大規模環境のジョブスケジューリングにおけるジョブ分布とトポロジ依存性の評価

高山 沙也加[1] 関澤 龍一[2] 鈴木 成人[3] 山本 拓司[3] 小口 正人[1]

概要：近年の HPC システムでは利用者の増加と利用者層の拡大に伴い, 投入されるジョブ数および必要ノード数は増加しており, また, 多様化している. そのためシステム規模は増加傾向にある. システムの運用において, ユーザの立場ではジョブ投入から実行されるまでの待ち時間が, 運用の立場では利用可能な計算機資源のうち実際に利用された割合である充填率が重視され, 大きな課題となっている. 本研究では, 大規模環境におけるジョブスケジューリングの高充填率と短待ち時間の両立を実現するパーティション分割システムの基礎検討として, ジョブ分布, トポロジ条件を変えてジョブスケジューリングした際の性能評価を行う.

An Evaluation of Job Distributions and Topology Dependency for Job Scheduling on Large-Scale HPC Systems

Sayaka Takayama[1] Ryuichi Sekizawa[2] Shigeto Suzuki[3] Takuji Yamamoto[3] Masato Oguchi[1]

1. はじめに

近年の HPC システムでは利用者の増加と利用者層の拡大に伴い, 投入されるジョブ数およびその必要ノード数は増加しており, また, 多様化している. 2019 年に計算資源の共用を終了したスーパーコンピュータ「京」のジョブ規模別計算資源利用状況を見てみると, 必要ノード数が 1000 を超えるジョブのノード割当時間積が半数以上を占めている [1]. このように, ジョブ規模の増大化に伴い, システム規模は全体的に増加傾向にある. システムの運用において, 図 1 のようにユーザの立場ではジョブ投入から実行されるまでの待ち時間が, 運用の立場では利用可能な計算機資源のうち実際に利用された割合である充填率が重視され, 大きな課題となっている. 待ち時間はシステム側で十分なノード数を用意することで, 充填率は必要ノード数に応じてジョブ受け入れに制限を設けることで改善可能だが, 一方を優先させることで他方が悪化する可能性がある. 投入ジョブの必要ノード数に応じてシステムとキューにパーティションを設けることで, 充填率と待ち時間の改善を図るという手法は実環境の運用で既に導入されている. しかし, 現行の HPC システムの運用ではパーティションは動的に変更されるものではなく, 管理者が過去の運用情報を基に設定している.

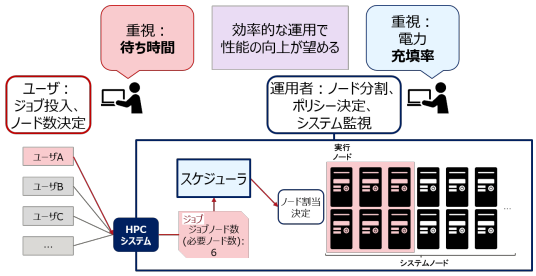

図 1 HPC システムの想定図

本研究では, 大規模環境におけるジョブスケジューリングの高充填率と短待ち時間の両立を実現するパーティション分割システムの基礎検討として, ジョブ分布, トポロジ条件を変えてジョブスケジューリングのシミュレーションを行う. そして, どのような条件でジョブスケジューリングの性能が変化するのか分析する.

1 お茶の水女子大学
2 富士通株式会社
3 株式会社富士通研究所

2. ジョブスケジューリング

HPC システムの運用において, コンピュータに投入される複数のジョブの起動や終了を制御, ジョブの状態を監視および報告をするソフトウェアとしてジョブバッチスケジューラが用いられる. スケジューリングアルゴリズムに基づいてジョブ処理の開始時刻, ノード割り当ての決定を行う. 適用されるスケジューリングアルゴリズムやネットワークトポロジによって待ち時間や充填率は変化する.

