正確な walltime 指定を促進し、TSUBAME の稼働率を高めるための消費ポイント計算式の変更について

各ジョブ実行により消費される TSUBAME ポイントの計算式が変更されます。現在は、実際の実行時間により計算されますが、さらにジョブ投入時に指定された walltime も加味されます。

- 実際は短時間で終わるが、指定 walltime が長いジョブ　⇒　消費ポイント増加
- 実際の実行時間に近い、正確な walltime を指定するジョブ　⇒　消費ポイント減少（最大で 20%減）

なお、最終的な課金額には、別途審議中の課金改訂案（9.1～22.8%の値上げ予定）も影響することにご注意ください。

TSUBAME2 の稼働開始以来、特に繁忙期の従量課金キューにおいて、ジョブ投入後、開始までの待ち時間が数日間になることが珍しくない。この問題に対処するために学術国際情報センターでは継続的な複数の対策を講じてきた（平成 23 年度の X キュー導入、平成 25 年 9 月の GPU 更新、現在議論中の U キューの導入など）。しかし近年、「見かけ上の稼働率に余裕があるにも関わらず、ジョブ開始が必要以上に待たされるケース」が頻発し、これに対処するには既存の対策だけでは不十分と判明した。

この現象の最大の要因は、「実態からかけ離れた、長すぎる実行時間(walltime 値)が投入時に指定されたジョブ」である。たとえば平成 24 年度には、実際の実行時間が 1 時間未満であるにも関わらず指定 walltime 値が 24 時間であるジョブが 69,470 個存在した。このようなジョブは、システムに一時的に利用できない計算資源を多く発生させ、システム稼働率を低下させてしまう（詳細については Appendix A を参照）。この問題はシステム全体に関わるものであり、ほぼ正確に walltime 値を指定しているジョブなどに対しても、結果的に悪影響を及ぼしてしまう。

ところが、現在の TSUBAME の消費ポイントの計算ルールでは

- 24 時間以下の指定 walltime 値までは係数が変わらない。
- 指定 walltime 値が実際の実行時間と乖離していてもペナルティがない。たとえば、「指定 walltime 値が 24 時間であり 1 時間で終了するジョブ」も「指定 walltime 値が 2 時間であり 1 時間で終了するジョブ」も、TSUBAME ポイントの消費量は同じである。

上の議論のように、指定 walltime 値を長くすること自体も資源を占有している、という観点に基づき、以下の 1 節・2 節に示すようなシステム改修を行う。この改修は、より正確な walltime を指定する利用者が優遇され、そのような利用者の増加により、システム全体の

資源有効利用と、利用者にとっての待ち時間の短縮をねらいとするものである。

1. 指定した walltime 値を考慮した消費 TSUBAME ポイント計算式の変更

現在のルールでは、各ジョブの実際の実行時間のみに比例して消費 TSUBAME ポイントを計算していたが、変更後は、ジョブ投入時に指定した walltime 値がより正確な値であるほど優遇されるような、新しい計算式を用いる。なおジョブスクリプトの単純な記述ミスなどによって、意図せずジョブ開始に失敗したジョブが極端な消費ポイント増加にならないよう考慮される。各ジョブ実行により消費される TSUBAME ポイントは以下の計算式により計算される。

変更の対象は従量制課金キュー(S 系、L 系、G、X)とする。並行して議論中の U キューが開設された場合には、それも対象とする。計算方法の異なる予約キュー及び定額利用キュー(V)に変更はない。

なお実施時期は、利用者への準備期間を設けるため、2014 年 4 月より新計算式によるポイント消費量を併記する試験運用を開始し、2014 年 8 月の中旬より実際に新計算式によるポイント消費を行う本運用を開始する。

旧計算式

node×実際の実行時間×que×pl×et

新計算式

1) 実際の実行時間 ＞300 秒の場合：

node×(0.7×実際の実行時間 + 0.1×指定した walltime)×que×pl×et

2) 実際の実行時間≦300 秒の場合：

ジョブ開始失敗と見なし、旧計算式を用いる

node	ノード数
que	バッチキューごとの係数
pl	プライオリティオプションの係数
et	時間延長オプションの係数

以下では、いくつかの典型例について、ジョブ実行に対する消費ポイントがどう変化するか例示する（node, que, pl, et は全て 1 と仮定）。

- 指定 walltime が 24 時間であり、1 時間で終了するジョブ：
  旧計算式：1 ポイント　⇒　新計算式：3.1 ポイント
- 指定 walltime が 3 時間であり、1 時間で終了するジョブ：
  旧計算式：1 ポイント　⇒　新計算式：1 ポイント（変化なし）

- 指定 walltime が 1.5 時間であり、1 時間で終了するジョブ：
  旧計算式：1 ポイント　⇒　新計算式：0.85 ポイント

このように、**指定 walltime が実際の実行時間に近いほうが優遇**される。なお、実際の料金がどのように変化するかは、並行して議論される TSUBAME ポイントの単価変更の議論も合わせて考慮されたい。

