Vol.2015-HPC-150 No.1
2015/8/4

「京」上のジョブの分析とジョブミックス生成手法の提案

宇野 篤也[1,a)]　関澤 龍一[3]　山本 啓二[1]　若林 大輔[2]　庄司 文由[1]

概要：スーパーコンピュータ「京」の運用は，2012 年 9 月の供用開始から 2 年以上が経過した．2014 年度末時点の登録課題数（終了分も含む）は約 250 課題，ユーザ数では約 1,200 名で，2014 年の 1 年間に処理したジョブは 44 万件以上，1 日平均では 1,200 件以上であった．今回，これら「京」で実行されたジョブの特性について分析・評価を行ったのでその内容について報告する．また，これらのジョブ特性をもったジョブミックスの生成手法を提案する．

1. はじめに

スーパーコンピュータ「京」は，理化学研究所と富士通株式会社が共同開発した汎用並列スーパーコンピュータで，運用は理化学研究所 計算科学研究機構（AICS）が行っている．「京」は国内の大学や研究所に設置されたスーパーコンピュータを高速ネットワークで接続した HPCI システムの中核を成すシステムに位置付けられており，その計算資源は共用法[*1]に基づいて登録機関[*2]で選定された利用者に提供されている．

「京」の運用は，2012 年 9 月の共用開始から 2 年半以上が経過した．2015 年 3 月末時点の登録課題数（終了分も含む）は約 250 課題，ユーザ数では約 1,200 名で，1 日あたりの平均アクティブユーザ数は約 120 名と非常に多くのユーザが利用している．登録課題も戦略プログラム利用枠の 5 分野をはじめとして色々なテーマで利用されており，実行されるジョブの特性も様々である．システムを効率的に運用するためには，実行されているジョブの状況を分析することは重要である．特に，OS やファイルシステム，スケジューラ等の各種パラメータを適切に設定するためには，ジョブの特性（規模や数，ステージングファイルサイズ，実行までの待ち時間等）を把握することが不可欠である．

今回，これら「京」で実行されたジョブの特性について分析・評価を行ったので，その結果について報告する．また，スケジューラ等システムソフトウェアを評価するために有用となる，ジョブ特性を反映させたジョブミックスの生成手法についても検討を行ったので，その生成手法についても述べる．

1　国立研究開発法人理化学研究所 計算科学研究機構
2　株式会社富士通ソーシアルサイエンスラボラトリ
3　富士通株式会社
a)　uno@riken.jp
*1　特定先端大型研究施設の共用の促進に関する法律
*2　登録施設利用促進機関

2. 「京」の概要

2.1 「京」の構成

図 1 に京のシステム構成概要を示す [1]．「京」は，82,944 台の計算ノードと 1.27PiB のメモリ，5,184 台の I/O ノード，11PB のローカルファイルシステム (LFS) と 30PB のグローバルファイルシステム (GFS)，管理用/制御用サーバ，Pre/Post サーバおよびフロントエンドサーバから構成される．

ユーザはインターネット経由で，フロントエンドサーバにログインして「京」を使用する．ユーザファイルは GFS に置かれ，ジョブの実行に関連するファイルは，GFS と LFS 間でシステムによりジョブ実行前後に自動で転送される．GFS から LFS への転送はステージイン，LFS から GFS への転送はステージアウトと言い，両方をまとめてファイルステージングと呼ぶ．ユーザは，ジョブ実行時の

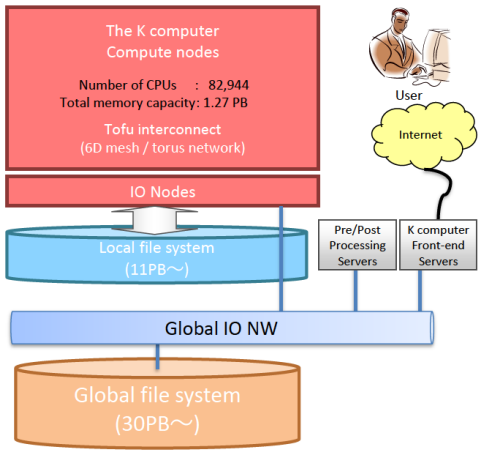

図 1　「京」のシステム構成

表 1 「京」のリソースグループ構成

名称	ノード数	最大経過時間	運用期間
small	1 – 384	24H	大規模時以外
large	385 – 36,864	24H	大規模時以外
huge	36,865 – 82,944	8H	大規模時のみ
micro	1 – 1,152	30min	常時
interact	1 – 384	6H	大規模時以外