3. 関連研究

スケジューリングのコストと効率のコントロールを目的として, 時間軸方向をいくつかの区間に区切り, 区間単位でジョブスケジューリングを行う区間スケジューリングが提案されている [2]. この検証では, 充填率は直近の未来では短いスケジュール区間, 遠い未来には長いスケジュール区間を使った場合に最も高く, 待ち時間はスケジュール区間が長いほどノード数の少ないジョブの待ち時間が短くなることが確認されている.

また, HPC システムのジョブスケジューリングでは後述の Backfill が用いられることが多いが, その効率はジョブ実行時間の見積の正確さに依存する. 利用者のジョブ実行時間見積の正確性を示す指標を導入し, その指標に応じた優先度で Backfill する手法によって通常の Backfill と比較して最大で 6% の処理性能が向上したことが示されている [3].

以前の研究では, 異なるシステム規模, ジョブ分布条件でジョブスケジューリングを行い, 充填率と待ち時間の評価を行った [4]. その結果, ジョブの必要ノード数とその分布がジョブスケジューリングの充填率と待ち時間の違いに大きく影響を与えており, 効率的なシステム運用のためにはパーティションサイズの動的な変更が必要となることが明らかになった.

本研究ではジョブの必要ノード数の分布とネットワークトポロジに注目し, ジョブスケジューリングにおけるジョブ分布とトポロジ依存性の評価を行う.

4. 実験概要

大規模環境における最適なパーティション分割の決定に向けた基礎検討として, 様々なジョブ分布, ネットワークトポロジでジョブスケジューリングのシミュレーションを行い, 充填率と待ち時間の評価を行う.

ノードごとの性能差やネットワークによる不都合がない環境を前提とする. また, システムを構成するノードの扱いはいずれも平等とする. ジョブスケジューリングには Slurm Workload Manager(Slurm)[5] を用いる. Slurm はあらゆる規模の HPC システムで利用されているジョブスケジューリングシステムで, Sequoia(Lawrence Livermore National Laboratory) や Piz Daint(Swiss National Supercomputing Centre) のような TOP500 のスーパーコンピュータにも用いられている. パラメータ変更や新しいポリシー選択によるスケジューリングの影響の確認に数日から数週間かかる可能性がある. そこで, 本研究では Slurm Simulator[6] を用いる. ジョブデータには, 後述のジョブミックスを利用する. ジョブ密度はシステムノードに対する投入ジョブ規模の多寡の指標とし, 次式で求めた値とする.

$$Input_Ratio = \frac{\sum_{i=0}^{n} NodeJob_i \times Duration_i}{TotalSubmit \times SystemNode} \times 100$$

この時, NodeJob はそのジョブが必要とするノード数, Duration はジョブの実行時間, TotalSubmit は総投入時間, SystemNode はシステム全体のノード数を表す.

各ジョブ条件毎に 2 つジョブデータを生成し, 充填率と待ち時間の表には 2 つのデータを用いたスケジューリングシミュレーションの結果の平均を用いる. シミュレーションの共通の条件を表 1 に, 実験で用いたジョブ条件を表 2 記す. ジョブ条件の最小必要ノード数はシステムノードの 1/2048, 1/256, 1/32, 1/16, 1/8, 1/4 程度となるように定めた. 実際に生成された各条件のジョブ分布を図 2-8 に示す.

表 1 シミュレーションの共通条件

システムノード数	9216
ジョブの取得期間	7 日間
ジョブ密度	100%

表 2 実験に用いた各ジョブ条件

条件	最少必要ノード数	最大必要ノード数
1	1	9216
2	4	9216
3	36	9216
4	288	9216
5	576	9216
6	1152	9216
7	2304	9216

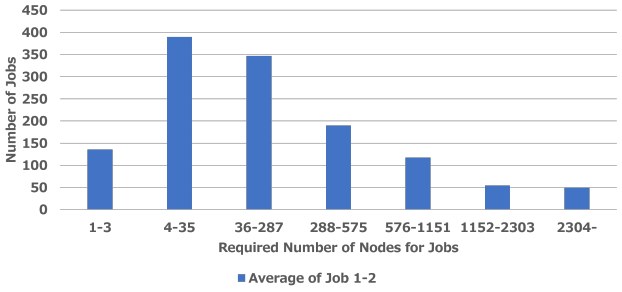

図 2 条件 1 のジョブ分布

トポロジには, tree, 3d torus, none の三つを比較に用いる.

