントがあらかじめ登録しておいた関数を遠隔実行プログラム側から呼び出すコールバック機能を提供することで，この機能を容易に実装可能としている．

通常のプログラムが関数呼び出しの際に関数ポインタを引数として指定すると，呼び出された関数内から渡された関数を呼び出すことが可能になるのと同様に，遠隔実行プログラムに渡す引数としてクライアントコンポーネントが関数ポインタを指定することで，遠隔実行プログラムからその関数を呼び出すことが可能になっている．

サーバ毎の属性設定機能

Ninf-G2 では，非均質性への対処としてデータ転送モード，通信路の暗号化手法，利用通信ポート，バックエンドサーバ上で有効なバッチシステム，設定したい環境変数等に対して種々のオプションを提供し，利用者がバックエンドサーバ単位で指定できるようになっている．例えば，バッチシステムの違いに関しては対応する GRAM の job manager の種類を，通信路の暗号化に関してはSSL を用いるかどうかを指定できる．これらの情報は，コンフィグレーションファイルに規定することで有効になる．

(2)高性能性，スケーラビリティの実現

リモートオブジェクトの実装

タスク並列型の処理では，遠隔実行プログラムに渡されるデータの大半が共通で，一部のパラメータのみが変化することが多い．このような処理では，遠隔実行プログラムに何度も同じデータを転送しなければならず，無駄な通信が発生する．遠隔実行プログラムが呼び出し終了後も起動を継続し状態を保持するようにし，さらに複数の関数呼び出しを受付可能にすれば，最初の呼び出し時のみ全てのデータを転送し，その後は変化するパラメータのみを別の関数呼び出しとして渡すことで，通信量を低減できる．Ninf-G2 では，そのような機能を持つ遠隔実行プログラムをリモートオブジェクトとして実装し，呼び出すことを可能にしている．

関数ハンドル同時作成機能

クラスタを利用してタスク並列型処理を実行すると，クラスタ上に多数の遠隔実行プログラムが起動される．遠隔実行プログラムの起動はGRAMを介して行われるが，GRAM の呼び出しには，認証やバックエンドサーバ上のバッチ処理システムへのジョブ投入のためのコストが発生する．したがって，個々の遠隔実行プログラムの起動のたびに GRAM を呼び出すと起動コストが増大する．Ninf-G2 では，GRAM の提供する複数のジョブを一度に起動する機能を利用し，一回のGRAM呼び出しでまとめて複数の遠隔実行プログラムを起動する機能を実装している．

MDS バイパス機能の提供

MDS による遠隔実行プログラムの呼び出し情報の検索コストは大きく，場合によっては数十秒程度かかってしまうことがある．また，MDS サーバの動作は不安定で，常にインタフェース情報が取得できるとは限らない．そこで，Ninf-G2 では MDS を利用した引数情報取得に加え，MDS を利用せずクライアント計算機上に格納されたローカルなファイルから引数情報を取得する機能も提供することで引数情報取得コストの低減を図っている．

4. 性能評価

3 つのサイトに分散配置されたクラスタを用いて Ninf-G2 の性能評価を行った．評価に際しては，通信性能，遠隔実行プログラム起動コストといった Ninf-G2 の基本性能を測定するとともに，典型的なタスク並列プログラムである気象シミュレーションプログラムを用いて実行効率を評価した．

4.1 実験環境

表1．実験環境のネットワーク及び計算機特性

サイト名 (クラスタ名)	AIST		TITECH	KISTI
	(KOUME)	(UME)		
プロセッサ	Dual Pentium 1.4GHz	Dual Pentium 1.4GHz	Dual Athron 1.6GHz	Pentium 2.0 GHz
プロセッサ数	10	64	256	64
1ヶ月予報実行時間(sec)		36	35	42
TCP/IP Latency (msec)		0.04	1.5	17.2
TCP/IP Throughput (MB/sec)		116.0	9.3	1.1

実験には，産業技術総合研究所(以下，AIST)，東京工業大学(以下，TITECH)，韓国Korea Institute of Science, Technology and Information(以下，KISTI)の3サイトに設置された4台のLinuxクラスタを利用した．各クラスタ及びサイト間ネットワークの諸特性を表1に示す．

プロセッサ特性の測定では，ベンチマークプログラムとして用いる気象シミュレーションプログラムの実行性能を測定した．クラスタによって最大20%程度の処理時間差が存在している．

ネットワーク特性の測定では，クライアントをAIST KOUMEクラスタに固定し，他の3台のクラスタとの間でソケットを用いたping-pongプログラムを実行した．KOUMEクラスタとLAN接続されているUMEクラスタと比較して，TITECHクラスタは1/10倍程度，KISTIでは1/100倍程度のスループットとなっている．