ジョブスクリプトに指示を記述することでこれらファイルステージングをシステムに実行させることができる．

2.2 「京」の運用

「京」の基本的な運用方針は，計算資源の利用機会をユーザ間でできるだけ平等にすることである．スケジューリングアルゴリズムには FCFS(First-Come and First-Served) と，バックフィルを採用している．

「京」では計算ノード群を複数のリソースグループ (ジョブキュー) に分割して運用している．**表 1** に「京」で設定しているリソースグループの構成を示す．通常運用では 36,864 ノード以下のジョブを，大規模ジョブ実行期間では 36,865 ノード以上のジョブを実行することができる．大規模ジョブ実行期間は毎月第二火曜からの 3 日間で，期間が終了するか投入された大規模ジョブが全て実行された時点で通常運用に戻る [2][3]．「京」では，前述のとおり基本的にジョブ実行時にファイルステージングを行うが，例外として interact と micro はファイルステージングは必要ない．interact は会話型ジョブ，micro は 1,152 ノード以下かつ最長 30 分までのジョブを実行できる．micro のジョブは large のジョブの隙間を有効活用するのが目的で，ジョブを実行していないノードで直ちに実行できるようにするためにステージングを不要としているが，実行時間は短時間に制限されている．

ジョブの種類では，通常ジョブ，ステップジョブ，バルクジョブ，会話型ジョブをサポートしている．通常ジョブは普通のバッチジョブである．ステップジョブは複数のジョブの実行順序をユーザが定義できるジョブで，先に実行されたジョブの終了条件を使用して次のジョブの動作を決めることができる．バルクジョブは複数のジョブを同時に投入することのできるジョブで，パラメータを変えて複数回実行するアンサンブル計算等で利用されることが多い．

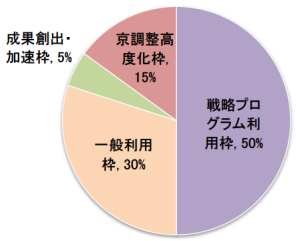

図 2 「京」の計算資源量配分内訳

2.3 「京」の計算資源

「京」が提供する計算資源量の内訳を**図 2** に示す．「京」では，計画停止や保守時間を除いた計算資源量を，戦略プログラム利用枠 (50%)，一般利用枠 (30%)，成果創出・加速枠 (5%)，京調整高度化枠 (15%) に配分している．

戦略プログラム利用枠は，社会的・学術的に大きなブレークスルーが期待できる 5 つの戦略分野で利用される計算資源である．戦略 5 分野とは

- 予測する生命科学・医療および創薬基盤
- 新物質・エネルギー創成
- 防災・減災に資する地球変動予測
- 次世代ものづくり
- 物質と宇宙の起源と構造

の 5 分野で，約 50%が割り当てられている．

一般利用枠は，公募によって決められた，科学的・社会的に優れた成果創出が期待される研究全般を対象としたもので，約 30%が割り当てられている．ここには産業利用課題も含まれている．

成果創出・加速枠は，一般利用及び戦略プログラム利用の課題の中から，早期の成果創出が期待できる課題に対する追加配分の枠で，約 5%が割り当てられている．

京調整高度化枠は，「京」の安全運転のためのシステム調整やユーザ利用支援のための研究開発および幅広い分野のユーザの利用に資する高度化研究に利用される計算資源で，全体の約 15%が割り当てられている．AICS 内の研究チームもこの枠で「京」を利用している．