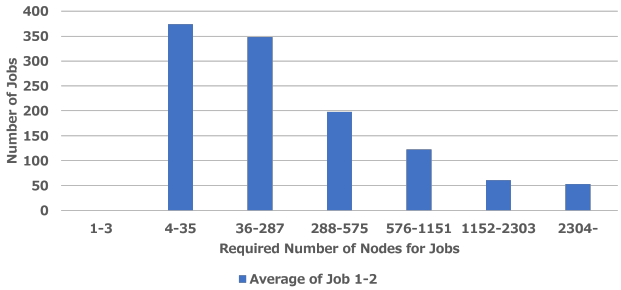

図 3　条件 2 のジョブ分布

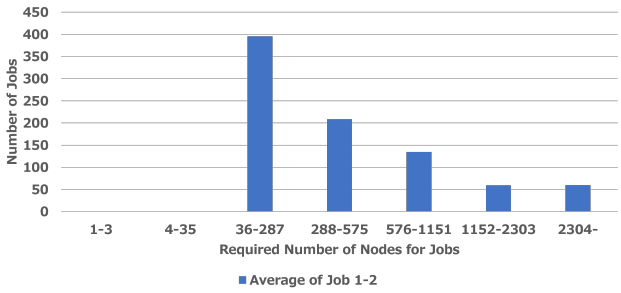

図 4　条件 3 のジョブ分布

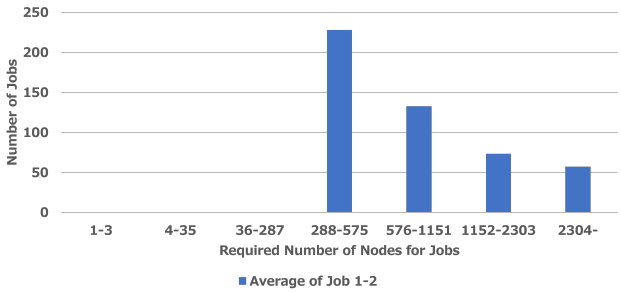

図 5　条件 4 のジョブ分布

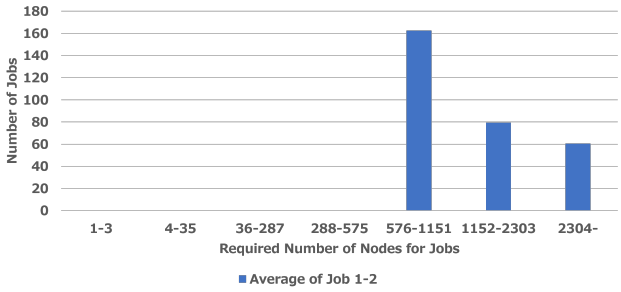

図 6　条件 5 のジョブ分布

4.1　ジョブミックス

ジョブミックス [2] とは，「京」ジョブ統計情報から最小必要ノード数と最大必要ノード数，そしてジョブ密度を設定し，ノード時間積がジョブ密度に達するまでジョブをランダムピックアップして生成されたジョブ群である．ジョブミックスの生成の際，「京」のジョブデータは必要ノード数，実行時間で分類されている．それぞれのグループの実際

