Vol.2012-HPC-136 No.22
2012/10/4

# ファイルステージングのあるジョブスケジューリングの評価

宇野 篤也[1]　庄司 文由[1]　横川 三津夫[1]

**概要**：スーパーコンピュータ「京」や地球シミュレータなどの大規模システムでは，計算ノードのファイル I/O 性能を確保するために 2 階層のファイルシステムを採用しており，ジョブ実行の一連の作業としてファイルシステム間でファイルを移動させるファイルステージング機構をジョブスケジューリングに組み込んでいる．本稿では，ファイルステージングがジョブスケジューリングに与える影響等についてソフトウェアジョブシミュレータを用いて評価したので報告する．

## 1. はじめに

スーパーコンピュータ「京」[1] や地球シミュレータ [2][3] などの大規模システムでは，計算ノードのファイル I/O 性能を確保するため，プログラムが実行される際に使用するファイルシステム（以下，ローカルファイルシステム）と，ユーザデータを格納するファイルシステム（以下，グローバルファイルシステム）で構成される 2 階層ファイルシステムを採用している (図 1)．単一のファイルシステムで構成されるシステムの場合，万オーダーの計算ノードからのアクセスへの対応と大容量の両方を同時に実現するためのシステムを構成するには，高いコストが必要となる．2 階層の構成をとることにより，単一のファイルシステムで構成するよりもコストを低く抑えつつ，ジョブ実行中の I/O 性能の最大化と大容量を両立させることが可能となる．

ローカルファイルシステムとグローバルファイルシステムの間は InfiniBand 等のネットワークで結合されることが多いが，ローカルファイルシステムの帯域に比べて狭く構成されていることが多い．そのため，2 階層ファイルシステムの場合には，単一ファイルシステムの場合よりもジョブ実行の準備に必要な時間が長くかかる．これを回避し，システムの利用効率を高く維持するために，backfill 等のアルゴリズムを使い，ファイルステージングによるシステム利用効率への影響を最小限にするようにコントロールすることがジョブスケジューラに求められている．

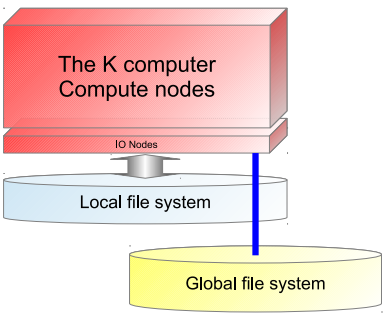


**図 1**　スーパーコンピュータ「京」のストレージ構成

「京」は 2011 年 4 月から試験利用を開始し，一部ユーザに実際に利用してもらいながら運用面での様々なパラメータを模索している．その中でも，ジョブスケジューリングについてはユーザに与える影響も大きく，容易に設定を変更することは難しい．そこで我々は，ソフトウェアジョブシミュレータ [4][5] を用いて各種パラメータを評価する事を考えている．

本稿では，その初期段階としてソフトウェアジョブシミュレータを用いて，ステージング帯域やステージング処理がジョブスケジューリングにどのような影響を及ぼすかについて評価したので報告する．

## 2. ソフトウェアジョブシミュレータ

今回の評価で使用したソフトウェアジョブシミュレータについて説明する．このシミュレータは以下を前提に開発されている．

- 対象はバッチジョブ
- ジョブの実行中はノードを専有
- CPU 時間ではなくユーザが宣言した経過時間をベースにスケジューリング

今回使用したスケジューラのジョブ状態遷移図を図 2 に示す．

ユーザがシステムにジョブ実行を依頼し (Submit)，バッチキューにジョブが投入されると (Accept)，ジョブキューにエントリーされる (Queued)．スケジューラは，定期的

