

仮想マシン環境におけるバッチ処理の性能予測手法の提案と評価 Performance evaluation method of batch jobs in virtual machine environments

馬場 輝幸[†] 田中 淳裕[‡]
Teruyuki Baba Atsuhiko Tanaka

1. はじめに

CPU 等のコンピュータリソースの有効利用やセキュリティの確保のために、仮想マシン(VM: Virtual Machine)の導入が進んでいる。仮想マシン環境では与えられたハードウェア条件に対して実行可能な仮想マシン数を見積もることが重要であり、これには割り当てるリソース量と得られる性能の関係を表す性能予測手法が有効である[1]。ところが、仮想マシン環境における従来の性能予測の多くは CPU 演算処理を対象にしており、ディスク処理を考慮した予測手法が無かった。この問題に対し、我々は仮想マシン環境でディスク処理のみを実行した際の仮想マシン性能予測について既に発表している[2]。本稿では、より一般的な処理を対象とするために、ディスク処理と CPU 処理が混在するバッチ処理に対して、処理性能を予測する手法を提案し、実測と比較評価する。

2. 対象システム

2.1 バッチ処理

本稿では、売り上げ集計等大量のデータをまとめて処理する際に利用されるバッチ処理を対象とする。バッチ処理とは、1つのジョブの終了を待って次のジョブが実行される処理のことである。クライアント/サーバシステムのようにジョブ要求がクライアント側のタイミングにより到着するトランザクション処理と区別するため、このような逐次的な処理を本稿ではバッチ処理と呼ぶ。

2.2 システム構成

対象とするシステム構成を図1に示す。仮想化ソフトウェア(VMM: Virtual Machine Monitor)により1台のコンピュータ上に複数の仮想マシン(VM)を構成する。各仮想マシン内で、CPUでの演算処理とディスクでの read 処理が混在するジョブをバッチ処理として実行し、その性能として平均スループット(単位時間あたりに処理される平均ジョブ数)を評価対象とする。また、仮想マシン間での独立性を保つために、仮想マシンごとに使用可能な CPU 使用率の上限を設定する。

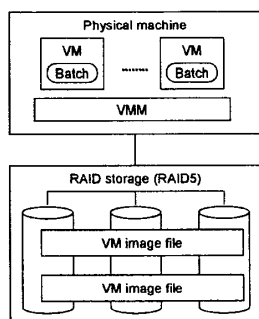


図1: 仮想マシンを用いたシステム構成例

3. 従来手法による性能予測

これまで、仮想マシンの性能予測手法としては、CPU 演算をトランザクショナルに実行する処理を対象にした予測手法が提案されている[3]。本章では、このトランザクショナル処理に対するモデルをバッチ処理に適応したモデルでの評価を行い、その問題点について説明する。図1に示したシステム構成は、待ち行列モデルで図2のように表せる。図2の待ち行列モデルにおいて、各仮想マシンは仮想 CPU を1つ占有し、ディスクは全仮想マシンで共有する。トランザクショナル処理を対象にした従来予測手法では、CPU 時間を100%利用可能な場合の単位時間あたりに処理可能なジョブ数が μ であり、仮想マシンに設定する CPU 使用率の上限が α ($0 < \alpha < 1$)であった場合に、仮想マシンが単位時間あたりに処理できるジョブ数を $\alpha\mu$ として待ち行列理論を用いて平均スループットを算出する。

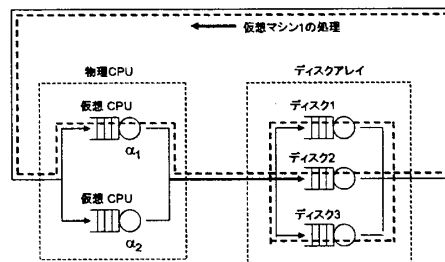


図2: バッチ処理を表す待ち行列モデル

3.1 従来手法による性能評価結果

従来手法での予測精度を評価するために、Xen 仮想マシンを用いて評価実験を行った。評価の実験構成は図1に示したシステム構成とした。物理マシンと RAID ディスクシステム間は 2Gbps の FC (Fibre Channel) で接続した。物理マシン及び仮想マシンのスペックを表1に示す。Xen はバージョン 3.2 を使用し、para-virtualization により仮想マシンを作成した。ドメイン U においてバッチ処理を実行し、その平均スループットを測定した。評価には、CPU 演算として乱数計算である rand 関数を用い、ディスク処理としては、キャッシュの影響を避けるためにランダム read を用いたテストプログラムを利用した。1ジョブあたりの rand 関数と read 関数の実行回数は指数分布に従っている。テスト負荷の平均 CPU 処理時間と平均ディスク処理時間は、CPU 処理時間=0.134 sec、ディスク処理時間=0.939 sec であった。この値は最大 CPU 使用率を 100% に設定した仮想マシンを1台構成し、この仮想マシン上で測定した値である。