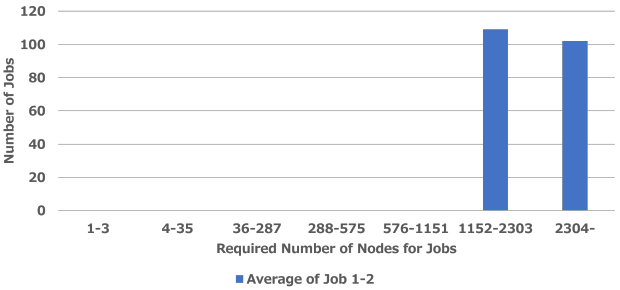

図 7　条件 6 のジョブ分布

図 8　条件 7 のジョブ分布

の実行時間やジョブ数，ジョブの投入頻度や分散といった統計情報を用いて，より実データに近い条件でジョブデータは生成される．

4.2　スケジューリングアルゴリズム

スケジューリングアルゴリズムの代表例として，First-Come-First-Served(FCFS) が挙げられる．FCFS では，ジョブは割り当て優先順位が到着順に定められ，処理が行われる．FCFS の利点としてはアルゴリズムがシンプルであること，ジョブ処理が公平であることが挙げられるが，ジョブの投入タイミングによっては充填率が著しく悪くなる可能性がある．

そのため，Backfill を有効にすることで，割り当て優先順が低い場合でも前方に実行可能領域があれば優先度の高いジョブを追い越して割り当てできるようにする．Backfill の導入の有無によるスケジューリングの違いを図 9 に記す．

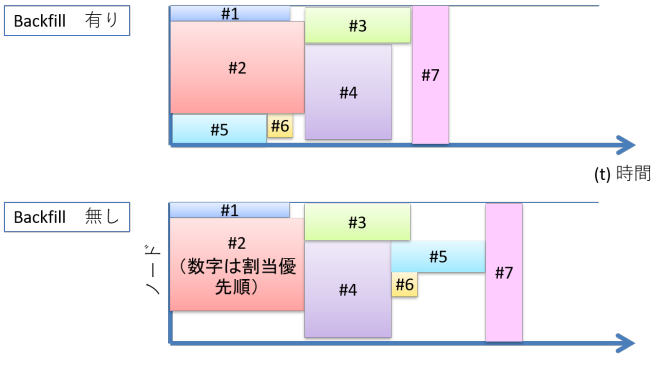

図 9　Backfill の有無によるスケジューリングの違い

4.3 ネットワークトポロジ

ネットワークトポロジはコンピュータやスイッチ等のノードの接続形態である．システムが大規模になるほど通信のボトルネックが及ぼす影響は大きくなる．そのため，HPC ネットワークの設計において，パフォーマンスに影響を与える主要な問題の 1 つとされる．本研究で検証したネットワークトポロジを以下に記す．

4.3.1 tree

tree はルートノードから枝分かれする様に伸びていく，階層型ネットワークと呼ばれるトポロジである．末端のノードが故障しても他のノード間通信には影響しないが，ルートノードが故障するとネットワーク全体の通信が障害を受けるため，ネットワークの障害耐性はルートノードに依存する．slurm では総ノード数以外に具体的な tree の構造を指定することができ，今回の実験では図 10 のような構造とした．深さは 3 とし，末端の枝を 36，それ以外の枝を 16 とした．

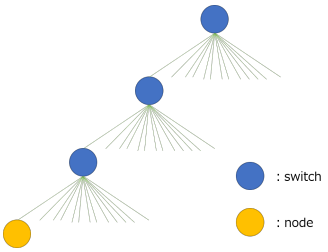

図 10　tree トポロジ

4.3.2 3d torus

3d torus は三次元トーラスの形状を利用したトポロジである．三次元トーラスは直方体において向かい合う面同士を合わせることで得られる．この決まりを一般化し，三次元配列にノードが配置されたメッシュ型トポロジとしている．図 11 は総ノード数 64 の三次元トーラスのトポロジの例である [7]．ノードは単純立方格子構造の一列に 4 つずつ配置されている．各ノードは隣接ノードの他に，計 6 つのノードと接続している．システム規模が大きくなるほど階層型