[1] 理化学研究所 計算科学研究機構
Advanced Institute for Computational Science, RIKEN

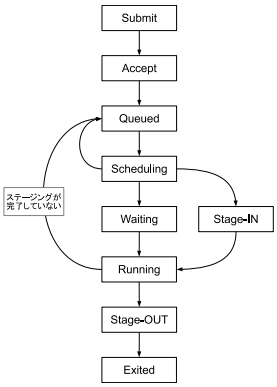


**図 2** スケジューラのジョブ状態遷移図

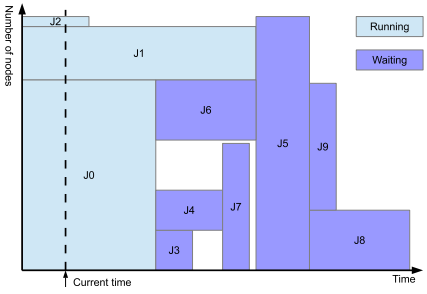


**図 3** Backfill アルゴリズム

にジョブが要求するリソース (計算ノードとローカルディスク) が確保できるか調べる (Scheduling)．リソースが確保できなかった場合には，そのジョブはリソースが確保できるまで待機する (Queued)．リソースが確保できた場合は，ステージングを開始し (Stage-IN)，実行開始時刻まで待機する (Waiting)．実行開始予定時刻までにステージングが完了した場合には，予定どおりジョブを実行する (Running) が，実行開始予定時刻までにステージングが完了しなかった場合には，計算ノードだけを開放し実行中のステージングが完了するまで待機し (Queued)，ステージング完了後に改めてスケジューリングを行う．この時，ステージング済みの計算ノードが優先的に利用されるようにスケジューリングを実施する．

実行が終わったジョブは，実行終了時点で計算ノードを開放し (Stage-OUT)，順次ステージアウトを行う．そして，ステージアウト終了後にディスクを開放する (Exited)．

スケジューリングアルゴリズムとファイルステージングの詳細について，以下に説明する．

### 2.1 スケジューリングアルゴリズム

今回の評価では次の 2 つのアルゴリズムを組み合わせて使用した．

- First Come First Served (FCFS)
- Backfill

FCFS はジョブが投入された順序で優先順位を付けて処理を行うアルゴリズムである．Backfill はノードの利用効率を上げるために，アウトオブオーダーでスケジューリングを実施するアルゴリズムである (図 3)．このアルゴリズムは，リソースに空きがある限りスケジューリングを行うため，比較的少ないリソースを要するジョブが先行してスケジューリングされ，多くのリソースを必要とするようなジョブは延々と待たされるという現象が起きやすい．これを回避するため，投入されてから一定時間実行されないジョブがある場合には，一時的に新規ジョブのスケジューリングを停止してリソースを確保するような制御を行なっている．

今回は簡単のため，計算ノードの選択には計算ノード間の位置関係を考慮せず，利用可能な計算ノードを順次割り当てるという単純なアルゴリズムを使用している．ただし，ステージング済みジョブについては，前述のようにステージング済みの計算ノードを有効活用するように計算ノードの選択を行う．

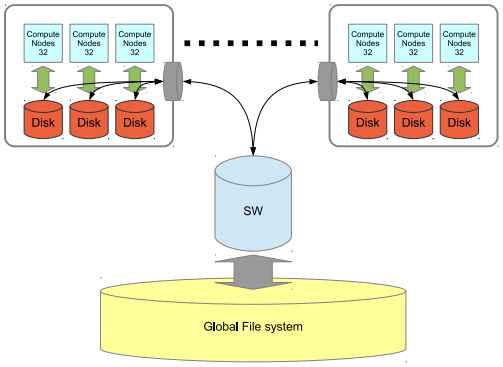


**図 4** シミュレーションモデル

**表 1** 評価に用いたシミュレーションモデルの各パラメータ値

| | |
|---|---|
| 総ノード数 | 9,216 |
| 1 クラスタのノード数 | 96 |
| 1 クラスタのローカルディスク数 | 3 |
| 1 クラスタの出力帯域 | 3,277 (MB/s) |
| 1 ノードあたりのディスク容量 | 128 (GB) |
| 1 ローカルディスクの帯域 | 2,400 (MB/s) |
| ファイルステージングの帯域 | 314,592 ～ 19,662 (MB/s) |

