大規模長時間実行アダプティブ Grid アプリケーションの実現

Implementation and Execution of Large-scale Long-run Adaptive Grid Applications

武宮 博 Takemiya Hiroshi

(1)数千プロセッサ規模の計算機資源を必要とする，(2) 数週間から数ヶ月に及ぶ長期実行を必要とする，(3)計算の進展に応じて動的に計算量が変化する，という特徴をもつアプリケーションを，GridRPC と MPI を組み合わせるプログラミング手法に基づき Grid 環境上で実行可能とした。本手法を用いることにより，上記 3 点の特徴を持つ Grid アプリケーションを構築するために必要な，プログラムの柔軟性，頑健性，効率性を実現することが可能となる。本手法の有効性を検証するために，大規模化学シミュレーションプログラムを本手法に基づき Grid 環境上で実行可能とし，環太平洋 Grid 環境上で長時間継続実行させる実験を試みた。その結果，障害発生や計算量の変化に対応して実行対象計算機やその利用規模を自動的に変更しながら 150,000CPU 時間に及ぶシミュレーションを実現することができた。

１．はじめに

広域に分散配置された計算機資源を連携させ，大規模複雑な問題を解く Grid Computing は，科学技術分野における種々の問題を解決するための新しいキーテクノロジとして注目されてきている。Global Grid Forum(GGF)等の活動を通して，Grid Computing を支える Grid 基盤や Grid ミドルウェアの持つべき機能に関する広範な議論が行われ，インタフェースの標準化，実装が推し進められてきた。このような基盤部分に関する研究開発の進展に伴い，実運用レベルの Grid 環境を構築する試み [1),2)] が世界各国で展開されているほか，バイオサイエンス，ナノサイエンス，原子力物理などの応用分野でも，Grid アプリケーションの実装が行われてきている。しかしながら，これらの分野において Grid アプリケーションの構築事例はまだ少数にとどまっており，その種類も，Grid 上で多数の独立なタスクを処理するもの [3)]か，Grid-enabled MPI[4)]を用いて既存の並列プログラムを Grid 上で動作させるもの [5)]に限られている。

未だに大規模 Grid アプリケーションの開発，実行が進展していない大きな原因として，実際に Grid アプリケーションを開発，実行する際の方法論が確立していないという点が挙げられる。Grid アプリケーションの開発，実行を加速し，Grid を実用レベルの技術とするためには，多様なアプリケーションの Grid 上での実行可能性を検討し，開発および実行の方法論を明確化する必要がある。産業技術総合研究所グリッド研究センター殿（以下，産総研グリッドセンター）では，2002 年より，Grid 基盤レベルの研究開発から Grid アプリケーションの研究開発に至る包括的な Grid 研究プロジェクトを推進してきた。報告者は本プロジェクトに参画し，Grid ミドルウェアの一つである Ninf-G[6)]と呼ばれる GridRPC システムの研究開発に参画するとともに，Ninf-G を用いて種々の実アプリケーションを Grid 環境上で実行可能とし（以下，既存アプリケーションを Grid 環境上で実行可能とすることを Grid 化と呼ぶ），広範に分散した並列計算機から構成される Grid テストベッド上での実行実験 [7),8),9),10)]を行ってきた。

本稿では，Grid 上での実行に適した新たな種類のアプリケーションとして，(1) 数千プロセッサ規模の計算機資源を要する，(2)数週間から数ヶ月に及ぶ長期実行を必要とする，(3)計算の進展に応じて動的に計算量が変化する，といった特徴を持つアプリケーション（以下，大規模長時間実行アダプティブアプリケーション，Grid 化されたものを大規模長時間実行アダプティブ Grid アプリケー

ションと呼ぶ）の Grid 化について報告する。このような特徴を持つアプリケーションとして，量子計算-分子動力学計算の連成化学シミュレーション[11]や，流体－構造連成シミュレーション[12]，あるいは分割統治法に基づく衝撃波解析シミュレーション[13]等が挙げられる。ここでいう“アダプティブ”とは，計算対象としている系の状況にあわせて必要な領域のみ詳細な計算を行う手法を意味し，計算量を抑えつつ高精度な結果を得るために広く用いられている手法である。また“連成シミュレーション”とは，複数の計算原理に基づくシミュレーションを連携させて複雑な系を計算する手法である。これらの手法は，計算アルゴリズムの進展，計算機能力の向上に伴い新しく提唱されてきたもので，従来不可能であった高い精度でのシミュレーションを可能にすることから，産業界を含め高い要求があるが，膨大な計算機資源を必要とすることからその利用が制限されていた。これらを Grid 化するための方法論を明確にすれば，Grid アプリケーションの開発，実行の加速が期待される。