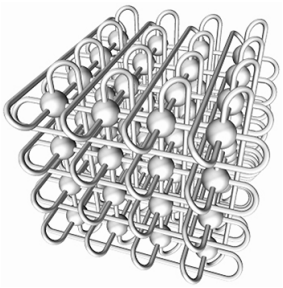

図 11　3d torus トポロジ

トポロジでは遅延しやすくなるが，トーラス相互接続はスイッチレスでありその構造から比較的高速，低遅延である．ただし，配線の複雑さが短所として挙げられる．slurm ではヒルベルト曲線を使用してノードを三次元空間から一次元空間にマッピングしている．

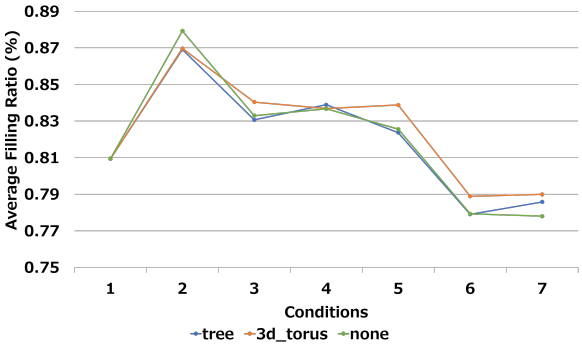

図 12　各トポロジの平均充填率

4.4 none

none は一次元トポロジである．大規模環境には不適だが，本研究では他トポロジとの比較のため検証した．

4.5 Slurm Workload Manager

Slurm Workload Manager[5] はあらゆる規模の HPC システムで利用されているジョブスケジューリングシステムである．あらゆるリソース管理プログラムと同様にジョブのスループット，システムの利用率，およびジョブの待ち時間に大きく影響する可能性のある多くのパラメータ設定が可能である．ただし，実験条件やポリシーの変更が実稼働 HPC システムにおけるスケジューラのパフォーマンスに与える影響を確認するために数週間要する場合がある．そこで，本研究では Slurm Simulator を用いる．

5. 実験結果

表 1，表 2 の条件下でのジョブスケジューリングシミュレーションの結果を記す．

5.1 充填率

各トポロジ，ジョブ条件でジョブスケジューリングを行った際の平均充填率を図 12 に示す．

全体の傾向としては，最小必要ノード数が大きいほど平均充填率は小さくなっており，ジョブ分布の影響がシステムノードへの割り当ての効率に表れている．ただし，条件 1 はその傾向に当てはまらない．条件 1 ではトポロジ間の差も小さく，小ジョブが大ジョブの割り当ての妨げとなって充填率が低くなっている可能性が考えられる．また，条件 5 以降ではトポロジ別充填率の差が大きく，ジョブ規模が大きいほど充填率に対するトポロジの影響が大きいと言える．

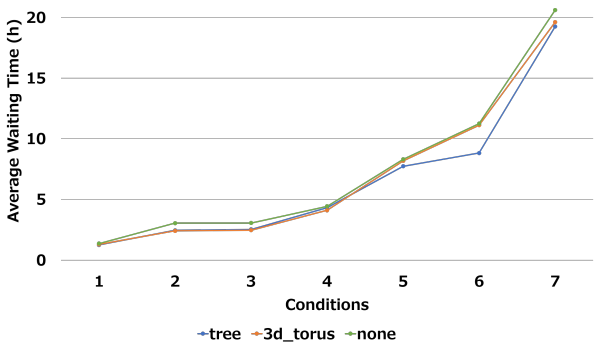

図 13　各トポロジの平均待ち時間

5.2　待ち時間

同様に, 平均待ち時間を図 13 に示す. 条件 1 から条件 5 までは平均待ち時間に相対的には大きな差はなく, いずれのトポロジでも条件 7 で平均待ち時間が大きく変化している. これは, ジョブの密度自体は他のジョブと同じく 100% だが, 大ジョブの割合が大きい条件だとキューが詰まりやすいためであると考えられる. また, 条件 6 以降はトポロジごとの平均待ち時間の差は大きくなっており, 必要ノード数の多いジョブが多数投入されるような環境ではトポロジの違いによって生じるジョブスケジューリングの性能の違いはより大きくなると考えられる. 充填率の結果も踏まえると, いずれのトポロジでも条件 6 がジョブスケジューリング性能の変化点であると考えられる.