### 2.2 ファイルステージング

グローバルファイルシステムからローカルファイルシステムへファイルを転送することをステージインと呼ぶ．逆に，ローカルファイルシステムからグローバルファイルシステムへファイルを転送することをステージアウトと呼ぶ．

このジョブシミュレータではファイルステージングは複数同時に実行されることを想定している．そのため，スケジューラは最初のステージングは処理時間を見積もらずに実行し，再スケジューリング時には過去のステージングで要した処理時間を参考にしてジョブスケジューリングを行

**表 2** 評価で用いたジョブミックスの特性

| ジョブタイプ | L | M | S |
|---|---|---|---|
| ノード数 (総数に対する%) | 50〜70 | 10〜50 | 0.1〜5 |
| 宣言経過時間 (H) | 6〜8 | 3〜6 | 0.5〜3 |
| ステージイン ファイルサイズ (GB) (ノードあたり) | 16 | 10〜14 | 4〜9 |
| ステージアウト ファイルサイズ (GB) (ノードあたり) | 32〜42.8 | 16〜36 | 4〜9 |
| 投入時間帯 | 常時 | 常時 | 9〜21(80%) 残り (20%) |
| 投入期間 (H) | 96 | 96 | 96 |
| 出現割合 (%) | 5 | 35 | 60 |

なっている．

## 3. スケジューリング評価

まず，今回のジョブスケジューリング評価で使用したシミュレーションモデル及びジョブ特性について説明する．

### 3.1 シミュレーションモデル

今回の評価で使用したシミュレーションモデルを図 4 に，各種パラメータを表 1 に示す．

図 4 に示すように，各クラスタは 96 台の計算ノードと，3 台のローカルディスクで構成され，32 台の計算ノードが 1 つのローカルディスクを共有している．ローカルディスクは，ネットワークスイッチを介してグローバルディスクに接続されている．総ノード数と総ローカルディスク数はそれぞれ，9,216 台，288 台で，ローカルファイルシステムの総帯域は 675 (GB/s) である．

グローバルファイルシステムについては，本来ならばジョブ単位でのアクセス集中を考慮したシミュレーションを行うべきであるが，今回は簡単のため，グローバルファイルシステムはファイルステージングの最大帯域よりも十分に大きいと仮定し，アクセス集中による性能低下は発生しないものとした．

### 3.2 ジョブミックス

今回の評価で使用したジョブミックスの特性を表 2 に示す．ジョブタイプとして L，M，S の 3 種類のジョブを用意した．このうち，ジョブタイプ S はチューニングジョブ，またはデバッグジョブのような小規模短時間のジョブで，日中に集中して投入されることが多いジョブを想定したものである．

評価にはこの特性をもつ 4 日分のジョブミックスを使用した．ジョブの実行時間は，ジョブタイプ L，M は宣言経過時間の 8 割前後とし，ジョブタイプ S はその性質から 3〜10 割と幅を持たせた．ジョブ数はノード実行時間積の累積がシステムのノード時間積に一致するようにし，ノード単位のディスククオータはステージインとステージアウトのファイルサイズの和の 1.1 倍とした．

**表 3** ステージング帯域を変化させた場合の 4 日間で実行されたジョブの割合 (%)

| ステージング帯域 | S | M | L | 全て |
|---|---|---|---|---|
| 100% | 98.7 | 89.2 | 77.8 | 95.8 |
| 50% | 94.0 | 87.5 | 77.8 | 92.0 |
| 25% | 97.7 | 88.3 | 72.2 | 94.6 |
| 12.5% | 98.4 | 85.0 | 72.2 | 94.4 |
| 6.25% | 95.8 | 85.0 | 66.67 | 92.3 |