大規模長時間実行アダプティブ Grid アプリケーションは，地理的に分散した並列計算機上で複数の細粒度並列プログラムが互いに疎に連携しつつ，状況に応じて動的に計算規模を変更しながら，総計数千プロセッサを用いて長時間に及ぶ計算を行う。このような実行を可能とするためには，1)アプリケーションの要求あるいは Grid を構成する計算機資源の状況に応じて実行対象計算機を変更していく柔軟性，2)発生する障害を検知，復旧して処理を継続していく頑健性，3)多数の計算機資源を用いてスケーラブルな処理を実現する効率性, が求められる。Grid-enabled MPI 等，従来採用されていた Grid 化手法では，これらの要件全てに対応することは困難である。これらの要件を満足する新しい Grid 化手法として，GridRPC と MPI を組み合わせる手法を提案する。本手法は, MPI によって並列化されたシミュレーションプログラムを GridRPC により動的に起動し，クライアントプログラムを介して連携させるものである。GridRPC の持つ遠隔プログラム動的実行機能および障害検知機能と，MPI の持つ効率的な並列実行機能を相補的に組み合わせることにより，上記の要件を満たす Grid アプリケーションの実装を可能とする。

本手法の有効性を検証するために，典型的な大規模長時間実行アダプティブアプリケーションの一つである Adaptive Hybrid QM/MD シミュレーションコードを Grid 化し，環太平洋 Grid テストベッド上で長期にわたり継続実行する実験を行った。実験結果に基づき本手法の有効性を評価するとともに, 得られた知見をまとめる。

２．大規模長時間実行アダプティブ Grid アプリケーション

2.1 大規模長時間実行アダプティブ Grid アプリケーションの実行

1.で述べた特徴を持つ大規模長時間実行アダプティブ Grid アプリケーションを実運用 Grid 環境で実行することを考えた場合，シミュレーション開始から終了まで同一計算機を継続して利用することは現実的でない。実運用 Grid 環境では，同一利用者による計算機の継続利用に関して CPU limit 等の形で制限が課せられていることが一般的である。また，定期的な保守作業等の理由で計算機システムが停止することもある。したがって，数ヶ月に及ぶシミュレーションを実行するためには，動的に利用対象計算機を変更しつつシミュレーションを継続する柔軟性が要求される。

また，大規模長時間実行アダプティブ Grid アプリケーションは，計算の進展に応じて計算量が変化するという特徴も有している。したがって，計算機システムの制約だけでなく，計算量の変化というシミュレーション自体の要求にも対応して柔軟に実行対象計算機を変更できなければならない。

次に，広域に分散した計算機を連携して長時間シミュレーションを実行する場合には，単一の計算機を利用する場合と比較して，ネットワークや計算機上で障害が発生する可能性が高い。大規模な計算機資源を利用すればさらにその可能性は高まる。したがって，障害の発生を前提とし，障害に対処しつつシミュレーションを継続できる頑健性が必要となる。

また，大規模長時間実行アダプティブ Grid アプリケーションは，数千プロセッサ規模の計算機資源を利用することから，それら資源を効率的に利用したシミュレーションの実行が必須である。したがって，実行の効率性が要求される。

2.2 要件

2.1 で述べた柔軟性，頑健性，効率性の 3 つの特質は，さらに細かな機能要件として細分される。以下に，各機能要件をまとめる。

(1) 柔軟性

・ 動的割付:必要に応じて動的に計算機上で起動され，

実行されなければならない。

- マイグレーション：利用する計算機を動的に変更しつつ実行を継続できなければならない。
- 計算規模の変更：問題規模の変化に応じて利用計算機の規模を変更できなければならない。
- 利用計算機の追加，変更：当初想定していなかった計算機を利用対象として新たに追加したり，運用スケジュールに応じて利用可能 CPU 数を変更する等，計算機の利用に関する情報を動的に変更できなければならない。