4.2 Ninf-G2通信性能測定

Ninf-G2を利用したping-pongプログラムを実装し，Ninf-G2の通信性能を測定した．前節で述べたソケットによる通信実験と同様，クライアントをKOUMEクラスタに固定し，他の3台との通信時間を計測した．

図2は遠隔実行プログラムからクライアントコンポーネントへの規定長データの転送に要した時間と通信速度を示している．各クラスタとも転送データ長が1KB程度まではレイテンシが支配的であり，通信時間はほぼ一定である．通信性能は100KB程度まで増大するが，次第に頭打ちになり100KB以上ではほぼ一定になる．

UME及びTITECHクラスタにおいては送信データ長が1KBを越えたところで顕著な通信性能の伸びが観測されるが，これは以下の理由による．Ninf-G2における通信のベースとなるNinfパケットは，固定長のヘッダ部と可変長のデータ部から構成され，これらはglobus_io_writev()関数を用いて送信される．しかし，現状ではこの関数が正常に動作しないため，暫定的にglobus_io_write()関数を用いて両部

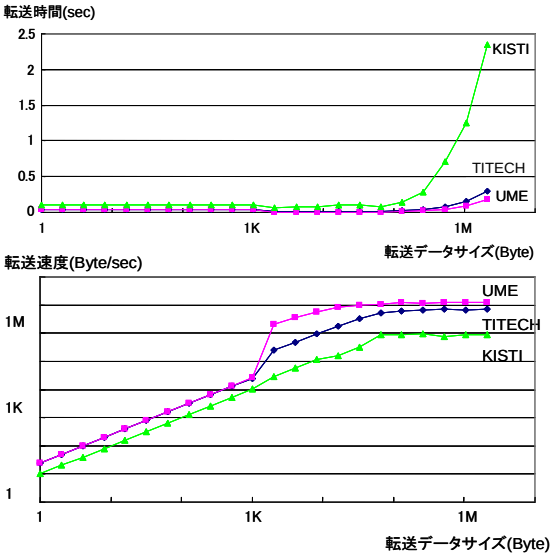

図2．Ninf-G2を用いたping-pongテストにおける通信時間（上図）と通信速度（下図）

を別々に送信している．従ってデータ部のサイズが小さい場合，データ部の送信に遅延が生じる(KISTIクラスタでは，1KBにおける通信時間が100msecを超えており，発生した遅延は目立たなくなっている)．このことから，globus_io_writev()関数が改善されれば，UME及びTITECHクラスタにおける送信データ長が小さい場合の性能改善が期待できる．

4.3 遠隔実行プログラム起動コスト測定結果

TITECHクラスタを対象に，関数ハンドル同時作成機能を利用した場合と個別に関数ハンドルを作成した場合の2つのモードにおいて利用プロセッサ数を変化させ，遠隔実行プログラムの起動時間を測定した．複数のプロセッサを利用した場合，起動時間はプロセッサによってばらつきがあるため，平均起動時間を算出した．結果を図3に示す．

関数ハンドル同時作成機能を利用した場合，プロセッサ数に対する起動コストの依存性はわずかである．それに対し，個別に関数ハンドルを作成した場合，起動コストは利用プロセッサの数に強く依存している．利用プロセッサ数が1の場合の起動コストと比較して，プロセッサ数が10の場合は3倍(12秒)，50の場合は30倍(97秒)程度の時間を要している．

原因は，個々に関数ハンドルを作成すると，利用プロセッサの数に比例してクラスタのフロントノードにGRAMのjob managerが起動され負荷が増大すること，およびjob managerが発行するジョブ実行要求がバッチジョブシステムにおいてシリアライズされることによる．実際，50プロセッサを利用した場合にフロントノードの負荷をuptimeコマンドで計測すると，最大40を超える値が観測された．

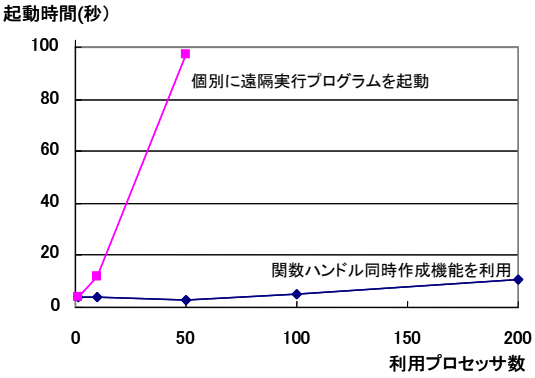

図3．TITECHクラスタにおける遠隔実行プログラム起動時間測定結果

また，100以上の関数ハンドルを個々に作成しようとすると，フロントノードの負荷が大きくなりすぎ作成に失敗するという現象がみられた．それに対し，関数ハンドル同時作成機能を利用すると200個の関数ハンドルでも安定して生成できた．