なお，京調整高度化枠以外の枠は HPCI 課題での利用である．

今回，共用開始後の 2012 年 10 月[*3] から，2015 年 3 月末までの 2 年 6 か月間に「京」で実行されたジョブを対象に分析を行った．

3. ジョブの分析

3.1 ジョブ数と計算資源量

2012 年 10 月から 2015 年 3 月までの 2 年 6 か月間に実行された，ジョブの数と使用された計算資源量の月毎の状況を**図 3** と**図 4** にそれぞれ示す．総ジョブ数は約 980,000 件で，使用された総計算資源量は約 1,200,000,000 ノード時間であった．グラフ中，使用された計算資源量が他の月と比較して少ない月がいくつかある．このうち，2012 年 11 月，2013 年 2，8，12 月，2014 年 7 月はシステムのメンテナンスをそれぞれ 1 週間程度実施し，その間システムが停止していた影響によるものである．一方，2014 年 2 月，2015 年 3 月はローカルファイルシステム障害の影響で長期間システムが停止したため，使用された計算資源量が少

[*3] 共用開始は 2012 年 9 月 28 日だが，9 月中に実行されたジョブは他の月と比較して非常に少ないため，集計から除外している

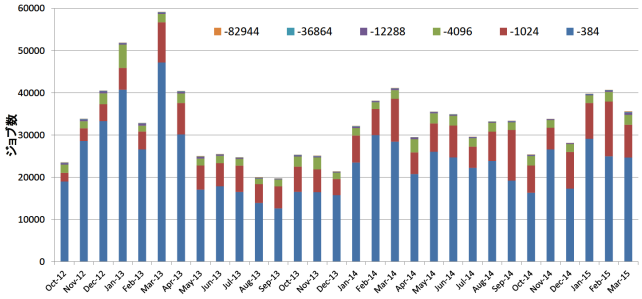

図 3 実行されたジョブ数

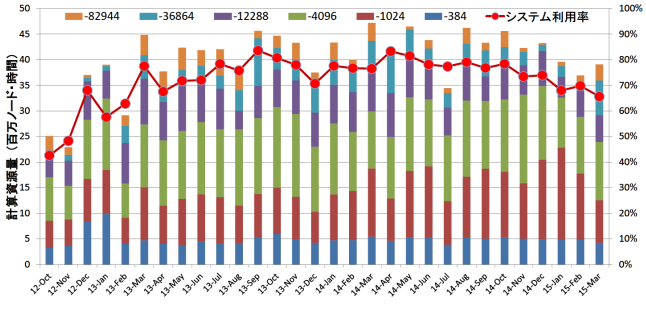

図 4 使用された計算資源量

なくなっている．

グラフの傾向が 2013 年 2 月を境に変わっているが，これは，運用体制を変更したことによるものである．共用開始直後は，計算ノードを複数のリソースグループに分割せず，一つのリソースグループで運用を行っていた．これは，大きなジョブの隙間で小さなジョブが実行され，システム全体の利用効率が向上することを期待したことによるものであるが，実際にはノード数は少ないが経過時間の長いジョブが多くあり，結果として大きなジョブのスケジューリングを阻害し，システム全体の利用率*4を低下させる原因となっていた．これを改善するため，2013 年 2 月に計算ノードを small と large の 2 つのリソースグループに分割した．また，大規模ジョブのノードを確保するために，大規模ジョブ実行直前にシステム利用率が大きく低下する現象が見られたため，large より大きなジョブ (36,865 ノード以上) は huge での実行のみとしシステム全体の利用率の改善を行った．

システム利用率は共用開始後から上昇し，2013 年度後半には約 80%となったが，2014 年度の後半から徐々に低下した．これは，2014 年度後半に投入されたジョブ数が減少した影響によるものである．「京」では，各課題の計算資源を上期（4 月〜9 月），下期（10 月〜3 月）の 2 期に分割して配分していて，上期の計算資源を全て消費した場合，下期の計算資源を利用することができるようになっている．2014 年度は上期だけで年間の割り当て計算資源のほとんどを消費するケースが多くあったため，下期で多くのジョブを実行できなかったことが原因ではないかと考えている．システム利用率は，2013 年度の平均は約 75.9%，2014 年