(2) 頑健性

- 障害の検知：シミュレーション実行中に発生する種々の障害を検知できなければならない。計算機システムのダウン，ネットワークの切断といったあらわな障害（以後，陽的障害）だけでなく，計算機が他のユーザに利用されているために実行を長時間待たなければならないといったことも広い意味での障害（以後，陰的障害）と捉え，対応できなければならない。
- 障害の復旧：発生した障害から自動的に復旧できなければならない。

(3) 効率性

- 計算機資源の効率的管理：地理的に分散した数千台規模の計算機資源を効率的に管理できなければならない。
- 通信，計算の効率的実行：個々の細粒度並列シミュレーションおよびそれらの連携を効率的に実現できなければならない。

3．Grid 化手法

3.1 既存 Grid 化手法

Grid アプリケーションを開発する代表的な手法である Grid-enabled MPI や GridRPC は，上記の要件を全て満足することが困難である。

Grid-enabled MPI には，既存の MPI プログラムを修正なしでそのまま Grid 環境上で実行できるという利点がある。また，大規模シミュレーションを効率的に実行するための通信機構を提供している。しかし，Grid-enabled MPI には，以下の問題点が存在する。

(1) 静的なシステム構成

MPI では動的にプロセスを生成し，プロセスの構成を変更することができない。MPI-2 では動的なプロセス生成関数が規定されているが，対象計算機上にプロセスが生成されるまで呼び出し元がブロックするため，対象計算機上でのプロセス生成に長時間を要すると，処理全体がブロックしてしまう。

(2) 低い頑健性

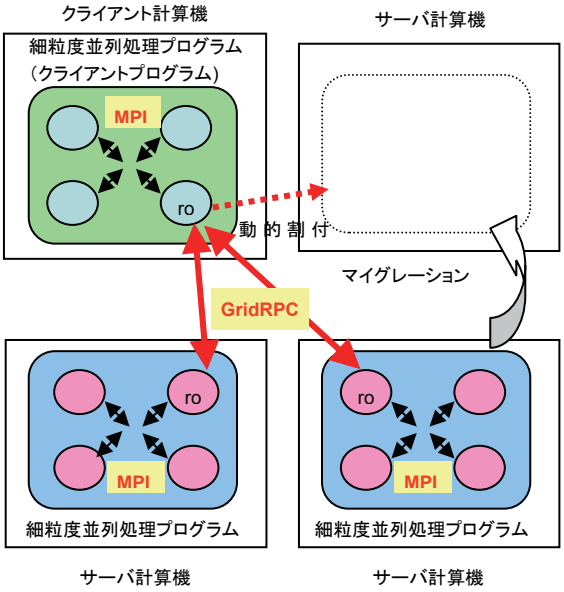

図 1 GridRPC+MPI を用いたプログラミング手法により実現されるシステム構成例

Grid 環境を構成するネットワークや計算機システムにおいて障害が発生すると，MPI ではジョブがキャンセルされてしまう。したがって，Grid-enabled MPI を用いた場合，障害が発生するたびにプログラム全体を起動しなおす必要がある。MPI において耐故障性を高める機構に関しては，Fault Tolerant MPI[14] 等が提案されているが，まだ研究段階であり，実用段階には至っていない。

一方，GridRPC は既存の逐次プログラムにおける関数呼び出しを遠隔関数呼び出しに変更することで，容易に Grid アプリケーションを構築可能であるという特徴を持つ。動的な遠隔プログラムの起動が可能であるため，柔軟性の実現も可能である。さらに，ネットワークや計算機資源の障害検知も考慮した設計となっており，耐故障性を持つアプリケーションを開発可能である。しかし，GridRPC を用いたアプリケーションの実装では，大規模な計算資源を効率的に管理することが困難である。単一のクライアントが全てのサーバプログラムを管理しなければならず，シミュレーションが大規模になった場合，その管理コストが増大してしまうからである。