4.3 気象シミュレーション実行性能測定結果

気象シミュレーションシステムは，Ninf-G2を用いて順圧S-model[9]と呼ばれる逐次FORTRANプログラムをGridアプリケーションとした(Ninf化した)ものである．順圧S-modelは，短中期における大域的な気象変動を予測するプログラムであり，アンサンブル予報と呼ばれる手法を採用することにより予報精度の維持を図っている．アンサンブル予報は，擾乱を加えた多数のシミュレーション（サンプルシミュレーション）を行い，それらの結果に対して統計処理を行うことで，初期データとして用いられる観測データに含まれる誤差や計算途中に発生する誤差の成長を抑える手法である．

各サンプルシミュレーションは独立に実行できるため，順圧S-modelは典型的なタスク並列プログラムとなっている．我々は原プログラムからシミュレーション実行部分を抜き出し遠隔実行プログラムとすることで，順圧S-modelをNinf化した．

Ninf化に際して，クライアントコンポーネントをマルチスレッド化した．生成されたスレッドによりシミュレーションデータの作成，各バックエンドサーバに対する関数ハンドルの作成，遠隔関数呼び出しは全て並列に実行される．したがって，実行対象とするバックエンドサーバの一部で関数ハンドルの作成に時間を要しても，別のクラスタに対する関数ハンドルが先に生成されれば，それを用いてサンプルシミュレーションを開始できる．

このプログラムをAIST，TITECH，KISTIのクラスタに実装し，一ヶ月予報シミュレーションの実行性能を測定した．測定は，TITECHクラスタの利用を基本とし，1，50，150，200プロセッサを利用した場合の結果転送時間，個々のサンプルシミュレーションの実行時間，シミュレーション全体の経過時間を測定した．また，TITECHクラスタの200プロセッサに順次UMEクラスタ50プロセッサとKISTIクラスタ50プロセッサを追加し，最終的には合計300プロセッサを用いたシミュレーションを実施した．

表2．TITECHとAISTのクラスタを用いた気象シミュレーション実行結果

プロセッサ数 (AIST+TITECH)	結果転送時間 (秒)	シミュレーション時間 (秒)	総経過時間 (秒)
1	0.07	35.1	179
50	0.36	36.1	192
100	0.60	36.4	194
150	0.74	36.9	199
200	0.60	38.1	205
200+50	0.53	37.6	225
200+50+50	11.40	53.8	450
125+25+25 125+25+25	0.73	38.7	270

サンプルシミュレーションの実行時間は，クライアント側が遠隔実行プログラムにデータを送信する直前から実行結果を受け取った直後までの時間を計測している．スケーラビリティを調べるために，サンプルシミュレーションの総数は，プロセッサ数の5倍になるよう調整した．

表2に実験結果を示す．表の各欄は，サンプルシミュレーションにおける結果転送時間，シミュレーション時間の平均値，シミュレーション全体の経過時間を示す．

TITECHクラスタ上でプロセッサ数を変化させた場合，100プロセッサ程度までは結果転送時間が増大するものの，その後さらにプロセッサ数を増大させてもあまり変化がない．また，プロセッサ数の増加に伴い経過時間も徐々に増加するが，これは遠隔実行プログラム起動時間の増大が主原因である．200プロセッサを利用した場合でも経過時間の増大は30秒弱であり，良好な結果が得られている．

TITECHクラスタにさらにUME，KISTIクラスタを加えて実行した場合の結果転送時間，シミュレーション時間をみると，UMEクラスタを加えただけではほとんど変化がなかったが，KISTIクラスタも加えた場合，大きく増大した．

このコストの増大原因は二通り考えられる．すなわち，一つは単位時間当たりのデータ転送量がクライアント計算機とバックエンドサーバを結ぶネットワークの転送能力を超えてしまい，ネットワークがボトルネックになってしまっている可能性．もう一つはクライアントの処理量がクライアント計算機の能力を超えてしまい，クライアントがボトルネック

になってしまっている可能性である．この原因を調査するために，クライアント計算機の2ノードを用いて同時にクライアントコンポーネントを起動し，各々175プロセッサ((AIST, TITECH, KISTI)=(25,125,25)とする）を用いて750サンプルシミュレーションを実行させた．その結果（表2最下欄に表記），結果転送時間，シミュレーション時間とも大きく改善された．したがって，性能劣化の主原因はクライアントがボトルネックになっていたためであると考えられる..