次に, 必要ノード数でジョブを分類して各グループの待ち時間の傾向を見る. 分け方は表 3 に示す. 図 14-19 は各

表 3　必要ノード数によるジョブのグループ分類

グループ	下限	上限
1	1	3
2	4	287
3	288	575
4	576	1151
5	1152	2302
6	2304	-

グループの平均待ち時間である.

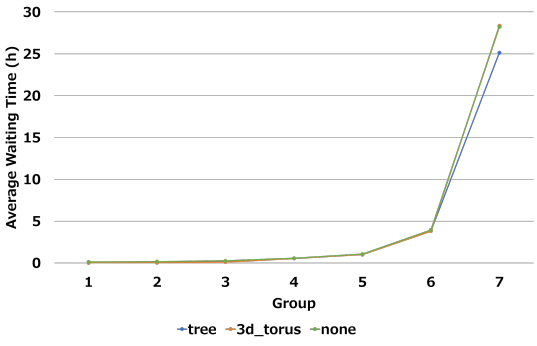

図 14　条件 1 の各グループの平均待ち時間

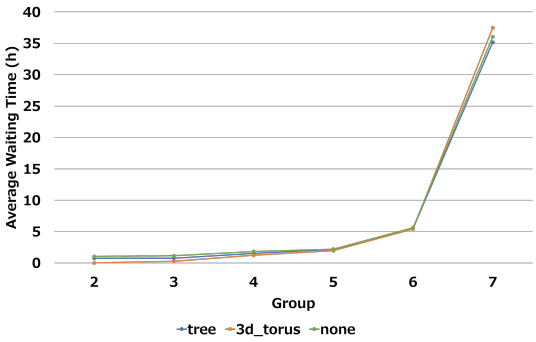

図 15　条件 2 の各グループの平均待ち時間

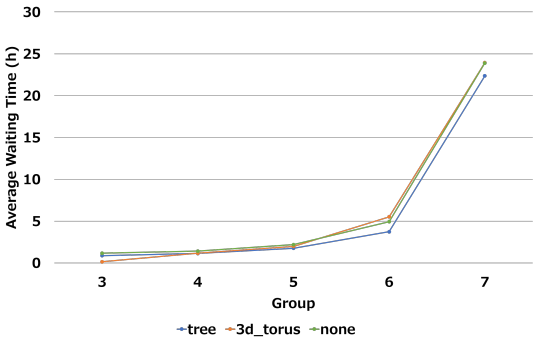

図 16　条件 3 の各グループの平均待ち時間

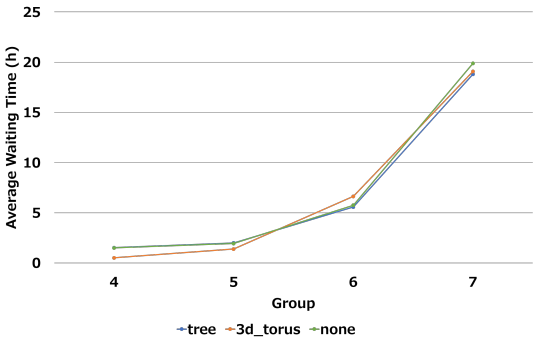

図 17　条件 4 の各グループの平均待ち時間

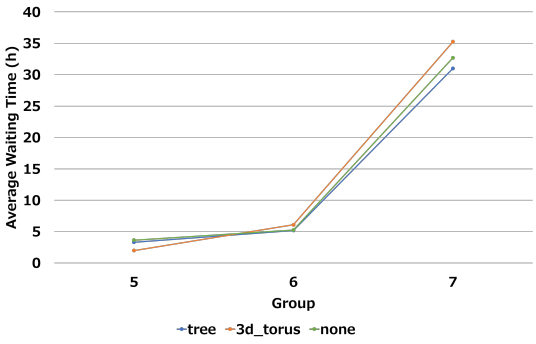

図 18　条件 5 の各グループの平均待ち時間

これらの結果を見ると, いずれの条件でもグループ 6 とグループ 7 の間に変化点があるように見られる.