3.2 提案手法

本稿で提案する手法は，GridRPC と MPI を組み合わせるもので，MPI によって並列化された遠隔プログラムを GridRPC により動的に起動し，連携する。本手法を大規模長時間実行アダプティブアプリケーションに適用した場合のシステム構成例を図 1 に示す。アプリケーションを構成する複数の細粒度並列シミュレーションプログラムは並列計算機上に各々動的に割り付けられ，クライアントプログラムを中心として疎に連携する。このような構成は，アプリケーションの特質に適合しているだけでなく，分散配置された並列計算機から構成される Grid のトポロジーにもうまく適合している。

本手法は上記の要件を以下のように支援する。まず，GridRPC の持つ動的な遠隔実行プログラム起動機能と多様な障害検知機能を利用してプログラムの柔軟性および頑健性の実現を支援する。また，MPI の持つ遠隔実行プログラム内での効率的な並列処理機能と GridRPC の遠隔実行プログラム管理機能を組み合わせることにより効率性の実現を支援する。GridRPC のみを用いた場合には、多数の計算資源を単一のクライアントが管理しなければならないため，効率的な管理が困難であるという問題点が存在した。しかし，本手法では GridRPC のクライアントは MPI によって生成されるプロセスグループの単位で計算機資源を管理するため，管理対象の個数を少なく抑えることが可能であり，その結果，管理コストの低減が期待できる。

4．大規模長時間実行アダプティブアプリケーションの Grid 化

4.1 Adaptive Hybrid QM/MD シミュレーションコード

Adaptive Hybrid QM/MD シミュレーションコードは，材料工学分野における次世代電子デバイスやマイクロマシンの設計において，亀裂発生等ナノスケール特性を解析するために開発されたプログラム[16]である。

本コードは，Density Functional Theory (DFT) に基づく詳細なQM (Quantum Mechanics) シミュレーションと経験的原子間ポテンシャルを用いた古典MD (Molecular Dynamics) シミュレーションを組み合わせる。古典MDシミュレーションを用いて系全体の振る舞いを計算するとともに，高精度ではあるが長時間の計算を要するQM シミュレーションを用いて，亀裂発生の可能性のある領域等，詳細な解析が必要な領域(以後，QM 領域)の原子の振舞いを計算しMDシミュレーションの結果を補正することにより，現実的な時間内に大規模系の挙動を精度よく解析可能としている。

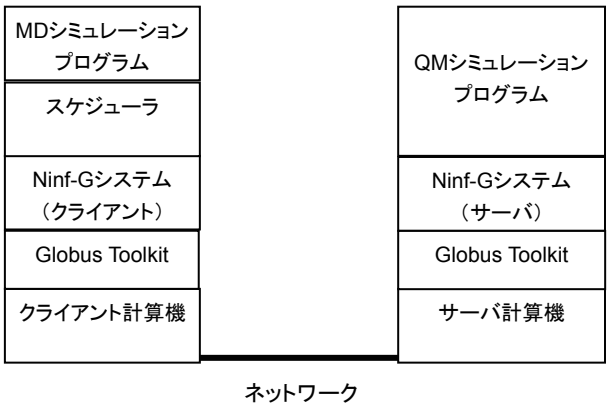

図2 Grid化大規模Hybrid QM/MDシミュレーションプログラムの構成

MD，QM 両シミュレーションは各々細粒度の並列処理により計算が行われる。特にQM シミュレーションの計算量は膨大で，典型的な系に対し数百～数千台規模のプロセッサを用いて1タイムステップの計算に10^3秒程度，シミュレーションの完了には数週間から数ヶ月を要する。QM シミュレーションとMDシミュレーション間ではQM領域内に存在する原子の位置，速度，力といったデータの交換が必要であるが，QM領域の粒子数をNとすれば，計算量がNの3乗に比例するのに対し，通信量はNの１乗にしか比例しないため，両者の連携は疎である。また，本シミュレーションでは，系の挙動に従い計算実行中にQM領域の再定義が行われ，QM領域の数や領域に含まれる原子の数が変化する。その結果，動的に計算量が変化する。したがって，Adaptive Hybrid QM/MD シミュレーションコードは，大規模長時間実行アダプティブシミュレーションの特徴を満足している。