*4 使用されたノード時間積/利用可能だったノード時間積

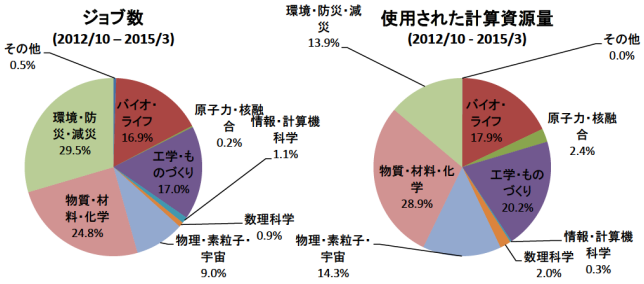

図 5 分野別のジョブ数と使用された計算資源量

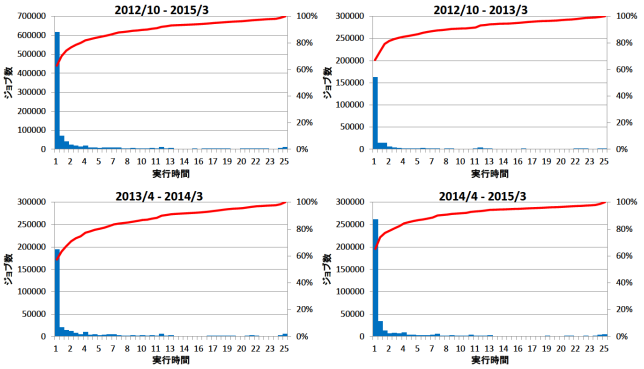

図 6 実行時間別のジョブ数

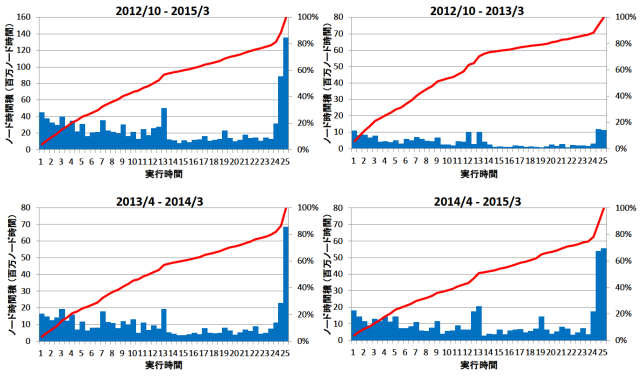

図 7 実行時間別の使用された計算資源量

度は後半に低下したにもかかわらず平均で約 75.3%と高い結果であった．

3.2 ジョブの分野

HPCI 課題は課題申請時に属する分野を記入することになっている．これに基づいて各ジョブを 8 種類の分野に分類した．**図 5** に分野別のジョブ数と使用された計算資源量の割合を示す．

戦略 5 分野に該当する分野の割合が全体的に高いことがわかる．また，「原子力・核融合」や「物理・素粒子・宇宙」といった分野のジョブは，ジョブ毎の使用計算資源量が他の分野と比較して大きいこともわかる．

