B-008

仮想化におけるリアルタイム制御技術
RealTime in Virtualization Technology

金 成昊† 大島 訓†
Sungho Kim Satoshi Oshima

1. はじめに

近年、サーバ台数の増加による管理負荷の増大やサーバシステムのハードウェア性能の著しい向上より仮想化技術が注目されている。仮想化技術の導入により、１つの物理ハードウェアの上で複数のＯＳを動かすことができ、これにより既存サーバシステムの集約や計算機リソースの効率的な利用が可能である。一方、サーバ仮想化技術は企業におけるファイル共有サーバ、Web サーバ等の情報システムのみならず、電力監視制御システム等の社会インフラシステムへと適用範囲が拡大している。しかし、社会インフラの制御システムの適用においては、以下の課題が挙げられる。まず、サーバ集約による障害集中に対する信頼性問題である。従来システムでは１つのサーバ障害範囲がそのサーバにかぎる場合が多く、仮想化技術を適用し複数のサーバを集約した場合、物理サーバの障害はその物理サーバ上で運用する全仮想サーバへ影響を与えてしまう[1]。次に CPU、メモリ、ＩＯ等の計算機リソースを複数の仮想サーバで共有することによる性能の問題である。複数の仮想計算機で共有されるメモリ等の共有資源への競合による処理遅延が発生し、仮想サーバ上のプロセスのリアルタイム性の保証が困難なケースが発生する。本論文では、制御システムへ仮想化技術を適用する際に必要な各仮想サーバの性能保証問題において、仮想サーバ上の運用しているプロセスに優先度に応じて仮想サーバの実行優先度を調整する仮想サーバスケジューラを提案する。

2. 提案と実装

通常、仮想サーバのスケジューラにおいて Hypervisor が各仮想サーバの内部状態が把握できない。そのため、可能な限り仮想サーバ間の干渉を避ける方法で性能保証をする方針が考えられる。例えば、各仮想サーバに割り当てる資源を占有割当する方法が挙げられる。近年、マルチコア及び NUMA 等のハードウェアの高機能化によって低コストのＰＣサーバ計算機の普及により仮想サーバへ専用のコア、メモリノード及びＩＯデバイスを割り当てることが可能であるシステムが多くなっている。しかし、制御システムの場合、広域に配置されるサーバ群の集約には制限があるし提供先によっては高機能なサーバの適用が困難な場合もある。そのため、すべての計算機資源が占有割当可能とは限らないケースが多いのが制御システムの特徴でもある。このような問題の解決策として、仮想サーバの内部状態を反映したスケジューラ方式がある。[2]では、Hypervisor により各仮想サーバのランキューを精査することにより仮想サーバのスケジュール優先度を決定している。ただし、本方式ではすべての仮想サーバのランキュー精査する処理時間が集約度によって増大する。また、Hypervisor の処理オーバーヘッドが大きい。一方、[3]では、仮想サーバのスケジューラのタスクスイッチ処理を準仮想化しタスクスイッチが発生する度に、Hypervisor のスケジューラが介入される。Hypervisor は図１の方針によって仮想サーバの優先度を調整することによってシステム全体のプロセス優先度の調停を行っている。

P = [Pg ×(Pmax/Pgmax)]

P:仮想サーバ実行優先度
Pg:仮想サーバ上プロセス優先度
Pmax:仮想サーバ実行優先度のMax値
Pgmax:仮想サーバ上プロセス優先度最大値

図１ 優先度調整方針

しかし、本方式においては仮想サーバ上のすべてのタスクスイッチについて Hypervisor のスケジューラが介入することによるオーバーヘッドが大きい。本論文では、仮想サーバ上のスケジューラとの協調の仕方を変え、制御システムのプロセス特徴である処理の周期性を取り入れた仮想サーバのスケジュール方式を提案する。

2.1 仮想サーバスケジュール方式概要

[3]の方針のように、タスクスイッチ発生度に Hypervisor 処理介入を避けるために、まず仮想サーバ上のプロセスをリアルタイム優先度のプロセスと、非リアルタイム優先度プロセスに分ける。

制御システムにおいては、リアルタイム優先度のプロセスは複数の同優先度のプロセスから構成され、周期的に処理を行う。また、これらのプロセス間において順次または協調的に処理を行い一つの制御タスクを遂行する。その特徴に注目し、複数の同優先度のプロセス群を一つのリング状に管理する機構を仮想サーバに設ける。優先度によってリングの数は増えることになる。異なる優先度のリング間の資源共有を避け、排他による Priority Inversion 問題をさける。ただし、異なる優先度のリングが制御している資源へのアクセスが必要な場合は一時的に属するリングの変更を行う。ただし、リング変更は、該当リングに属するプロセスの１周期時間に対してのみ許可される。また、各リングには該当リング状のプロセス処理の開始と終了を示すプロセスが存在する。その開始及び終了プロセスには Hypervisor へ通知する hypercall を実装し、Hypervisor に対して仮想サーバ上の状況を知らせる。

2.2 システム概要と実装

提案方式の実装において、Hypervisor として Linux の標準仮想マシンモニタである KVM を用いる。アーキテクチ

† 株式会社 日立製作所 横浜研究所 Hitachi Ltd. Yohohama Research Laboratory

Copyright © 2012 by
The Instiute of Electronics, Information and Communication Engineers and
Information Processing Society of Japan All rights reserved.