4.2 Adaptive Hybrid QM/MD シミュレーションコードの Grid 化

3.で述べたように，GridRPCとMPIを組み合わせることにより，柔軟性，頑健性，効率性を実現するための多様な機能が提供されるが，これらの機能だけで全ての機能要件が満足されるわけではない。提供される機能を基にして，上位のレイヤで実現すべき機能も存在する。

Adaptive Hybrid QM/MDシミュレーションをGrid化するにあたり，上位レイヤで実現すべき機能を検討し，アプリケーションによらず共通して必要な機能をまとめたスケジューリングレイヤおよびアプリケーション固有の

知識に依存する機能をまとめたアプリケーションレイヤに分割して実装した（図２参照のこと）。以下に各レイヤにおいて実装された機能の概要を示す。詳細に関しては文献9)，10)を参照いただきたい。

スケジューリングレイヤにおいては,(1)計算機の状態を管理し，利用対象計算機を選択する計算機選択機能，および(2)障害発生等の理由から中断したシミュレーションを復旧,継続する状態回復機能の2つの機能を実装した。計算機の選択手法に関しては，種々のものが考えられるが，今回は選択した計算機をなるべく継続利用する方法を採用した。この方法では，障害が発生した時も可能な限りその計算機の再利用を試みる。これは，今回対象とするシミュレーションが,数百から数千個のCPUを持つ大規模クラスタ上での実行を必要とするという特性に基づいている。大規模な計算機資源を必要とするシミュレーションの場合，障害発生時に代替計算機を動的に確保することは困難であり，仮に存在しても大規模シミュレーションのマイグレーションコストが大きいからである。また，状態回復機能の実装に際しては，以下の点を考慮した。一般に系の時間変化を計算するシミュレーションの処理内容は，初期パラメータを読み込む初期化処理および実際のシミュレーションを行うシミュレーション処理から構成され，シミュレーション処理は時間進展ループ内で繰り返し呼ばれる構造となっている。この特性に着目し，アプリケーションから渡される初期パラメータデータおよびシミュレーションの入力データをスケジューリングレイヤ内で保持しておき，状態回復に利用するようにした。

アプリケーションレイヤの実装に際しては,MDとQMシミュレーション間で転送されるデータ量の削減を図った。障害発生等の理由により実行対象計算機が変更される場合に備えて，QMシミュレーションの継続実行に必要なデータを毎回クライアント計算機上に転送しておく必要がある。しかし，QMシミュレーション継続に必要なデータ量は膨大で，典型的な系のシミュレーションにおいて1GB程度のデータを毎回転送しなければならない。このような大量のデータ転送は，シミュレーション実行効率を低下させる。Hybrid QM/MDコードにおける転送データの大部分は，Self Consistent Field計算(SCF計算)の初期値推定に用いられるため，実行計算機が変更された場合には乱数を基にSCF計算の初期値を推定することとし，上記データを不要とすることで転送量を低減することにした。その結果，転送量は数MB程度に抑えられるようになった。ただし，乱数に基づきSCF計算の初期値を推定すると経験上SCF計算時間が増大するため，初期値推定用のデータを実行計算機上に保存しておき，実行計算機が変更されない限りそのデータを利用して推定を行うようにした。

５．長時間実行実験

5.1 実験概要

日本国内および米国Tera Grid，南カルフォルニア大学に設置された計8台のクラスタを利用し，19日間にわたってAdaptive Hybrid QM/MDシミュレーションの長時間実行実験を行った。利用したクラスタの仕様を表1に示す。実験に先駆けて，利用計算機のスケジュールを行った。スケジュールの際には，動的なQM領域数の増加や障害発生に伴う利用計算機数の変化に備えて，5台のクラスタに加え1台の予備クラスタをスケジュールし，領域数の増加に備えることにした。 スケジュールは大きく5つのフェーズに分類される。

フェーズ0 (9日間）：AIST Super Cluster(P32,M64 及びF32 クラスタ, 以下ASC) のうち, P32 のみを用いる。（予備的計算フェーズ），

