1. 地球シミュレータ・システム

地球シミュレータのリクエスト実行システムと運用状況

海洋科学技術センター 地球シミュレータセンター
板倉 憲一
itakura@es.jamstec.go.jp

海洋科学技術センター 地球シミュレータセンター
宇野 篤也
uno@es.jamstec.go.jp

地球シミュレータの利用形態

■ プログラム開発環境

地球シミュレータ（ES）と周辺装置の接続を**図-1**に示す．ESのユーザは，地球シミュレータセンターの端末室からログインサーバにまずアクセスする．ログインサーバは12CPUのサーバ系ワークステーション（WS）である．本特集第1.2章「地球シミュレータのプログラミング環境」で述べたプログラミングを行うクロス環境は，このログインサーバ上に構築されており，ユーザはここでプログラム開発を行う．ログインサーバでは，ユーザの小規模から中規模のデータを保持するS系ディスクをNFSによって共有しているため，ユーザデータの管理も行える．さらに，大規模データストレージ用のMTをコントロールするユーティリティなども導入されている．

また，ユーザプログラムの実行リクエストのキューイングシステムもこのログインサーバから利用する．キューイングシステムの管理は別の管理用WSで動作している．本稿では，このキューイングシステムを用いた地球シミュレータでのリクエスト実行システムと運用状況について解説する．

■ アプリケーションプログラムの実行

ESは，全システムを用いるとLinpack性能約35TFLOPSの実効性能を有するが，通常運用時にはノード数および利用時間を分割し，多くのユーザによって共同利用されている．ユーザがアプリケーションプログラムを実行するには，アプリケーションの実行用シェルスクリプトを作成し，そのスクリプトにプログラムを実行させるために必要な情報をコメントのかたちで加える．このスクリプトをリクエストとしてバッチ処理キューイングシステムに投入する．キューイングシステムはユーザが投入したリクエストをシステムの実行状況に合わせてスケジューリングし，アプリケーションプログラムを実行していく．