3.3 ジョブの実行時間

図 6 と**図 7** に実行時間別のジョブ数と使用された計算資源量を示す．棒グラフは 30 分毎の値を，赤線は累積の

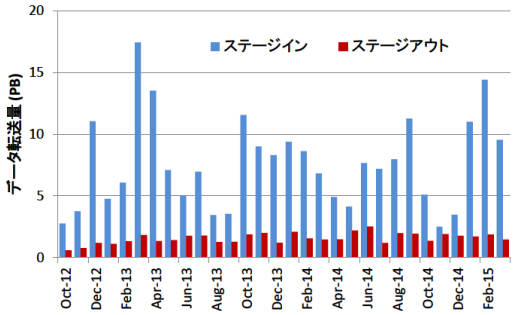

図 8 月毎のデータ転送量

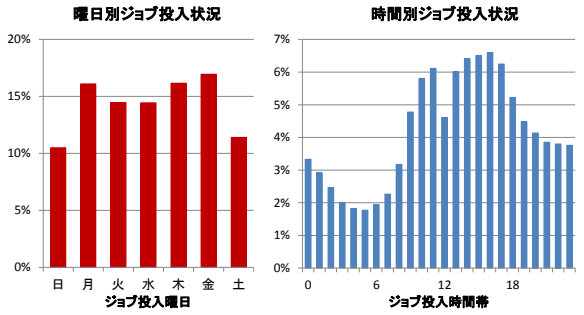

図 10 ジョブの投入時刻

割合をそれぞれ示している．会話型ジョブを含めた全てのジョブを対象に，年度毎と 2 年 6 か月分それぞれについて集計を行っている．グラフの経過時間の上限が 25 時間になっているが，実行時間が 24 時間を超過してシステムにより強制停止されたジョブが 25 時間として計上されている．

約 60%のジョブが実行時間 30 分未満のジョブで，3 時間未満のジョブが約 80%を占めている．2014 年度は 2013 年度と比較して，実行時間 30 分未満のジョブ数が増えているが，これは 2014 年 1 月から導入した mirco のジョブの影響と考えている．一方，利用された計算資源量でみると，4 時間未満のジョブが全体の約 20%を，12 時間未満のジョブが約 55%を占めている．

年度毎の推移を見てみると，ジョブ数の割合はあまり変化がみられないが，計算資源量の割合では年々ジョブ毎の実行時間が長くなっていることがわかる．特に 12 時間や 24 時間付近の計算資源量が他に比べて多くなってきている．これはプログラム開発等ではなく，プロダクトランのジョブが多く実行されていることによるものと思われる．

3.4 ファイルステージング

図 8 に月毎のファイルステージングでのデータ転送量を示す．月平均では，ステージインは約 7.6PB，ステージアウトは約 1.6PB で，最大転送量ではステージインは約 17.5PB，ステージアウトは約 2.5PB であった．

「京」では効率よくデータ転送を行うために，必要最低限のデータをステージングするようにユーザにお願いしている．そのため，ステージアウトの転送量がステージインに比べて非常に少なくなっているものと思われる．

3.5 ジョブの実行待ち時間

図 9 にジョブの実行待ち時間の推移を示す．ユーザが指定したノード数と経過時間別に，通常ジョブがシステムに投入されてから実行されるまでの待ち時間を示している．このグラフでは，大規模ジョブ実行期間等の予め予定されている待ち時間は除外している．また，ステップジョブや会話型ジョブのような通常のジョブとは異なるスケジューリングを行うジョブやステージングを必要としない micro のジョブも除外して集計している．

グラフから分かるように，指定した計算資源の大きさに比例してジョブの待ち時間が増加している．計算資源の大きさに関係なく，全てのグラフに待ち時間が長い個所が何か所かある．前述のとおり，「京」では，ユーザの計算リソースを上期と下期の 2 期に分割して配分している．そのため，各期末が近付くと投入されるジョブ数が増加し，ジョブの実行待ち時間が長くなる傾向がみられる．このように期末に発生する混雑は，配分した計算資源量の総量が，実際に「京」が処理できる資源量を大きく超えたことが原因の一つと推測されたため，2014 年度は配分する計算資源量の総量を 2013 年度比で 85%に変更した．その結果，2014 年度は 2013 年度と比べてジョブの待ち時間は改善されたものの，まだ各期末は混雑する傾向にある．

3.6 ジョブ投入時刻

図 10 にジョブ投入時刻の分布を示す．左が曜日別の投入状況，右が時間帯別の投入状況である．平日に比べて週末のジョブ投入数は少ない．平日では，月曜と金曜が多く，週末に近づくにつれてジョブ投入数が増加する傾向がみられる．時間帯別のジョブ投入状況をみてみると，朝 9 時台からジョブ数が増加し，12 時台に若干減少したあと 16 時台まで引き続き増加している．その後，朝方にかけて緩やかに減少していくという傾向がみられる．つまり，基本的にユーザの業務時間に沿って増減していることがわかる．