より一般的な議論のために、図 2 に、指定 walltime を 1 とした場合の、実際の実行時間の相対値と消費 TSUBAME ポイントとの関係を示す。指定した walltime が実際の walltime の 2 倍以下の場合に消費 TSUBAME ポイントは減少、2 倍以上の乖離がある場合に増加となる。

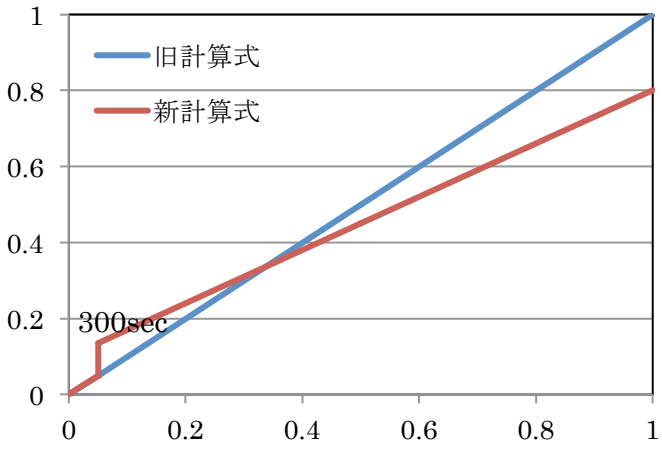

図 1　指定 walltime を 1 とした場合の実際の実行時間と消費 TSUBAME ポイントの関係

2.　時間延長オプション(1 時間)の新設

従量キュー(S 系、L 系、G、X キュー、並行して議論中の U キューが開設された場合には、それも対象とする)において時間延長オプションに et=0(1 時間まで、係数 0.9) を新設し、これをデフォルトの設定値とする。なお、新設に伴いこれまでのオプション値は 1 つずつ繰り上げとする。表 1、表 2 に新旧の時間延長オプションについて示す。この変更により、**指定 walltime が一時間以下のジョブが優遇**される。

なお、定額キュー(V キュー)においても et=0(1 時間)を新設し、これをデフォルトとするが、係数に変更はない。

表 1　旧係数表

オプション	係数	最大実行時間
0 (default)	1	24 時間
1	2	48 時間
2	4	96 時間

表 2　新係数表

オプション	係数(S 系、L 系、G、X)	係数(V)	最大実行時間
0 (default)	0.9	1	1 時間
1	1	1	24 時間
2	2	2	48 時間
3	4	4	96 時間

Appendix A：長すぎる指定 walltime を持つジョブが稼働率を下げる理由

TSUBAME のバッチキューにおいてジョブの実行順を決める際には、図 A-1(左)のような単純な早い者勝ちではなく、稼働率を高めるために、指定 walltime 値を利用した「バックフィル処理」と呼ばれる手法を用いている。バッチキュー内には様々な並列度のジョブが混在するため、早いもの順に（先に投入されたジョブを優先して）単純にスケジュール表を埋めるだけでは、表の中に空きが生じることは避けられない。バックフィル処理は、このような空きをできるだけ埋めるために（図 A-1（右））、以下の処理を行う。スケジュール表の中に空きができたときに、並列度および実行時間が空き部分に収まるような小規模ジョブを待ちキューのなかから見つけ、先に実行するようスケジュール表を書き換える（バックフィル）。これによりシステム全体の稼働率を向上させることをねらいとする。