フェーズ1（2日間）： ASC の3 台のクラスタを用いる，

フェーズ2（3日間）： ASC 及びNCSA とPSCに各々設置されたTeraGrid クラスタ2 台を用いる，

フェーズ3（3日間）： ASC 及びUSC のクラスタを用いる，

フェーズ4（2日間）：ASC，U-Tokyo 及びTITECH のクラスタを用いる。

表１．実験に利用したクラスター覧。

サイト名は, AIST: 産業技術総合研究所, NCSA: National Center for Supercomputing Applications, PSC: Pittsburgh Supercomputing Center, USC: University of Southern California, U-Tokyo: 東京大学, TITECH: 東京工業大学を表す。

計算機名	サイト名	利用CPU数
P32	AIST	768
M64	AIST	256
F32	AIST	256
NCSA	NCSA	128
TCS	PSC	256
USC	USC	512
ISTBS	U-Tokyo	128
Presto	TITECH	256

実験では, SIMOX(Separation by Implanted of Oxygen)と呼ばれるIC チップ製造技術に関するシミュレーションをおこなった。SIMOX は，高速に加速された酸素原子をシリコン基盤に入射しSIO_2 絶縁層を形成する技術である。シミュレーションでは，シリコン基板上に入射位置の異なる5つの酸素原子を配置し入射させ，酸素原子と基盤内のシリコン原子の挙動を調べる。本シミュレーションでは，個々の酸素原子及びその周辺のシリコン原子がQM 領域として動的に定義され，酸素原子がシリコン基盤の内部に侵入するにつれてQM 領域に含まれる原子の数は徐々に増加する。また，シミュレーション開始直後のQM 領域数は5つであるが，酸素原子とシリコン原子の相互作用の結果によっては，1つのQM領域が2つに分割されたり，複数のQM 領域が1つに統合されたりする。

5.2 実験結果

実験では，19日間で150,000CPU時間を用い，タイムステップ数にして計270ステップの計算が行われた。また，10日間の本計算において最長5日間にわたる継続実行を実現することができた。実験結果に関して柔軟性，頑健性，効率性の観点からまとめる。

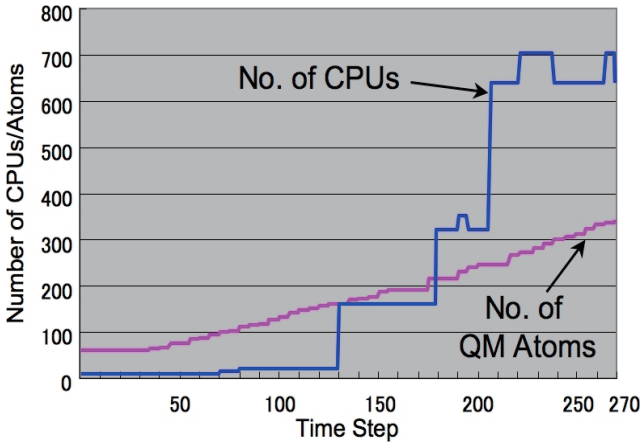

図3 QM原子数と利用CPU総数の変化

（1）柔軟性

図３にQM 原子数の変化とそれに伴う利用CPU総数の変化を示す。実験開始時にQM 領域数は5，QM原子の総数は62個，利用CPU の総数は10であった。シミュレーションが進むにつれて徐々にQM 原子の総数は増加し，最終的には341 個になっている。それに伴い利用CPUの総数も増加し，実験フェーズ0の途中から100CPUを越え，最終的に最大702 まで増加している。

図中に見られる一時的な利用CPU 数の増加は，QM領域数が一時的に5 から6 に増加したことを表している。このような利用CPU 数の増加に伴い，実験期間中に計244回の動的割り付けが発生している。これらの結果は，GridRPCとMPIを組み合わせるプログラミング手法に基づき実装したアプリケーションが動的な計算量の増加に対応し，利用CPU 数を自動的に調節できることを示している。

（2）頑健性

実験期間中，770 回の障害発生が観測された。これらのうち，遠隔実行プログラムの起動に失敗したケースが712 回と大半を占める。失敗の主な原因は，Globus Tool kitのjob manager が起動されていなかった，MPI プロセス間の通信用メモリ資源の確保ができなかった，バッチキューがactive になっていなかった，ディスク容量制限を越えてしまい起動に必要なファイルの生成ができなかった等である。起動に成功すると，比較的安定してシミュレーションが実行された。