4. ジョブミックス生成手法

ジョブスケジューラの性能評価などを行う場合，評価に使用するジョブミックスの特性は非常に重要で，その特性が評価結果に大きく影響する場合がある．ジョブミックスの特性は，計算機センター毎に大きく異なる傾向があるため，ジョブスケジューラの性能等を多角的に評価する場合は，様々な特性のジョブミックスを使用する必要がある．複数の計算機センターのジョブミックスを公開しているサイト [4] もあるが，公開されている情報は古く，規模も小さく，種類も少ない．また，同じ特性を持つジョブミックスを使用する場合でも，統計的な評価を行うためには同一条件で内容の異なるジョブミックスを複数本用意する必要があるため，実データを用いるのは困難である．そこで，

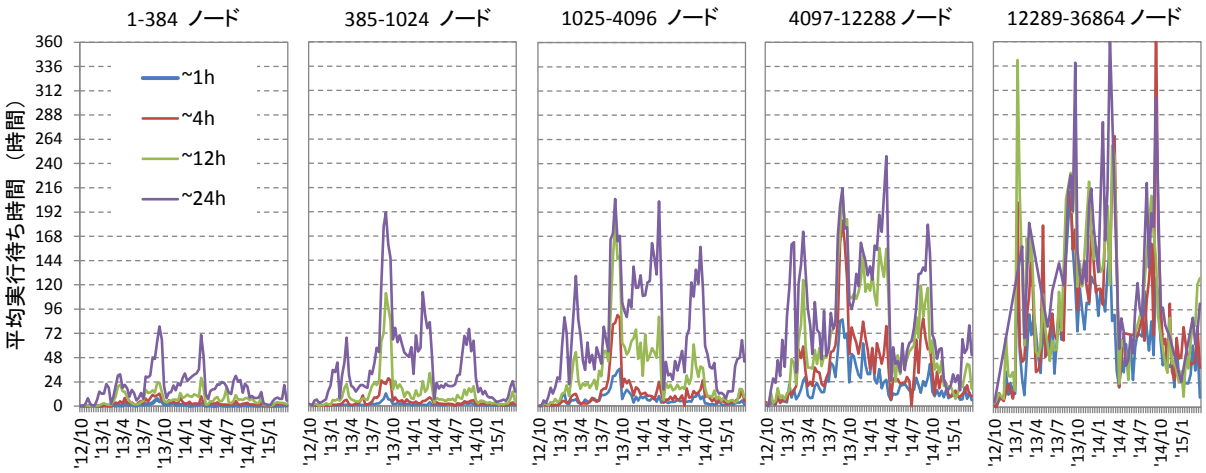

図 9 ジョブの実行待ち時間

表 2 確率分布マップの各パラメータ単位

名称	単位	自由度
ノード数	12 ノード単位 (スケジューリング単位)	1 – 6,912
指定経過時間	10 分単位 (最大 24H)	1 – 144
実行時間	1%単位 (指定経過時間に対する割合)	1 – 100

特定の特性をもつジョブミックスを容易に生成できる手法について検討を行った．

4.1 生成手法

今回，ジョブミックスの生成手法として，

- 統計情報に基づいた生成手法
- 確率分布マップによる生成手法

の 2 種類の手法について検討を行った．基になるデータには，2012 年 10 月から 2015 年 3 月までの間に「京」の large で実行されたジョブ (約 216,000 件) を使用した．

4.1.1 統計情報に基づくジョブミックス生成

統計情報に基づくジョブミックスの生成手法では，ジョブのノード数，指定経過時間，実行時間等のパラメータは正規分布に従うものと仮定した．しかし，large 全体を一つの確率過程で表現するのは困難なので，各パラメータをいくつかのグループに分割し，それぞれが正規分布に従うものとしてデータを作成した．

4.1.2 確率分布マップによるジョブミックス生成

本手法では統計情報に一般的な確率過程を使用せず，代わりに確率分布マップを使用する．確率分布マップでは，各パラメータの組み合わせが発生する確率を一つ一つ計算して求める．例えば，10 ノード，指定経過時間が 10 時間，実行時間が 9 時間のジョブが何件発生したかをカウントする．この場合，パラメータ数が多いと組み合わせ数が膨大になるので，各パラメータはある程度の丸めが必要になる．確率分布マップの作成には多くのサンプルが必要で，確率分布マップに存在しないジョブは決して生成されないという問題点もある．

