モード遷移削減による仮想計算機の高速化

高橋由直† 新城靖†,‡ 佐藤聡† 中井央§ 板野肯三†,‡

筑波大学† 科学技術振興機構‡

1 はじめに

近年、Intel Virtualization Technology(VT) 等のハードウェアによる仮想化支援機能により CPU の仮想化は高速になっている。しかし入出力の性能に大きな問題がある。入出力性能の改善のために、仮想化計算機に特化した準仮想ドライバを開発するという手法がある。完全仮想化に基づく仮想計算機向けに提供されている準仮想ドライバには VMware Workstation のグラフィックスドライバ (Linux,FreeBSD,Solaris,Windows 用) や Novell 社の Xen のネットワークドライバ (Windows 用) があげられる。これらの準仮想ドライバは個別に手書きで開発が行われている。そのため開発コストが非常に多くかかり、小数のゲスト OS にしか提供されていない。そのため多くの OS が動作するという完全仮想化の利点を生かすことができない。

本論文では、既存のデバイスドライバを入力として完全仮想化に基づく仮想計算機用の準仮想ドライバを自動生成する手法を提案する。本提案手法では完全仮想化における特権命令の発行の際に生じるモード遷移を減らすことにより入出力を高速化する。本研究の対象は仮想計算機モニタ (Virtual Machine Monitor, VMM) として Xen, および Linux KVM、ゲスト OS として Linux, および FreeBSD、デバイスとしてディスク、およびネットワークである。

2 モード遷移について

Intel VT では VMX root モードと VMX non-root モードが存在する。VMM は VMX root モードで動作し、完全仮想化のゲスト OS は VMX non-root モードで動作する。VT を利用した完全仮想化の仮想計算機では VMX non-root モードで動作するゲスト OS が特権命令を発行しない限り、通常の物理ハードウェア上で動作しているときと違いはない。もしもゲスト OS が特権命令やセンシティブな非特権命令を発行するとゲスト OS が発行した命令を CPU がキャッチし、VMX におけるモードを non-root から root へ変更し、VMM に制御を移す。このモード遷移のことを VM Exit と呼ぶ。命令のエミュレーションが VMM によって行われた後、もう一度モードを non-root に移す。このモード遷移を VM Entry と呼ぶ。特権命令などが発行されるたびにこのモード遷移が行われオーバーヘッドになる。

本論文ではこの VMX におけるモード遷移によるオーバーヘッドを、特権命令をライブラリ呼び出しに変更することで軽減する手法を提案する。また、本論文では同じ手法を Xen における完全仮想化の仮想計算機で生じるドメイン切り替えを減らすことにも適用する。

3 準仮想ドライバの自動生成

準仮想ドライバの自動生成のために、本研究ではアセンブラプリプロセッサを用いる。本研究で提案する準仮想ドライバの自動生成の流れを図 1 に示す。通常はデバイスドライバのソースをプリプロセッサ、コンパイラ、アセンブラ、リンカの順に処理を行い実計算機用のバイナリを生成する。この通常の処理の流れを図 1 の破線の矢印で示す。本研究ではこの破線の流れの途中でアセンブラプリプロセッサを用いてモード遷移を減らしたコードに書き換える。この処理の流れを図 1 の実線の矢印で示す。このように本方式では同じデバイスドライバのソースコードからより高速な準仮想ドライバを生成することができる。

本方式の利点は、第 1 に準仮想ドライバを開発するコストを削減できる点にある。準仮想ドライバは 1 つの VMM について、対応するゲスト OS の数だけ開発が必要である。また、VMM 間の互換性がないため 1 つの OS でも VMM 毎に違うドライバの開発が必要になる。本方式では既存のデバイスドライバから準仮想ドライバを自動生成することで、これらの開発コストを削減することができる。

本方式の第 2 の利点として、アセンブラプリプロセッサを使用しているので既存のコードを扱いやすいことがあげられる。同様のことを C 言語のレベルで行うことも考えられるが、インラインアセンブラに対応できないという問題がある。また C 言語で入出力を行う関数の名前も OS ごとに異なっているという問題もある。アセンブリ言語レベルでの処理ではそのような問題は生じない。

3.1 命令の一括処理

本研究ではアセンブラプリプロセッサを用いてモード遷移の発生する命令の一括処理化を行う。一括処理化とは、本来 1 命令ごとに実行される複数の処理を一つの処理単位にまとめることである。

一括処理化の対象として、本研究では特に入出力命令に着目する。ソースコードがアセンブリ言語にコンパイルされた段階で output byte(OUTB) 命令が存在した場合、アセンブラプリプロセッサによって OUTB 命令を PUSH 命令と一括