従来予測手法の評価として、仮想マシンを1個構築し、この仮想マシンの CPU 使用率の上限値を変化させた際の平均スループットを測定した。CPU 使用率の上限を設定すると、仮想マシンは設定された割合の CPU 時間を占有し、他の仮想マシンが使用することは出来ない。測定結果を図3に示す。予測値は、最大 CPU 使用率を 100% に設

[†] 日本電気株式会社 システムプラットフォーム研究所
System Platforms Research Laboratories, NEC Corporation

[‡] 日本電気株式会社 研究企画部
Research Planning Division, NEC Corporation

定した場合の値をもとに計算しているため、図3の結果は横軸のCPU使用率上限が100%の点で予測と実測が一致している。しかし、CPU使用率上限が25%のときは、スループットにおいて、実測は予測を約50%も上回っており、予測誤差が大きく、性能予測として問題である。

表1: 評価に用いた物理マシン及び仮想マシンのスペック

	Dom 0	Dom U
CPU	3.2 GHz 4コア	
Disk	RAID5 (500GB x 3)	
仮想化ソフトウェア	Xen 3.2.0 (para-virtualization)	
Memory	1 GB	256 MB
OS (Kernel)	Fedora Core 6 (Linux 2.6.18)	Fedora Core 6 (Linux 2.6.18)

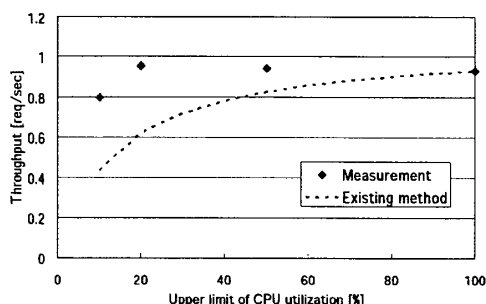


図3: 仮想マシンのCPU使用率上限とスループットの関係 (従来手法による予測値と実測値の比較)

3.2 従来手法での予測誤差の原因

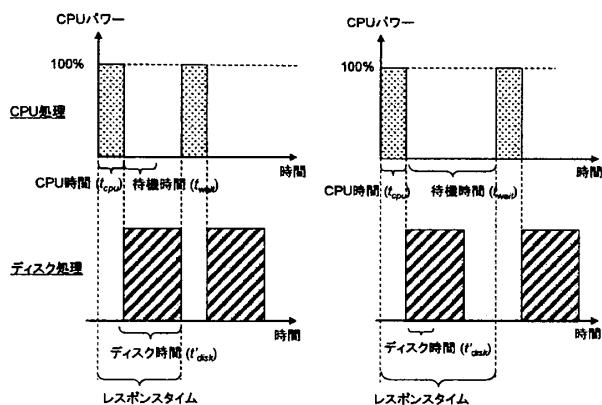
従来手法での予測誤差の原因を考察するために、実際の仮想マシンの動作と従来手法が想定した動作モデルを比較する。

はじめに、実際の仮想マシンの動作について説明する。実際の仮想マシンはCPUを時分割(タイムシェアリング)で共有しているため、図4に示すように仮想マシンは割り当てられた時間内において、CPUパワーを100%使用する。ただし、仮想マシンにCPU使用率の上限(α)が設定されている場合は、ある時間内(T)においてこの仮想マシンがCPU処理を実行できる時間は αT 内に制限される。このように仮想マシンに設定されたCPU使用率の上限を維持するために、仮想マシンがCPUを使用できない時間帯が挿入される。本稿では、CPU使用率上限維持のために挿入される時間を待機時間(t_{wait})と呼ぶ。複数の仮想マシンが存在する場合、この待機時間には他の仮想マシンのCPU処理が実行される。

待機時間がディスク処理時間より短い場合(図4(a))には、CPU処理とディスク処理が待機時間無しで繰り返し実行されても、ディスク処理中にCPUは未使用となるため、CPU使用率は設定された上限より小さくなる。このときの処理のレスポンスタイムは、CPU時間とディスク時間の和となり、CPU使用率の上限(α)に無依存になる。