表 2 に，使用した各パラメータの単位を示す．自由度は $6,912 \times 144 \times 100 = 99,532,800$ 通り となるが，実際に有効な組み合わせは，今回の場合は 42,149 通りであった．

4.2 評価結果

図 11 にオリジナルデータと各生成手法で生成したジョブミックスの統計情報を示す．これらのグラフからも分かるように，どちらの生成手法も統計的な結果はよく一致している．

図 12 に，各データの実際の分布状況を示す．散布図からも分かるように，オリジナルデータの分布状況にはかなり偏りがみられ，統計情報に基づくジョブミックスではその偏りをうまく再現できていないことが分かる．一方，確率分布マップによるジョブミックスでは，オリジナルデータのもつデータの偏りをうまく再現できている．実行時間/指定経過時間の軸上のデータ間隔がオリジナルデータと比較して空いているが，これは今回のジョブミックス生成では実行時間を 1%単位で表現していることに起因するものである．

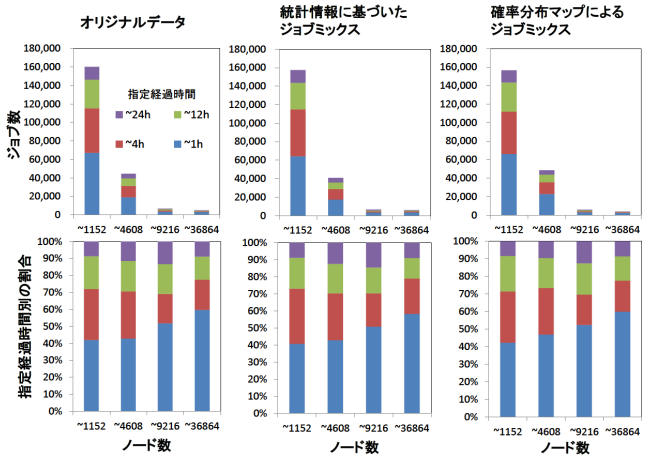

図 11 各ジョブミックスの統計情報

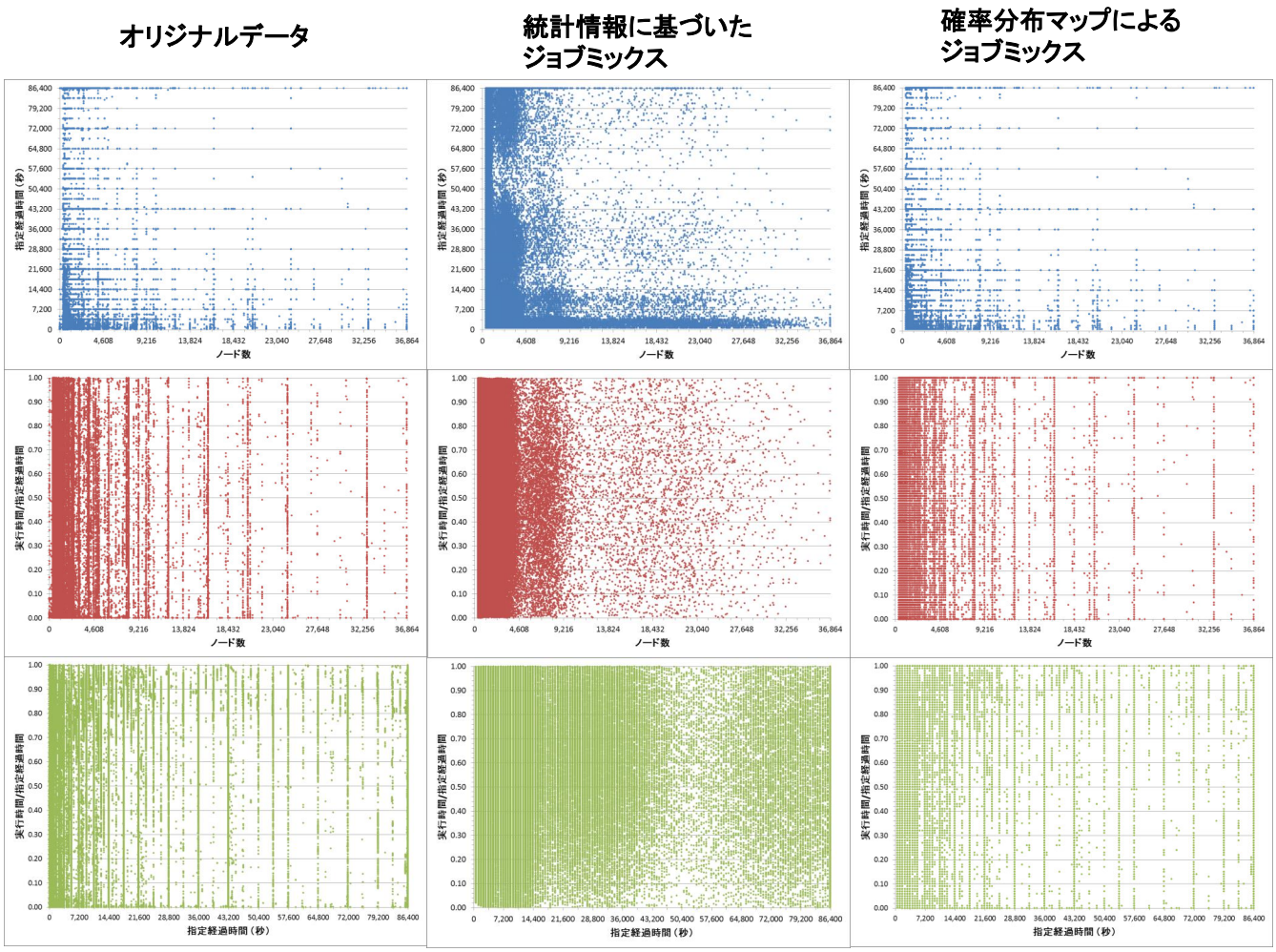

図 12 各ジョブミックスのジョブ分布状況

5. おわりに

本報告では，「京」で実行されたジョブについて分析結果と，そのジョブ特性をもつジョブミックスの生成手法について述べた．分析結果からも分かるように，「京」におけるジョブの特性は偏りを持っており，一般的な確率過程で表現することは難しい．本稿で提案した一般的な確率過程の代わりに確率分布マップを使用する手法では，作成したジョブミックスは元データのジョブ特性を精度よく再現できたが，パラメータの組み合わせによっては自由度が膨大になる場合がある．また，確率分布マップの作成には自由度に見合った十分な量のジョブの情報が必要なことや，確率分布マップに存在しないジョブは生成されないなどの問題点もある．