Speeding up Virtual Machines by Reducing Mode Transition

Yoshinao TAKAHASHI, Yasushi SHINJO, Akira SATO, Hisashi NAKAI, Kouzo ITANO

† University of Tsukuba

‡ Japan Science and Technology Agency

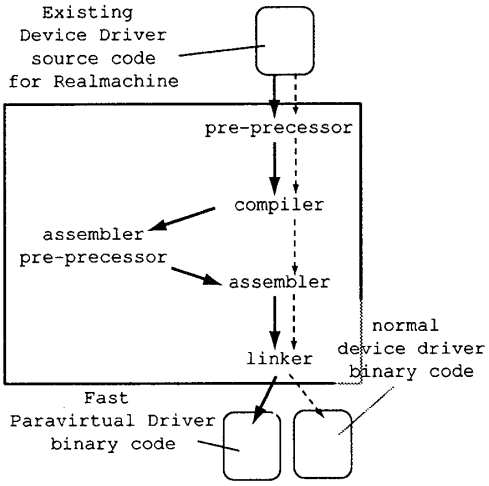

図1 準仮想ドライバ生成の流れ

処理化のためのサブルーチン呼び出しに変更する。このサブルーチンが実行されると、OUTB 命令の引数として利用されていた値をキューに保存する。このキューをバッチキューと呼ぶ。バッチキューに保存しておいた値を利用して、後で処理を一度に行う。input byte(INB) 命令も同様に入力用のバッチキューを用いて一括処理を行う。

一括処理を行った命令の実行タイミングとしては以下のようなものを検討している。

- ゲスト OS へのタイマ割り込みが発生した時
- hlt 命令 (計算機の待機のための命令) が実行された時
- 入力命令の実行において、入力用のバッチキューが空になった時
- バッチキューに溜まった出力命令の数がある閾値を超えた時

3.2 エミュレーションコードのゲスト OS への移動

入出力性能の向上のために、エミュレーションのためのライブラリをゲスト OS 内に挿入することでモード遷移を削減する手法を提案する。アセンブリプロセッサによってデバイスドライバにおける入出力命令をライブラリ呼び出しに書き換える。このライブラリでは命令のエミュレーションを行い、実際の入出力が必要な時にのみモード遷移を発生させ VMM 側で処理を行う。この結果、モード遷移の回数を手書きの準仮想ドライバと同程度まで削減することができる。

3.3 アセンブラプリプロセッサの実装

本論文ではアセンブラプリプロセッサをコンパイラ自動生成器 ANTLR(ANother Tool for Language Recognition) [1] を用いて開発する。ANTLR は LL(k) 構文解析を用いた構文解析器生成器である。ANTLR では EBNF に似た独自形式の文法ファイルから、字句解析器、構文解析器、木解析器などを生成することができる。

4 関連研究

Pre-virturalization [4] は、センシティブなアセンブリ命令の書き換えを行うアセンブラプリプロセッサを用いた仮想化手法である。Pre-virtualization は準仮想化の Xen や L4 などの準仮想化を対象としている。これに対して、本研究は VT を用いた完全仮想化を対象としている。

LilyVM は x86 アーキテクチャ向けの VMM である [3]。LilyVM ではアーキテクチャ依存部分をアセンブラプリプロセッサを用いる機械語命令の静的変換によってゲスト OS のその部分の移植を不要にしている。I/O 部分には手書きの準仮想ドライバを挿入する。本研究では LilyVM と同様にアセンブラプリプロセッサを用いることで移植作業を不要にしている。また、LilyVM と違い I/O 部分を対象とする。

VMware Workstation におけるネットワーク入出力高速化 [6] でも、入出力命令の一括処理を行っている。ただし、その手法では OS 内のデバイスドライバには変更を加えず、カーネルレベルで動作するモジュールにおいて入出力命令の内容を保存し、後にユーザレベルの VMM において一括化処理を行う。本研究では OS 内のデバイスドライバをアセンブラプリプロセッサによって変更する。また、本研究では一括化はゲスト OS 内で行う。

5 まとめと今後の課題

本論文では完全仮想化の仮想計算機のための高速なデバイスドライバの自動生成手法を提案した。本手法では通常は特権命令として実行される命令を、命令のエミュレーションを行うライブラリ呼び出しに書き換えることで VMX non-root モードから VMX root モードへのモード遷移を削減する。ライブラリ呼び出しへの書き換えは、ANTLR を用いて作成したアセンブラプリプロセッサを用いて行う。

今後はエミュレーションのためのライブラリの実装と、ANTLR を用いたアセンブラプリプロセッサの実装を行う。実装後は自動生成したデバイスドライバの性能評価を行う。

参考文献

[1] *ANTLR Parser Generator*. http://www.antlr.org/, 3rd edition, Novenber 2007.

[2] Avi Kivity, Yaniv Kamay, Dor Laor, Uri Lublin, and Anthony Liguori. kvm: the linux virtual machine monitor. In *Proceedings of the Linux Symposium*, pp. 225–230, Ottawa, Ontario, 2007.

[3] Hideki Eiraku and Yasushi Shinjo. Running BSD kernels as user processes by partial emulation and rewriting of machine instructions. In *BSDCon*, pp. 91–102, 2003.

[4] Joshua LeVasseur, Volkmar Uhlig, Matthew Chapman, Peter Chubb, Ben Leslie, and Gernot Heiser. Pre-virtualization: soft layering for virtual machines. Technical Report 2006-15, Fakultät für Informatik, Universität Karlsruhe (TH), July 2006.

[5] Paul Barham, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Rolf Neugebauery, Ian Pratt, Andrew Warfield. Xen and the art of virtualization. *SOSP 2003*, pp. 19–22, Oct 2003.

[6] Jeremy Sugerman, Ganesh Venkitachalam, and Beng-Hong Lim. Virtualizing I/O Devices on VMware Workstation's Hosted Virtual Machine Monitor. In *Proceedings of the General Track: 2002 USENIX Annual Technical Conference*, pp. 1–14, Berkeley, CA, USA, 2001. USENIX Association.