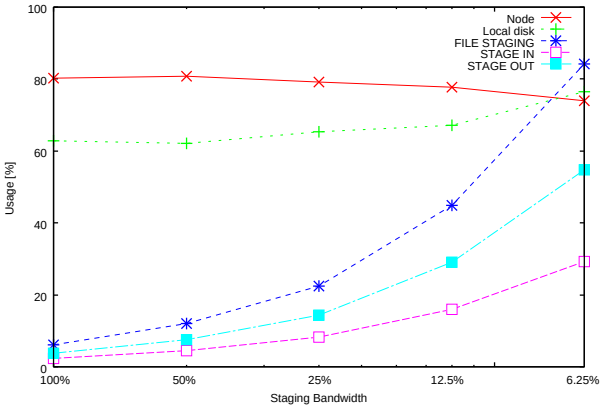


**図 5** ステージング帯域を変化させた場合のリソース使用率

### 3.3 ソフトウェアジョブシミュレータによる評価

今回は，ステージング帯域を変化させた場合のジョブスケジューリングへの影響と，ファイルステージング処理 (ステージイン，ステージアウト) に優先順位をつけた場合のジョブスケジューリングへの影響について評価した．

#### 3.3.1 ステージング帯域

ステージング帯域がジョブスケジューリングに及ぼす影響を，各クラスタからの総出力帯域を基準として，100% (基準値)，50% (1/2)，25% (1/4)，12.5% (1/8)，6.25% (1/16) と変化させて評価した．このシミュレーションでは，ファイルステージングの多重度 (同時実行数) はステージイン，ステージアウトに関係なく無制限とした．

まず，システム側からみたジョブスケジューリングの評価として，各リソース (ノード利用率，ローカルディスク利用率，ステージング帯域) の利用率を調べた．

図 5 に各ステージング帯域における各リソースの平均利用率のグラフを，表 3 に 4 日間で実行されたジョブの割合を示す．シミュレーション自体は全てのジョブの実行が終了 (ステージアウト) するまで実施しているが，そのうち最初の 4 日間にジョブ実行が開始されたジョブの割合を表 3 に示している．参考までに，図 6〜図 10 に同一ジョブミックスを投入した場合のステージング帯域毎のノード利

**表 4** ステージング帯域を変化させた場合の平均実行待ち時間 (H)

| ジョブタイプ | S | | | M | | | L | | | 全て | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ステージング帯域 | 開始 | 終了 | 待ち | 開始 | 終了 | 待ち | 開始 | 終了 | 待ち | 開始 | 終了 | 待ち |
| 100% | 2.38 | 0.05 | 2.43 | 10.77 | 0.24 | 11.01 | 15.50 | 0.44 | 15.94 | 4.76 | 0.11 | 4.87 |
| 50% | 2.28 | 0.05 | 2.33 | 11.39 | 0.24 | 11.63 | 14.63 | 0.55 | 15.18 | 4.80 | 0.11 | 4.91 |
| 25% | 2.0 | 0.08 | 2.08 | 11.21 | 0.40 | 11.61 | 21.10 | 1.14 | 22.24 | 4.78 | 0.19 | 4.97 |
| 12.5% | 1.95 | 0.19 | 2.14 | 12.71 | 0.82 | 13.53 | 18.53 | 2.34 | 20.87 | 5.00 | 0.41 | 5.41 |
| 6.25% | 3.35 | 0.90 | 4.25 | 15.16 | 3.28 | 18.44 | 21.46 | 5.67 | 27.13 | 6.69 | 1.61 | 8.30 |