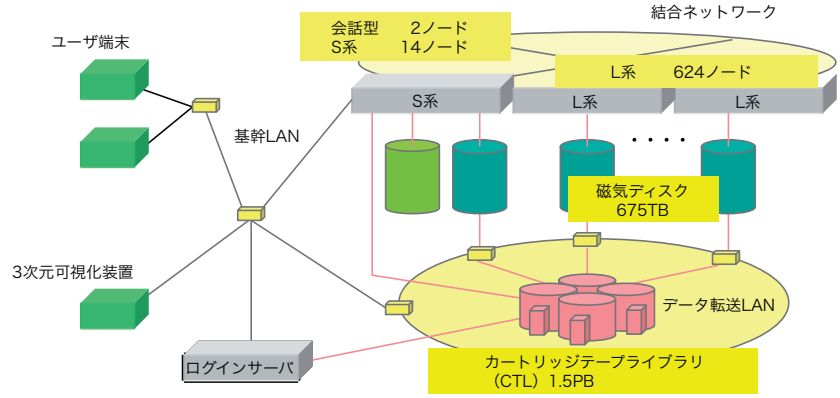

図 -1 地球シミュレータと周辺装置の接続

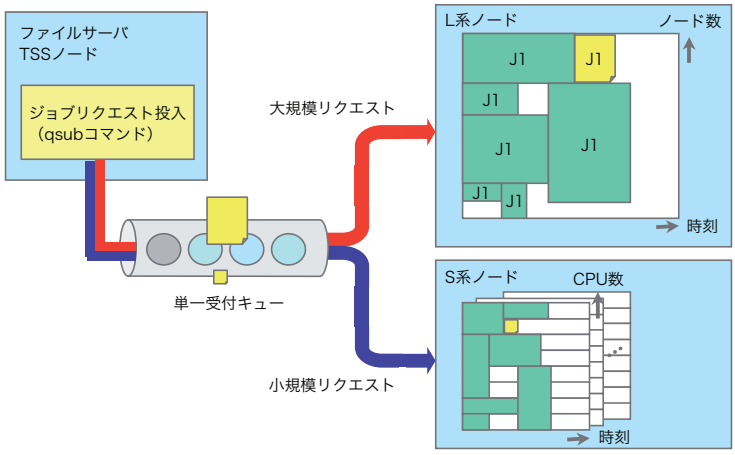

図 -2 ジョブリクエストスケジューリングの概念図

地球シミュレータの運用ポリシーでは，高性能プロセッサ，高性能ネットワークを他のユーザのプログラム実行の影響を受けずに利用できる環境を提供する．これによって，ユーザはアプリケーションプログラムの実行では，さまざまなシステムの擾乱を考慮する必要がないので，高いレベルでのチューニングを行うことができる．このようなチューニングレベルの高いアプリケーションプログラムを多く実行することで，システム全体の利用効率が向上し，貴重な CPU 時間を有効活用する運用が可能となる．

実際に，ユーザリクエストがどのように実行されるかを以下に示す．最低限必要なキューイングシステムへの指示は

・利用ノード数
・利用時間（elapse time）
・使用ディスク容量

の 3 つである．ユーザは，この指示情報の付いたスクリプトをキューイングシステムへ投入する（**図 -2**）．この時に，投入するキューはノード数や利用時間等に関係なく単一のキューである．この単一のキューに投入されたリクエストはキューイングシステムのスケジューラが受け取り，リクエストを「いつ」「どのノードへ」配置するかを決定する．スケジューラは ES 640 ノード中，特別な用途に用いる 16 ノードを除いた 624 ノードを管理し，投入されたリクエストを効率よく実行するように配置する．運用効率を上げるためにユーザの 1 リクエストの最大ノード数は 512 ノード，利用時間は最大 12 時間に制限している．また，ノード数の多いリクエストについてはノード数と占有時間の積で 1,536 ノード・時間の制限も行っている．このため，12 時間の実行が可能なリクエストは 128 ノードまでであり，最大の 512 ノードの場合には 3 時間に限定される．この制限によって，1 つのリクエストで多大な計算リソース（ノード・時間）を占有することを防ぎ，ユーザリクエストをできるだけ公平にスケジュールしている．