ャは x86_64 を対象とし、仮想サーバのゲストＯＳとしては Linux を用いる。

2.2.1 仮想サーバ側の実装

提案手法の優先度リング機構はゲストＯＳのドライバとして実装する。Linux で hypercall のインフラである paravirt_ops[5]を利用することも可能であるが、カーネル本体の変更部位が大きくなることになる。ゲストＯＳの変更は仮想化のメリットを十分に引き出す選択肢ではないと判断し、独自モジュールとしてドライバの形で実装を行う。また、リング状のプロセスの開始終了を Hypervisor へ通知するために本ドライバは virtio[4]で実装する。本ドライバ提供するインタフェースは以下に示す。

（1）プロセス開始通知
g_sched_notify_start(ring_id, ring_prio)
ring_id：リングの識別子
ring_prio：仮想サーバ上のリング優先度また、ドライバは開始と通知用のインタフェースをそれぞれ実装する。

（2）プロセス終了通知
g_sched_notify_end(ring_id, ring_prio)

2.2.2 ＫＶＭ側の実装

仮想サーバ側に実装するドライバのインタフェースから、ホスト OS 上の KVM は、各仮想サーバ上に実行するリングの情報が取得できる。取得する情報から、sched_setscheduler により仮想サーバの実行優先度を調整する。その際の仮想サーバの識別は、仮想サーバ上の実装される virtio ドライバのメモリアドレス(vring)から行う。KVM のアーキテクチャ上、仮想サーバはホスト OS からは一つのユーザプロセスとして認識されるため、sched_setscheduler により実行優先度調整が可能である。

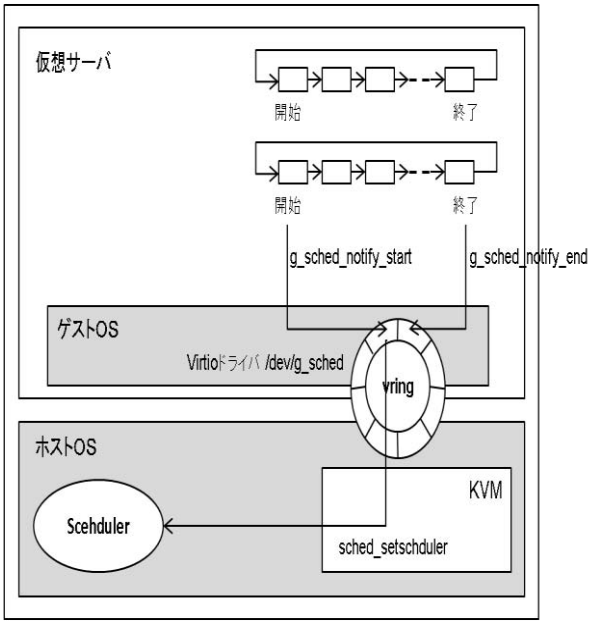

図 2　提案手法概要

2.3 評価

本システムを用いて、提案したスケジュール方式の評価を行った。１つの物理サーバで、複数の仮想サーバを運用し、Linux の資源割当サブシステムの cgroups を利用し、CPU、メモリ、ＩＯ帯域の割当を行った。ただし、CPU 資源割当においては、２つの仮想サーバを１つのコアに割当し提案手法の有効性を評価する。複数の仮想サーバを運用するケースにおいて仮想サーバの起動時には多量のＩＯが発生するため、CPU 負荷が高くなる。そのため、計算機リソースを共有している運用中の仮想サーバの処理性能に影響を及ぼす可能性極めて高い。このケースにおいて、起動直前の仮想サーバの実行優先度を他仮想サーバに比べて低くし、起動後制御プロセスの開始をもって該当仮想サーバの実行優先度を調整することにした。評価の結果、他仮想サーバの処理性能への影響を排除することができた。

3. 纏めと今後の予定

本論文では、仮想サーバのスケジューラと協調する Hypervisor の仮想サーバスケジュール方針を提案した。特に、制御システムのプロセスの特徴に注目し仮想サーバ上の制御プロセスを優先度毎にリング状に管理する機構を設ける。各リングでは処理開始と終了を Hypervisor に通知するために hypercall を virtio ドライバとして実装し、仮想サーバの内部状態を Hypervisor に知らせ、仮想サーバの実行優先度を制御する。また、外部割り込み処理のため、仮想サーバへ仮想割り込みをあげる前に仮想サーバの実行優先度を調整し、割り込み処理の性能を向上する。今後の課題として、制御システムの特徴を利用する仮想サーバの実行優先度調整の本方式の一般解の実現が挙げられる。実装において、仮想サーバの CFQ や RT グループスケジューラとの連携を考えている。また、外部割り込みに関する仮想サーバの優先度においては本論文では、[3]と同様な実装としているが、ゲストＯＳの変更を最小限することや処理性能改善のため、仮想サーバ上のリング機構との連携による実装方式を検討していく。

参考文献

[1] 金、カーネルトレース技術を用いた仮想化クラスタ技術の研究, 計算機アーキテクチャ, Vol.2011, No.5 (2011).
[2] Tadokoro, H., A Secure System-wide Process Scheduler across Virtual Machines, PRDC'10, pp.27-36 (2010)
[3] Jan Kiszka, Towards Linux as a Real Time Hypervisor, 11th Real Time Linux Workshop, pp.18-27(2009)
[4] Rusty Russel, vritio:Towards a de-facto statndard for virtual I/O devices, SIGOPS, Vol42, No.5, pp.95-103 (2008)
[5] Rusty Russel, lguest:Implementating the little Linux hypervisor, In Proceedings of Ottawa Linux Symposium (2007)

Copyright © 2012 by
The Instiute of Electronics, Information and Communication Engineers and
Information Processing Society of Japan All rights reserved.