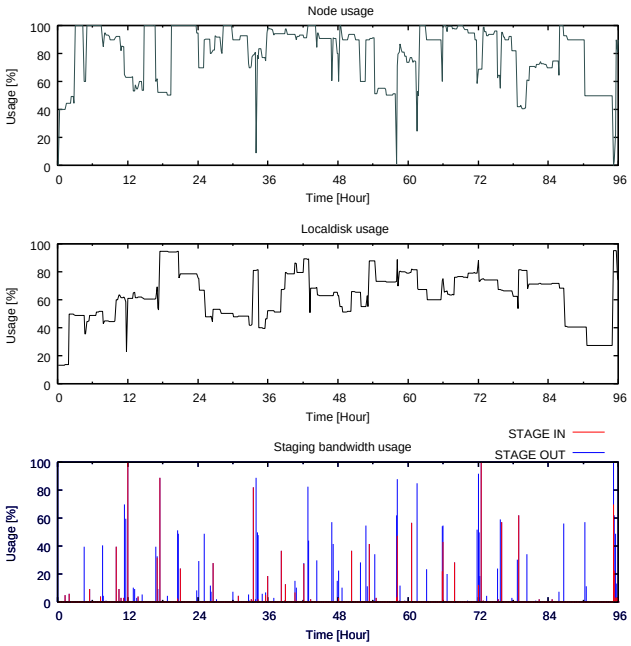


**図 6** ステージング帯域が 100%の場合

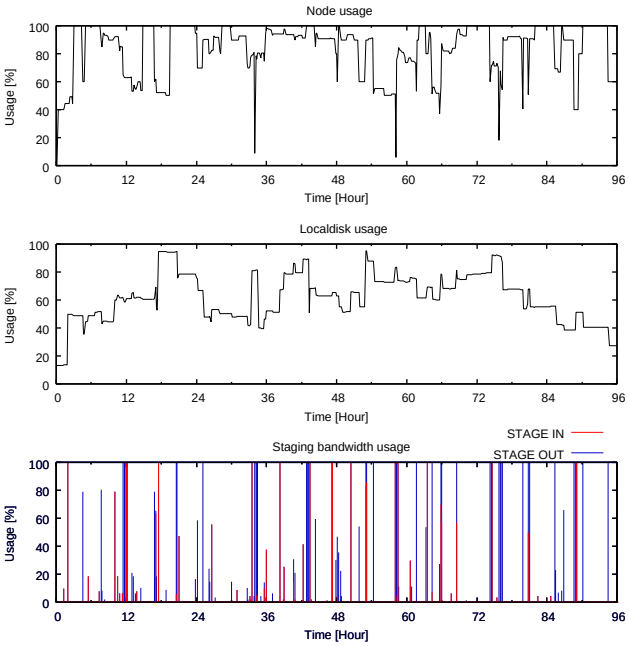


**図 7** ステージング帯域が 50%の場合

用率，ローカルディスク利用率，ステージング帯域利用率の推移グラフを示す．

図 5 のグラフからわかるように，ステージング帯域に余裕がある場合にはノード利用率やローカルディスク利用率への影響はほとんど見られないが，ステージング帯域に余裕がなくなる 6.25% では，ローカルディスクの利用率が上昇し，ノード利用率が低下している．今回使用したジョブミックスのステージング総量 (ステージインとステージアウトの総和) とシミュレーション期間 (4 日間) から，シミュレーション期間内に全てのジョブを実行すると仮定した場合にステージングに必要となる帯域を計算すると 約 22(GB/s) となる．これは，6.25%=19,662(MB/s) にほぼ一致する．この値は計算値よりも 1 割程度低いが，これはスケジューリング効率が影響していると考えられる．今回は非常に単純なノード選択アルゴリズムを使用したためノード利用率が高くなっているが，実際の運用ではノードの位置関係を考慮してスケジューリングを行うことが多いため利用率は低くなる傾向にあり，計算で求められる値よりも狭いステージング帯域でもシステムを効率よく運用することは可能と思われる．