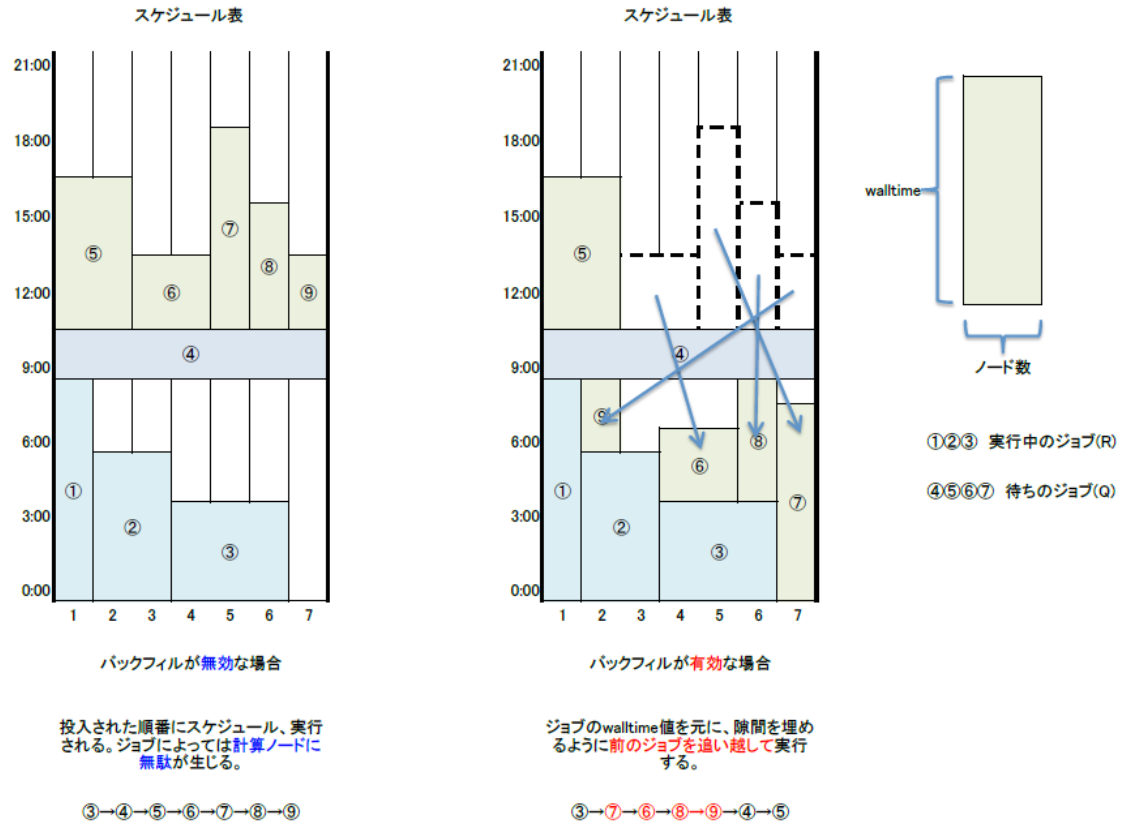

図 A-1: 単純なスケジュール手法(左)と、バックフィルを用いた手法(右)

ここでシステムが、バックフィル可能か否かの判定に用いる情報は（未来の予見ができない以上)、ジョブの実際の実行時間ではなくあくまで指定 walltime 値となる。しかしながらキューの中に、必要以上に指定 walltime 値が長すぎるジョブが多いと、図 A-2 のように、本来可能であったはずのバックフィルが働かない。つまり、このようなジョブの存在のために、利用不可能な資源を増やしてしまい、システム全体の稼働率を下げ、結果とし

てきちんと walltime を指定しているジョブへも悪影響を及ぼしてしまう。また、構造上最大 23 時間のジョブのみ走行可能な X キューにおいても同様の問題が発生し、その影響はより深刻となっている。

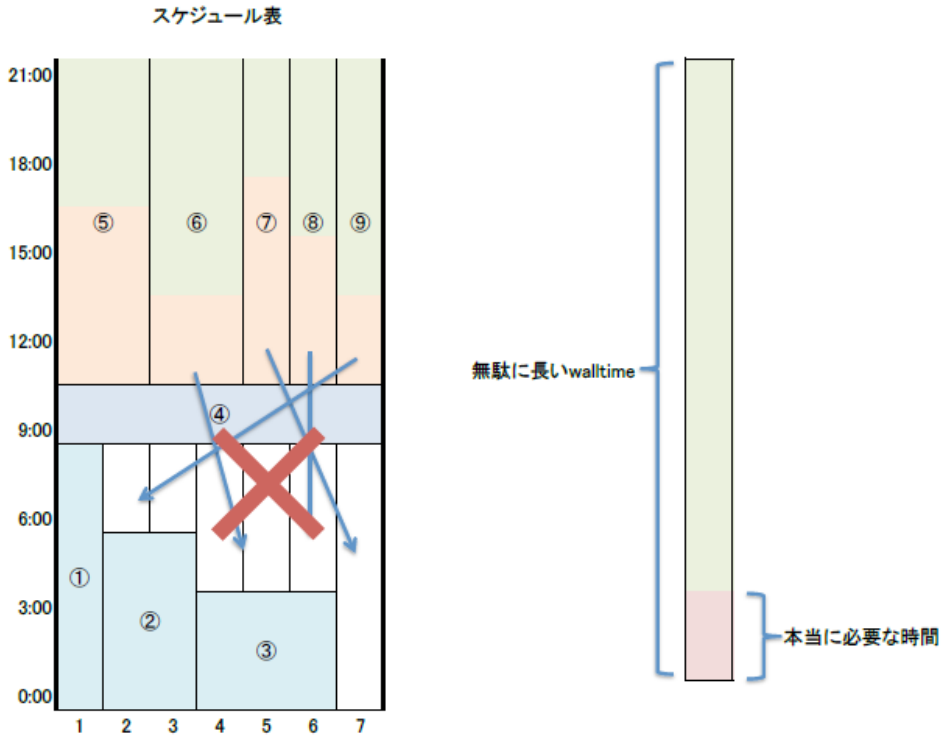

図 A-2：実際は短い実行時間で終了するが、長すぎる walltime が指定されたジョブが多いため、バックフィルができない様子