ジョブリクエスト実行ごとに，スケジューラから割り

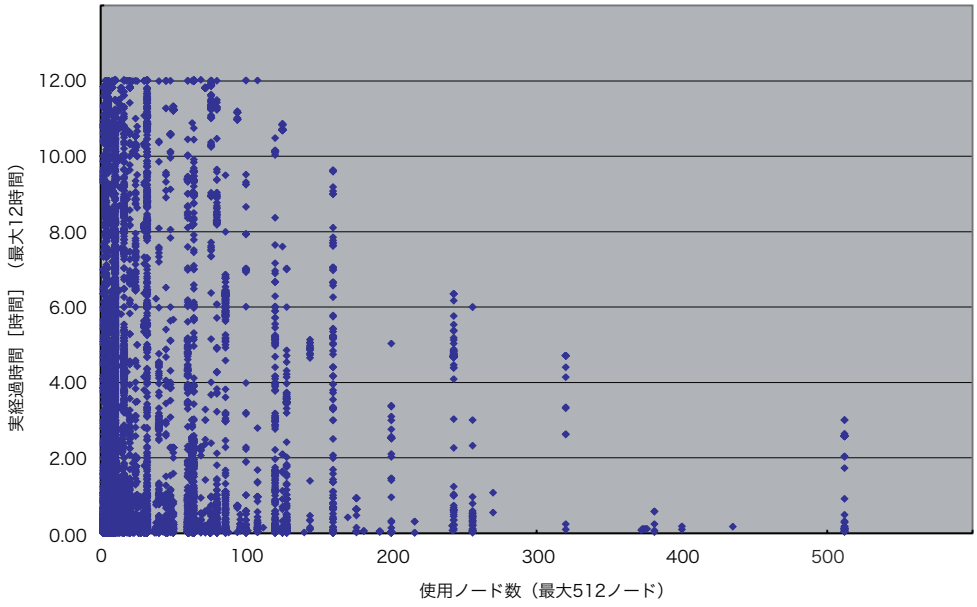

図 -3　平成 15 年 4 月から 9 月までのリクエストの要求ノード数と実経過時間

当てられるノードは変わるので，各ノードのローカルなストレージエリアにユーザデータを保存することは意味がない．また，640 のノードでディスクを共有することはパフォーマンスの点で不利である上，「他のジョブの影響は受けずに利用できる環境を提供する」という運用ポリシーに反する．そこで，ES ではユーザデータはすべて実行ノードの外部にあるストレージに保存し，リクエストの割り当てノードが決定した段階でそのノードで必要なデータを外部ストレージからコピーする（リコール処理）．逆にリクエスト終了時に外部ストレージへデータを戻す処理（マイグレーション処理）を行う．

ジョブのスケジューリング効率は，実行されるジョブの特性に大きく影響を受ける．特に ES では，経過時間をベースにジョブのスケジューリングを行っているため，ユーザの宣言経過時間と実際に使用された実経過時間の関係は重要である．スケジューラは投入されたリクエストのノード・時間を確保し，これらを積み重ねた全リクエストの実行予定表を作成する．しかし，実際にはユーザリクエストは宣言経過時間の前に終了する☆1．このため，リクエストが終了するごとに実行予定表に空きができ，スケジューラは再配置を試みる．ユーザ宣言経過時間と実経過時間に大きなずれがなければ再配置に与える影響は少ないが，両者の時間が著しく違う場合には，再配置によって実行予定表は大きく書き変えられる．この場合に，リクエストを実行するノードが変わらなければ，単に開始予定時刻を早めればよいが，実行ノードの割り当ても変える場合には，ノード間のファイル転送の時間も含めて考慮し，開始予定時刻が早められると判断した場合にはこのリクエストに対する再配置を行う．

ここまでは，リクエストの計算リソース（ノード・時間）によるスケジューリングについて述べたが，リクエストが使用するディスク容量もスケジューリングに影響する．リクエストが CPU やネットワークといった計算資源を占有する時間はリクエストの実行中だけであるが，ディスクを占有する時間はリコール処理が行われてからマイグレーション処理が行われるまでである．

地球シミュレータの利用実態

ES 上で実行されているリクエストについて調査を行った．ここでは，平成 15 年 4 月から 9 月までの 6 カ月間に ES 上で実行されたリクエストのうち，1 分以上の実経過時間で正常終了した 2 ノード以上を使用したリクエストを選択した．対象となるリクエストは 26,912 件であった．

まず，リクエストで使用するノード数と実経過時間について調査した．**図 -3** に使用ノード数と実経過時間の関係を示す．図 -3 から 10 ノード以下のリクエストが，大部分を占めていることが分かる．これは，ES では使