しかし，ジョブスケジューリング等の評価に必要なパラメータはそれほど多くなく，実際のジョブの持つ特性の偏りや，計算機センターで 1 年間に実行されるジョブ数を考えると，確率分布マップを定期的に作成することはさほど困難ではないと思われる．

本稿で提案する手法を用いて，各計算機センター毎の確率分布マップを作成し公開することができれば，様々な面で役に立つのではないかと考えている．

参考文献

[1] 黒川原佳，庄司文由：スーパーコンピュータ「京」システム概要，情報処理，Vol.53，Vol.53，No.8，pp.759-766 (2012).
[2] 山本啓二，宇野篤也，塚本俊之，菅田勝文，庄司文由：スーパーコンピュータ「京」の運用状況，情報処理，Vol.55，No.8，pp.786-793 (2014).
[3] Keiji Yamamoto, Atsuya Uno, Hitoshi Murai, Toshiyuki Tsukamoto, Fumiyoshi Shoji, Shuji Matsui, Ryuichi Sekizawa, Fumichika Sueyasu, Hiroshi Uchiyama, Mitsuo Okamoto, Nobuo Ohgushi, Katsutoshi Takashina, Daisuke Wakabayashi, Yuki Taguchi, Mitsuo Yokokawa: The K computer Operations: Experiences and Statistics, Proceedings of International Conference on Computational Science (ICCS), (2014)
[4] Parallel Workloads Archive (PWA), http://www.cs.huji.ac.il/labs/parallel/workload/