また、待機時間がディスク処理時間より長い場合(図4(b))は、CPU使用率上限を維持するために挿入される待機時間内にディスク処理が完了する。このときのレスポンスタイムは待機時間の長さによるため、CPU使用率の上限(α)に反比例する。

次に、従来技術のモデルが想定するCPUの動作状況を図5に示す。ディスク処理時間はCPU処理能力に非依存であるため変化しない。しかし、CPU時間に関しては、従来予測モデルでは、仮想マシンに設定するCPU使用率上限が α ($0 < \alpha < 1$)の場合は、CPU処理能力が α 倍に低減されたCPUを想定する。このため、 α の値に依らず、CPUパワーを100%利用する場合と比較するとCPUでの処理時間は常に $1/\alpha$ 倍に増加する。このため、 α の値を小さくすると、レスポンスタイムを大きく、スループットを小さく見積もることになる。



(a) 待機時間が短い場合 (b) 待機時間が長い場合
図4: 仮想マシンでのCPU処理とディスク処理の時間

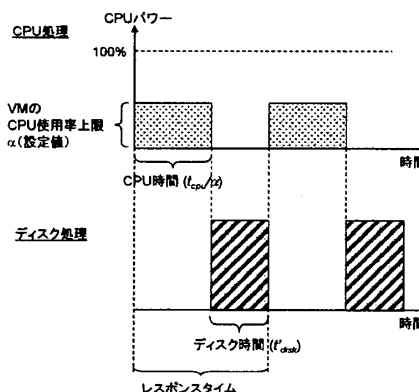


図5: 従来モデルが想定する仮想マシンの動作

4. 提案手法

3章で述べた従来手法での予測誤差の原因を考慮するために、図6に示すように、待機時間とディスク時間の大小関係により、CPU使用率上限を維持するための遅延を挿入するノードを追加した待ち行列モデルを提案する。提案手法の手順を図7に示し、以下で説明する。

- 1 CPU時間をもとに、CPU使用率上限設定を維持するために挿入される待機時間(t_{wait})を算出する。挿入される待機時間は、 $t_{wait} = t_{cpu}/\alpha - t_{cpu}$ で求まる。
- 2 複数仮想マシンがディスクを共有する影響を考慮したディスク時間を算出する。ここでは、従来方法で用いた図2に示す待ち行列モデルを使用する。従来方法と異なる点は、各仮想マシンでの処理時間として予め算出したCPU処理時間(t_{cpu})を使用する点である。従来方法では、各仮想マシンでの処理時間としてCPU使用率

の上限(α)を用いて t_{cpu}/α を使用していた。また、RAID構成のディスクでは、待ち行列計算においてディスク1台当たりの処理時間を使用する。このディスク1台当たりの処理時間は、RAID5を構成するディスク数が N の場合は t_{disk}/N で求まる。これらの処理時間を利用して、待ち行列ネットワークを畳み込み法[4]により解くことで、ディスク1台当たりの処理時間が算出する。そして、1リクエスト当たりのディスク処理時間(t'_{disk})は、ディスク1台当たりの処理時間をディスク数(N)倍した値として求める。

3 これまでに求めた CPU 使用率を維持するための待機時間(t_{wait})とディスク共有の影響を考慮したディスク処理時間(t'_{disk})の大小を比較する。

4(a) 待機時間(t_{wait})がディスク処理時間(t'_{disk})以下の場合 ($t_{wait} \leq t'_{disk}$)は、この仮想マシンの待機時間内にディスク処理が完了しないため、レスポンスタイムは $t_{cpu} + t'_{disk}$ となる。このレスポンスタイムの値からスループットは $1/(t_{cpu} + t'_{disk})$ 、CPU 使用率は $t_{cpu}/(t_{cpu} + t'_{disk})$ で求まる。

4(b) 待機時間(t_{wait})がディスク処理時間(t'_{disk})より大きい場合 ($t_{wait} > t'_{disk}$)は、この仮想マシンの待機時間内にディスク処理が完了するため、レスポンスタイムは $t'_{cpu} = t_{cpu} + t_{wait} = t_{cpu}/\alpha$ となる。このレスポンスタイムの値からスループットは αt_{cpu} 、CPU 使用率は α で求まる。