☆1 宣言時間を超過したリクエストはその時点で打ち切りとなり，直ちにマイグレーション処理に移る．

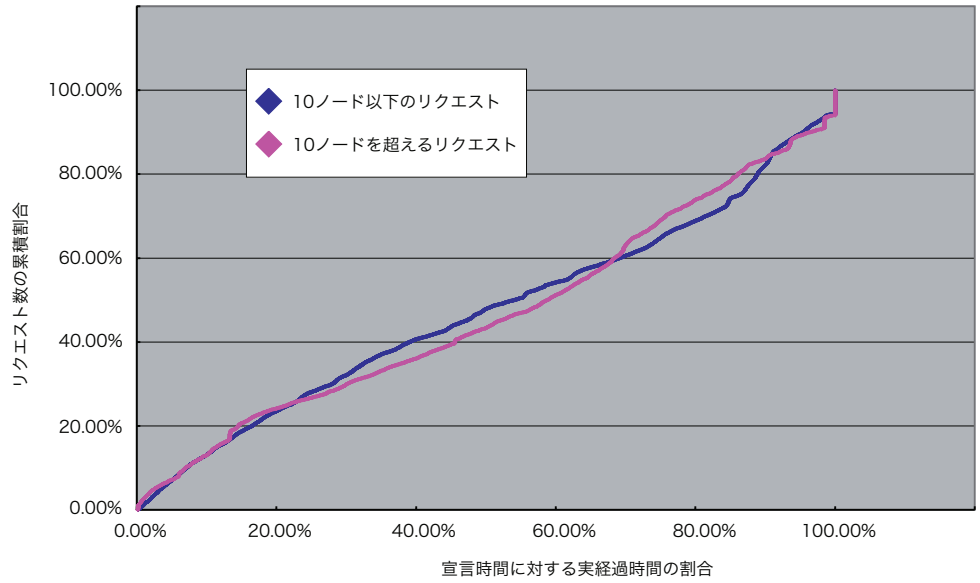

図 -4　宣言経過時間に対する実経過時間の割合の分布

用開始時には 10 ノードまでしか利用することができないという制限があるためである．10 ノードを超える大規模リクエストを実行するためには，ユーザは 10 ノード以下でプログラムを十分にチューニングしなくてはならない．そのため，経過時間の短い 10 ノード以下のリクエストが多くなっている．10 ノードを超えるリクエストには，特定のノード数で実行されるものが多い．これは，大気モデルや海洋モデル等の特定のリクエストで，プロダクトランを行っているリクエストである．また，占有するノード数と時間積による制限で 128 ノードを超えた実行は最大実行時間が減少し，最大ノード数 512 ノードでは最大 3 時間までとなっている．

図 -4 にユーザの宣言経過時間と実経過時間の関係を示す．この図は，横軸に宣言経過時間に対する実経過時間の割合，縦軸に累積リクエスト数の割合とした．26,912 件のリクエストを 10 ノード以下と 10 ノードを超えるリクエストに分けて集計した．

おおまかには一定に単調増加している．これは，宣言時間を完全に使い切るアプリケーションからそうでないものまで広く分布していることを示している．しかし，10 ノード以下と 10 ノードを超えるリクエストでは，若干違いがみられる．10 ノード以下のグラフは立ち上がりが早く，リクエストでは宣言時間に対して短かい時間で終了するものが多いことが分かる．特に 10 ノード以下のリクエストの場合，宣言時間の 10% 未満で終了するリクエストが多い．これは，チューニングを目的とした実行が多いためと思われる．一方，10 ノードを超えるリクエストの場合，宣言時間の 70% を超えるリクエストが 10 ノード以下のリクエストの場合と比べて多くなっている．ES では，リクエストのスケジューリングアルゴリズム上，宣言経過時間が短い方がリクエストが実行されるまでの時間が短くなる．また，ノード数と宣言経過時間の積での実行制限もあるため，大規模リクエストを効率よく実行するには，実行可能な最大経過時間ぎりぎりまで計算を行う必要がある．大規模リクエストでプロダクトランを行っているユーザは，これらのことを理解し，効率よくリクエストを実行している．

以上のことから，ES 上で実行されているジョブには，宣言経過時間と実経過時間に大きな差があり，使用するノード数にも偏りがあることが分かる．特に，大規模ジョブは特定のアプリケーションに偏っており，その多くはプロダクトランを行っている．ES 用ジョブスケジューリングアルゴリズムは，これらの特徴を基にシステム全体でノードの利用効率を上げ，ユーザリクエストのターンアラウンドタイムを短くするようにチューニングを行っている．

参考文献

1）Uno, A., Aoyagi, T. and Tani, K.: Job-scheduling on the Earth Simulator, NEC Research & DEvelopment, Vol.44, No.1, pp.47-52（2003）.
2）宇野篤也，板倉憲一：地球シミュレータ用ジョブスケジューリングアルゴリズムの評価，情報処理学会 HPC 研究会 2003-HPC-95（15），pp.83-88（2003）.

（平成 15 年 12 月 22 日受付）