障害が発生した場合，同一計算機への再割り付けを試みても成功する確率は低い。障害発生時に同一計算機への割り付けを試みて成功する確率は4.6%((成功34回，失敗706回）程度であり，同一計算機に対して規定回数の失敗を繰り返した後，最終的には予備クラスタに割り付けられた。今回の実験では，規定回数失敗するまで同一計算機に対して再起動を試みる手法を採用したが，規定回数を多く設定しても有効ではないことがわかった。

実験中発生した障害において，Ninf-G の提供するタイムアウト機構により110 回の障害が検知された。また，予約期間超過に関するタイムアウトに関しても，期間の超過を検出しその時点で予約されている別の計算機へと実行対象が変更されることが確認された。

（3）効率性

今回の実験では最大702CPU を用いたが，計算時間の大半はQM シミュレーションの実行で占められ，MD,QMシミュレーション間の通信時間やQMシミュレーションの起動コストは数%にすぎなかった。このように少ないコストで多数の計算機資源を管理できた要因は2つ存在する。一つは，QM シミュレーションにおいて乱数に基づくSCF 計算の初期値推定手法を採用することにより，MD，QM 両シミュレーション間のデータ転送量を低減したというアプリケーションレベルの工夫である。もう一つは，多数の計算資源をGridRPCのクライアントが直接管理するのではなく，それらをMPIプロセスグループ

単位で管理することにより，GridRPCクライアントの管理対象数を削減するという本稿で提案したプログラミング手法の特性によるものである。今回の実験では，利用可能な計算機資源の制約により数百プロセッサの利用に留まったが，このことは，数千プロセッサを要する場合でも本提案手法に基づきプログラミングすることにより効率的な管理が可能であることを示唆している。

６．おわりに

動的に計算規模が変化するという特性を持つ大規模長時間実行アプリケーションをGrid 化し，実行するための方法論について論じた。実装に際しては，柔軟性，頑健性，効率性という3種類の要求を満足する必要があることを示し，それらを満足するために実装すべき機能に関して考察した。また，典型的な大規模長時間実行アダプティブシミュレーションの例であるAdaptive Hybrid QM/MD シミュレーションコードをGrid 化し，環太平洋Grid 環境上で長時間継続計算実験を行うことで，我々のアプローチの有効性を検証した。

その結果，動的な計算規模の変化に対応し，実行対象計算機を変更しながら利用CPU 数を自動的に調節できることを確認した。実験中には多数の障害が発生したが，それらを自動的に検知，復旧してシミュレーションを継続できた。さらに，数百プロセッサ規模の計算機資源を効率的に管理できることも確認した。

しかしながら，実用レベルで大規模長時間実行Gridシミュレーションを実施するには解決すべき問題が数多く残っている。例えば，単に再起動を行うという単純な戦略では一旦発生した障害の復旧は困難であり，現段階ではユーザの介入による問題点の解決が必要である。より洗練された障害復旧の機構を考案することも重要であるが，より現実的なGrid 環境の運用管理形態を検討していくことがさらに重要である。

今回の実験において発生した障害の原因は，運用管理技術及び組織間の情報伝達技術の未熟さに起因するものが多い。またGrid 環境に特有な問題として，計算機が広域に分散しているために，あるサイトの計算機に障害が発生した場合，原因がわかっているにもかかわらず時差の関係でサイト管理者と連絡がとれず，解決が遅れるということも発生した。このような場合，他の計算機に計算を割り付けることで実行を継続するということが有効であることはわかったが，根本的な解決法ではない。広域なGrid 環境をどのように運用管理すればよいかを検討し，明確化することが急務である。そのために，今回のような実験を積み重ね，発生する障害に関する情報を蓄積していくことが重要である。

なお，本研究に際し，名古屋工業大学尾形修司助教授には，Hybrid QM/MD コードを提供いただいた．また，南カリフォルニア大学中野愛一郎助教授，および産総研グリッドセンター中田秀基研究員，田中良夫チームリーダ，関口智嗣センター長には，本研究の遂行にあたり貴重な御助言をいただいた．