次に，ユーザ側からみたスケジューリングの評価を行っ

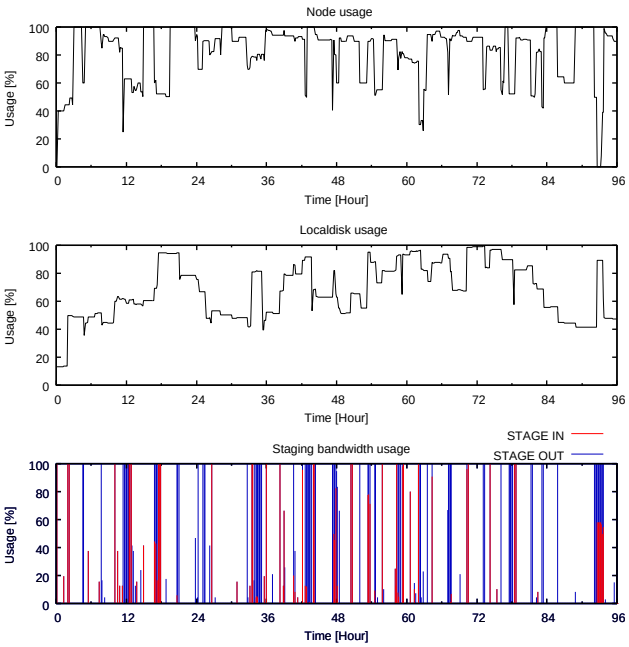


**図 8** ステージング帯域が 25%の場合

**表 5** ステージング処理に優先順位をつけた場合の 4 日間で実行されたジョブの割合 (%)

| ステージング帯域 | 優先順位 | S | M | L | 全て |
|---|---|---|---|---|---|
| 25% | 同じ | 85.59 | 89.74 | 85.71 | 86.62 |
| 25% | IN | 85.59 | 89.74 | 85.71 | 86.62 |
| 25% | OUT | 85.59 | 89.74 | 85.71 | 86.62 |
| 6.25% | 同じ | 86.49 | 82.05 | 85.71 | 85.35 |
| 6.25% | IN | 84.68 | 82.05 | 85.71 | 84.08 |
| 6.25% | OUT | 85.59 | 79.49 | 85.71 | 84.71 |