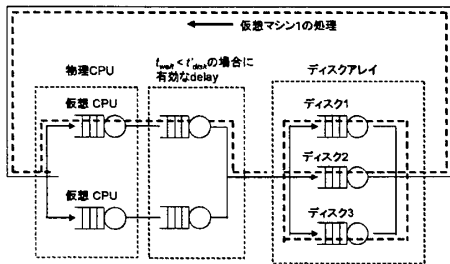


図6: 提案手法を表す待ち行列モデル

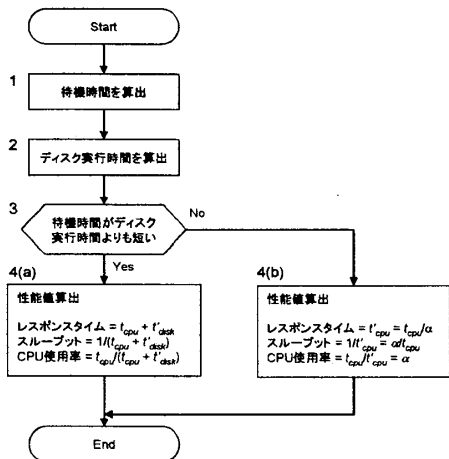


図7: 提案手法による仮想マシン性能予測の手順

5. 評価

評価のための実験構成は、3章での従来方法の評価構成と同じである。テスト負荷の平均 CPU 処理時間と平均ディスク処理時間は、CPU 処理時間=0.134 sec、ディスク処理時間=0.939 sec であった。はじめに仮想マシンを1台構

築し、CPU 使用率の上限を変化させてスループットを測定した。この測定結果を図8に示す。提案手法による予測値は、実測値とほぼ一致しており、6%以下の誤差である。また、仮想マシン数を変化させた場合の総スループットの予測値と測定結果を図9に示す。測定条件は、各 VM の CPU 使用率上限を 25% とし、各 VM 内でテスト負荷を実行した。複数 VM の場合も誤差 4% 以内でスループットが予測できる。評価実験から、提案手法を用いることで予測精度を改善できることが確認できた。

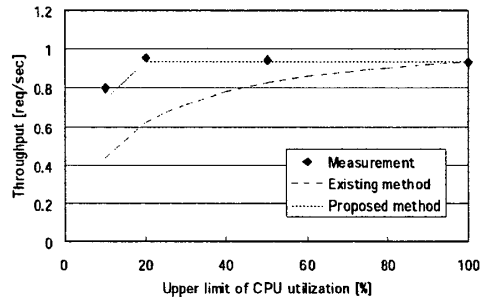


図8: CPU 使用率の上限に対するスループット

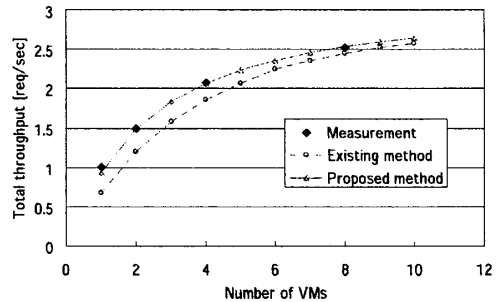


図9: 仮想マシン数に対するスループット

6. まとめ

本稿では、仮想マシン上で実行されるディスク処理を含むバッチ処理に対して、新たな性能予測手法を提案した。提案手法では、仮想マシンに予め設定される CPU 使用率上限を維持するために必要な待機時間とディスク時間を比較することで、実際の仮想マシンの動作に即した性能予測が可能となる。また、仮想マシンに Xen を用いた環境で提案手法を評価した。この評価結果から、仮想マシンの CPU 使用率上限、仮想マシン数の変化に対して本提案手法により誤差 6% 以内でスループットを予測できることを確認した。今後は、仮想マシンごとに負荷パターンや CPU 使用率上限値が異なる条件において評価する予定である。

参考文献

[1] Yiping Ding and Ethan Bolker, "How Many Guests Can You Serve? - On the Number of Partitions," in *Proceedings of CMG conference*, 2006.
 [2] Teruyuki Baba and Atsuhiko Tanaka, "Simple and practical disk performance evaluation method in virtual machine environments," in *Proceedings of SPECTS 2008*, pp. 344-351, 2008.
 [3] Anatoliy Rikun and Yiping Ding, "Optimization with Service Level Objectives in Virtual Environment," in *Proceedings of CMG conference*, 2006.
 [4] 亀田, 紀, 李 著「性能評価の基礎と応用」 pp.129-151, 共立出版, 1998.