また，SC2005 における実験に協力し計算資源を提供いただいたTeraGrid Exective Commitee メンバ諸氏，Pittsburgh Supercomputing Center, National Center for Supercomputing Applications, University of Southern California，東京大学田浦研究室，東京工業大学松岡研究室，産総研グリッドセンターに深謝する次第である。

参考文献

1) http://www.teragrid.org

2) http://public.eu-egee.org

3) H. Casanova, T. Bartol, J. Stiles, and F. Berman: “Distributing Mcell Simulations on the Grid, Journal of Supercomputing Applications”, Vol. 15, pp.243-257 (2001)

4) N. Karonis, B. Toonen, and I. Foster: “MPICH-G2: A Grid-Enabled Implementation of the Message Passing Interface”, Journal of Parallel and Distributed Computing, Vol.63, No.5, pp.551-563 (2003)

5) H. Kikuchi, R.K.Kalia, A.Nakano, P. Vashishta, H.Iyetomi, S.Ogata, T. Kouno, F. Shimojo, K.Tsuruta, and S. Saini: “Collaborative simulation Grid: Multiscale Quantum Mechanical/Classical Atomistic Simulations on Distributed PC Clusters in the US and Japan”, in Proceedings of SC2002 (2002)

6) 武宮博, 田中良夫, 中田秀基, 関口智嗣 : ”Ninf-G2: 大規模Grid環境での利用に即した高機能, 高性能GridRPCシステムの実装と評価”, 情報処理学会論文誌 : コンピューティングシステム，Vol. 45, No. SIG11 (ACS7), pp.144-150 (2004)

7) 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 No. SIG11(ACS3), pp.23-33 (2003)
8) H. Takemiya, K. Shudo, Y. Tanaka, and S. Sekiguchi: “Constructing Grid Applications Using Standard Grid Middleware”, Journal of Grid Computing, Vol. 1, pp. 117-131 (2003)
9) 武宮博，田中良夫，中田秀基，関口智嗣：”MPIとGridRPCを利用した大規模Gridアプリケーションの開発と実行：Hybrid QM/MDシミュレーション，情報処理学会論文誌：コンピューティングシステム，Vol. 46, No. SIG12 (ACS11)，pp. 384-395 (2005)
10) H. Takemiya, Y. Tanaka, S. Sekiguchi, S.Ogata, R. K. Kalia, A. Nakano, P. Vashishta: "Sustainable Adaptive Grid Supercomputing: Multiscale Simulation of Semiconductor Processing across the Pacific", in Proceedings of SC'06 (accepted) (2006)
11) P. Vashishta, R. K. Kalia, and A. Nakano: “Multimillion Atom Simulations of Dynamics of Oxidation of an Aluminum Nanoparticle and Nanoindentation on Ceramics”, Journal of Physical Chemistry B, Vol. 110, pp.3727-3733 (2006)
12) T. Kimura, and H. Takemiya: “Distributed Parallel Computing for Fluid-Structure Coupled Simulations on a Heterogeneous Parallel Computer Cluster”, International Journal of High Performance Computing Applications, Vol. 13, No. 4, pp.320-333 (1999)
13) P. Branicio, R. K. Kalia, A. Nakano, P. Vashishta: “Shock-induced Structural Transition, Plasticity, and Brittle Cracks in Aluminum Nitride Ceramic: A Molecular Dynamics Study, Physical Review Letters, Vol. 96, pp. 1-4 (2006)
14) E. Gabriel, G. Fagg, A. Bukovsky, T. Angskun, and J. Dongarra: “A Fault-Tolerant Communication Library for Grid Environments”, in Proceedings of 17th Annual ACM International Conference on Supercomputing (ICS’03), International Workshop on Grid Computing and e-Science (2003)
15) S. Ogata, F. Shimojo, R. Kalia, A. Nakano, P. Vashishta: “Hybrid Quantum Mechanical/Molecular Dynamics Simulations on Parallel Computers: Density Functional Theory on Real-space Multigrids”, computer Physics Communications, Vol. 149, pp.30-38 (2002)

武宮　博　　1989 年入社
公共ソリューション本部
Grid 技術の研究開発，Grid 技術の大規模シミュレーションプログラムへの適用
takemiya@hitachi-to.co.jp