このことは，MPI，OpenMP等を用いてクライアントコンポーネントをクラスタ上で並列実行させることで，さらに大規模なシミュレーションを効率よく実行できる可能性があることを示唆している．実際にクライアントを並列化し，性能評価を行うことは今後の課題である．

5. まとめと今後の課題

複数のクラスタから構成されるGrid環境での大規模シミュレーションに適したGridRPCシステムNinf-G2の実装機能及び性能評価について述べた．

Ninf-G2は，関数ハンドル同時生成機能やリモートオブジェクトのメカニズムを実装することで，遠隔手続き呼び出しに伴う起動コストや通信コストの低減を図るとともに，ハートビート機能，関数ハンドル作成タイムアウト機能，サーバ属性の個別設定機能を提供することで，非均質，不安定で動的に変化するGrid環境への対応を図っている．

現在，Ninf-G2はα版が公開されており，2004年度末には第一版がリリースされる予定である．

3サイトに配置されたクラスタを用いて，Ninf-G2の基本性能及び実行性能を測定した．その結果，関数ハンドル同時作成機能が起動コスト低減に有効であること，クライアントコンポーネントを多重化するといった工夫をすることで，数百プロセッサを用いて効率的にタスク並列処理を実行可能であることがわかった．

今後の課題は，先に述べたクライアントの並列化による効率的処理の可能性を実証することである．また，2003年度末に導入が予定されているAISTスーパクラスタやTeraGrid，ApGrid，PRAGMA等のGridテストベッド上のクラスタを加え，広域に分散した数千プロセッサ規模のGrid環境上で性能評価を行うことである．

謝　辞

本研究に際し，順圧S-modelプログラムを提供いただいた筑波大学田中博助教授に感謝いたします．

また，性能評価実験において計算資源を提供いただいたTITECH，KISTIに感謝いたします．

なお，本研究の一部は文部科学省「経済活性化のための重点技術開発プロジェクト」の一環として実施している超高速コンピュータ網形成プロジェクト(NAREGI: National Research Grid Initiative)によるものである．

参 考 文 献

1)Lee, C., and Talia, D.: “Grid Programming Models: Current Tools, Issues and Directions”, Grid Computing: Making the Global Infrastructure a Reality, pp.555-578 (2003).

2)Lee, C., Matsuoka, S., Talia, D., Sussman, A., Mueller, M., Allen, G. and Saltz, J.:”A Grid Programming Primer”, GWD-I, GGF Advanced Programming Models Research Group, (2001).

3)Catlett, C.: “Standards for Grid Computing: Global Grid Forum”, Journal of Grid Computing Vol.1, No.1, pp.3-7 (2003).

4)Seymour, K., Nakada, H., Katsuoka, S., Dongarra, J., Lee, C. and Casanova, H.: “Overview of GridRPC: A Remote procedure Call API for Grid Computing”, Proceedings of Grid Computing – Grid 2002, pp.274-278 (2002).

5) http://ninf.apgrid.org/.

6) http://icl.cs.utk.edu/netsolve/.

7)Tanaka, Y., Nakada, H., Sekiguchi, S., Suzumura, T. and Matsuoka, S.: “Ninf-G: A Reference Implementation of RPC-based Programmin Middleware for Grid Computing”, Journal of Grid Computing, Vol.1, No.1, pp.41-51 (2003)

8)Casanova, H., and Dongarra, J.: “NetSolve: A Network Server for Solving Computational Science Problems”, Proceedings of Supercomputing’96 (1996)

9)Caron, E., Deprez, F., Lombard, F., Nicod, J-M., Quinson, M. and Suter, F.: “A Scalable Approach to Network Enabled Servers”, Proceedings of the 8th International EuroPar Conference, LNCS Vol.2400, pp.907-910 (2002)

10)佐藤三久，朴泰祐，高橋大介：“OmniRPC: グリッド環境での並列プログラミングのためのGridRPCシステム”, 情報処理学会論文誌コンピューティングシステム, Vol.44, pp.34-45 (2003).

11) Takemiya, H., Shudo, K., Tanaka, Y. and Sekiguchi, S.: “Constructing Grid Applications using Standard Grid Middleware”, Journal of Grid Computing, submitted.

12)中島佳宏，佐藤三久，後藤仁志，朴泰祐，高橋大介：“CONFLEX-G: OmniRPCによるグリッド環境上での分子立体配座探索”,HPCS2004論文集，pp.95-102(2003).

13)Tanaka, H.-L. and Nohara, D.: “A Study of Deterministic Predictability for the Barotropic component of the Atmosphere”, Science Reports of the Institute of Geoscience, University of Tsukuba, section A, Vol.22 pp.1-21 (2001).

14)合田憲人，中村心至：“細粒度最適化問題アプリケーションのグリッドテストベッド上への実装”, HPCS2004論文集，pp.75-76 (2003).

15)武宮博，首藤一幸，田中良夫，関口智嗣：“Grid環境上における気象予報シミュレーションシステムの構築”, 情報処理学会論文誌コンピューティングシステム，Vol.44, pp.23-33 (2003).

16)田中良夫，中田秀基，朝生正人，関口智嗣：“Ninf-G2: 大規模環境での利用に即した高機能，高性能GridRPCシステム”, 情報処理学会研究報告, Vol.2003, No.83, pp.65-70 (2003).