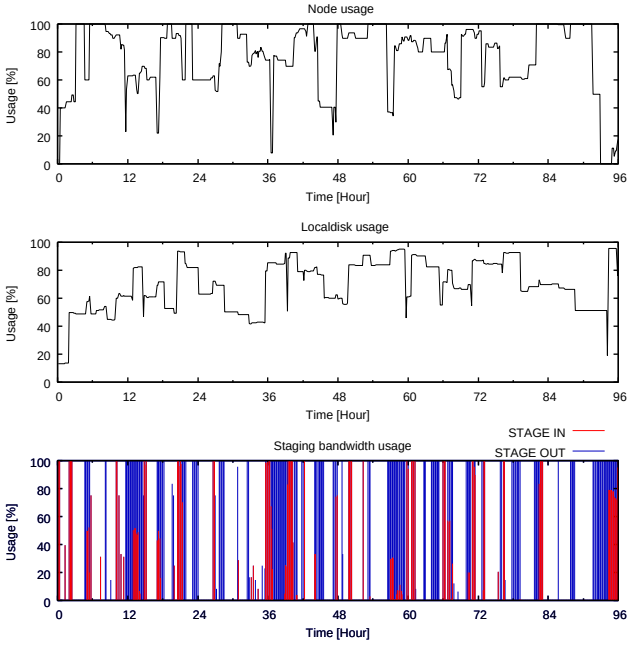


**図 9** ステージング帯域が 12.5%の場合

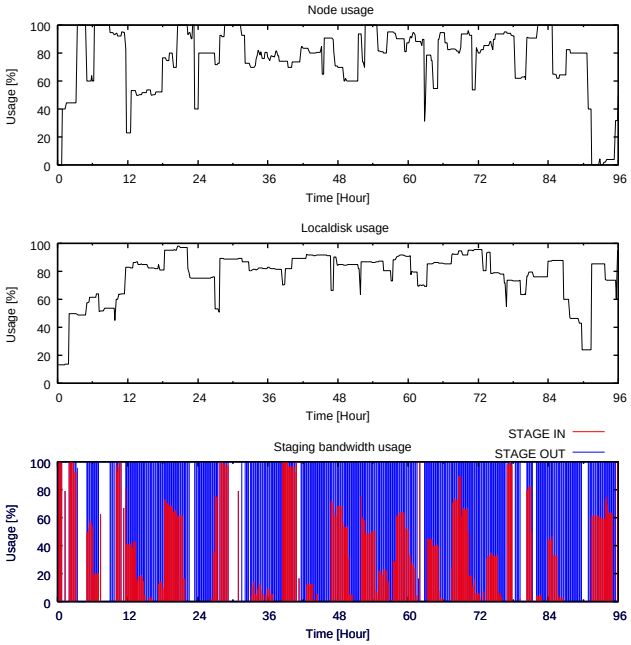


**図 10** ステージング帯域が 6.25%の場合

た．評価基準として，以下の待ち時間を定義する．

- ジョブ開始待ち時間 = ジョブ開始時刻-ジョブ投入時刻
- ジョブ終了待ち時間 = ステージアウト終了時刻-ジョブ実行終了時刻
- ジョブ待ち時間 = ジョブ開示待ち時間+ジョブ終了待ち時間

このジョブ待ち時間が小さいほど，ユーザから見たジョブのターンアラウンドタイムが短くなる．

表 4 にシミュレーションで用いた全てのジョブのジョブタイプ毎のジョブ開始待ち時間，ジョブ終了待ち時間，ジョブ待ち時間をそれぞれ示す．

ジョブ開始待ち時間はノードスケジューリングに大きく依存する．特にステージング帯域に余裕がある 100%～12.5% では，ステージインに要する時間よりもジョブ実行開始までの待ち時間が長いため，ステージング帯域の影響は殆ど無い．一方，ステージング帯域に余裕がない 6.25% では，待ち時間に大きく影響する．

ジョブ終了待ち時間は，今回のシミュレーションではステージアウト時間と等しいため，ステージング帯域に依存した結果となっている．特に，ステージング帯域に余裕が無くなる 6.25% ではその影響が大きく表れている．

結果から，全般的にステージング帯域の大きさに反比例して待ち時間は増える傾向にあることがわかる．しかし，ジョブタイプ毎の詳細，特にジョブ自体の数が少ないジョブタイプ L では，その傾向に当てはまらないものも散見される．これは，今回のシミュレーションでのサンプル数が十分でなく，乱数のゆらぎの影響が表れているためと思われる．

### 3.3.2 ステージング優先順位

次にステージイン処理とステージアウト処理の優先順位を変えることで，スケジューリングにどのような影響が現れるか評価した．ステージング帯域に余裕がある場合 (ステージング帯域が 25%の場合) と，ステージング帯域をほぼ使い切る場合 (ステージング帯域が 6.25%の場合) で実施した．また，ステージング処理に差異をつけやすくするために，ステージング処理の多重度を 3 としている．

表 5 に最初の 4 日間で実行されたジョブの割合を，表 6 に各リソースの利用率を，表 7 に全てのジョブの平均待ち時間をそれぞれ示す．

これらの表からわかるように，帯域に余裕がある場合にはステージング処理の優先順位による影響はほとんど見られない．一方，帯域に余裕のない場合には，その影響が現れている．特にステージアウト処理を優先的に処理した場合，ノードの利用率が 25% の場合に比べて低下している．これは，ステージアウトを優先するために，次のジョブの準備 (ステージイン処理) が間に合わない場合が発生するためである．