以上の結果から, ジョブスケジューリングの性能変化にはジョブの最小必要ノード数が関係しており, 性能の変化点となるようなノード数が存在すると考えられる.

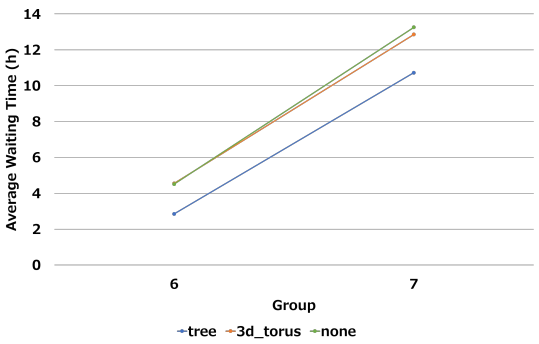

図 19 条件 6 の各グループの平均待ち時間

6. まとめと今後の予定

大規模環境におけるジョブスケジューリングの高充填率と短待ち時間の両立を実現するパーティション分割システムの基礎検討として, 必要ノード数分布, トポロジの条件を変えた際のジョブスケジューリングの評価を行った. シミュレーションの結果, 最小必要ノード数が大きいジョブ分布ではトポロジの違いによる充填率, 待ち時間の差の増加が確認された. また, 必要ノード数によってジョブを分類し, 各グループの平均待ち時間を分析した. その結果, グループ 6 とグループ 7 の間に変化点があることが分かった. これらの結果から, ジョブの特定の必要ノード数がジョブスケジューリング性能の変化点となっていると考えられる.

今後は, トポロジの違いとジョブ分布を踏まえたシステムパーティションの決定手法および動的なパーティション決定システムの構築を考えたい.

謝辞

本研究の一部はお茶の水女子大学と富士通研究所との共同研究契約に基づくものです. 本検討に際し, 「京」コンピュータのジョブミックスをご提供いただき, ご討論いただきました理化学研究所計算科学研究センターの宇野篤也氏に感謝致します.

参考文献

[1] 「京」の稼働状況. *https://www.r-ccs.riken.jp/jp/k/machine-status/*, Accessed: May 2020.
[2] 山本啓二, 宇野篤也, 関澤龍一, 若林大輔, 庄司文由ほか. 区間スケジューリングを用いたジョブスケジューリングの性能評価. 研究報告ハイパフォーマンスコンピューティング (HPC), Vol. 2014, No. 3, pp. 1–5, 2014.
[3] 滝澤真一朗, 高野了成ほか. 正確な実行時間指定を促すジョブスケジューリング. 研究報告ハイパフォーマンスコンピューティング (HPC), Vol. 2018, No. 2, pp. 1–9, 2018.
[4] 高山沙也加, 関澤龍一, 鈴木成人, 山本拓司, 小口正人. 投入ジョブ情報を踏まえたシステムノードのパーティション分割の考察. 第 12 回データ工学と情報マネジメントに関するフォーラム 論文集, Vol. 2020, pp. H4–4, 2020.
[5] Slurm Workload Manager. *https://slurm.schedmd.com/*, Accessed: May 2020.
[6] Nikolay A Simakov, Martins D Innus, Matthew D Jones, Robert L DeLeon, Joseph P White, Steven M Gallo, Abani K Patra, and Thomas R Furlani. A slurm simulator: Implementation and parametric analysis. In *International Workshop on Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems*, pp. 197–217. Springer, 2017.
[7] 次世代スーパーコンピュータの新システム構成を決定：富士通. *https://pr.fujitsu.com/jp/news/2009/07/17.html*, Accessed: May 2020.