**表 6** ステージング処理に優先順位をつけた場合のリソース利用率 (%)

| ステージング帯域 | 優先順位 | ノード | ローカルディスク | ステージング帯域 | (IN) | (OUT) |
|---|---|---|---|---|---|---|
| 25% | 同じ | 78.59 | 67.65 | 44.78 | 15.67 | 29.11 |
| 25% | IN | 78.59 | 67.06 | 43.92 | 15.26 | 28.66 |
| 25% | OUT | 77.79 | 66.43 | 43.16 | 14.69 | 28.47 |
| 6.25% | 同じ | 72.39 | 72.69 | 81.26 | 27.25 | 54.01 |
| 6.25% | IN | 75.09 | 76.24 | 83.02 | 28.07 | 54.95 |
| 6.25% | OUT | 69.87 | 71.16 | 77.29 | 25.83 | 51.46 |

**表 7** ステージング処理に優先順位をつけた場合の平均実行待ち時間 (H)

| ジョブタイプ | | S | | | M | | | L | | | 全て | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ステージング帯域 | 優先順位 | 開始 | 終了 | 待ち | 開始 | 終了 | 待ち | 開始 | 終了 | 待ち | 開始 | 終了 | 待ち |
| 25% | 同じ | 1.68 | 0.09 | 1.77 | 10.46 | 0.38 | 10.84 | 19.15 | 1.04 | 20.18 | 4.30 | 0.19 | 4.49 |
| 25% | IN | 1.68 | 0.09 | 1.77 | 10.46 | 0.38 | 10.84 | 19.14 | 1.04 | 20.18 | 4.30 | 0.19 | 4.49 |
| 25% | OUT | 1.63 | 0.08 | 1.71 | 10.40 | 0.39 | 10.79 | 19.03 | 1.04 | 20.07 | 4.25 | 0.19 | 4.44 |
| 6.25% | 同じ | 4.53 | 1.30 | 5.83 | 16.76 | 2.75 | 19.50 | 22.73 | 5.62 | 28.35 | 7.97 | 1.78 | 9.75 |
| 6.25% | IN | 3.34 | 1.75 | 5.09 | 14.51 | 3.18 | 17.69 | 21.99 | 5.23 | 27.21 | 6.55 | 2.19 | 8.74 |
| 6.25% | OUT | 7.57 | 0.65 | 8.22 | 18.93 | 2.20 | 21.13 | 27.73 | 5.68 | 33.41 | 10.88 | 1.18 | 12.06 |

一方，ユーザ側からみた場合，ステージアウトを優先する場合にはジョブ終了後に速やかにステージアウトが実行されるため，体感的な待ち時間は短く感じられると思われる．

## 4. おわりに

本稿では，ファイルステージングのあるジョブスケジューリングの評価を行った．ソフトウェアジョブシミュレータを用いて，ステージング帯域やステージング処理の優先度がジョブスケジューリングに与える影響を評価した．シミュレーションにより，目標とするジョブの処理量から必要なステージング帯域を決めることができ，ステージング処理の優先度を調整することでノード利用率を改善することが可能なことがわかった．

一方，今回のシミュレーションでは実行回数 (サンプル数) が不十分なため，乱数によるゆらぎを十分に補正することができていない．また，シミュレーションモデルにも簡単化した部分もある．今後は，これらの改善やフェアシェア等の他のアルゴリズムを組み入れた評価等を実施していきたい．

### 参考文献

[1] 特集：スーパーコンピュータ「京」，情報処理，Vol. 53, No. 8, pp.752–pp.807.
[2] 特集:地球シミュレータ, 情報処理, Vol. 45, No. 2, pp.113–pp.151.
[3] Ken ’ichi Itakura, Toshiyuki Asano, Atsuya Uno: “ Storage Performance Analysis in ES and ES2. ” SC2008 conference, 2008, Austin.
[4] Atsuya Uno, Tatsuya Aoyagi, Keiji Tani : Jobscheduling on the Earth Simulator, NEC Reseach & Development, Vol.44, No.1, pp.47–52 (2003).
[5] 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, 情報処理学会研究報告, Vol.2003-HPC-95, pp.83–88 (2003).