z/VM仮想計算機におけるDCSSを用いたLinux間メモリ共有

井口 寧[†] 佐藤幸紀[†] 上埜元嗣[†]
宮下夏苗[†] 芝崎丈男[††] 北沢強[††]

計算機システムが多様なサービスに用いられるようになる中で，メインフレームのz/VMやVMwareなどの仮想計算機システムが注目されている．仮想計算機システムでは，複数のソフトウェア環境が全く独立に動作するため，各仮想計算機間の独立性は高い反面，共通で使用するコード領域なども独立してメモリ上に置かれるため，メモリ資源の利用効率が十分でないという問題点がある．

そこで本論文では，メインフレームのz/VMハイパーバイザで利用可能なDCSSを用いて，仮想計算機間でメモリを共有することによってシステム全体のメモリ使用効率を向上させた場合のメモリ共有効率について評価した．本来仮想計算機間メモリ共有を考慮されていないLinuxでは，カーネルのメモリ共有による節減は約2.7%と効果は少なかったが，DCSSファイルシステムを用いることによって，主要なdaemonについてもRead Onlyメモリ領域を仮想計算機間で共有することができ，Linux全体では約25%の節減が可能であることが分かった．また，Open Officeなどのアプリケーションでは，最大で40%程度の節減が可能であることが明かとなった．

Inter-Linux memory sharing among virtual machines using DCSS on z/VM

Yasushi Inoguchi,[†] Yukinori Sato,[†] Kanae Miyashita,[†]
Mototsugu Ueno,[†] Takeo Shibazaki[††] and Tsuyoshi Kitazawa[††]

Virtual machine such as z/VM on a mainframe system and VMware are researched and developed for various services. Although a virtual machine offers highly separated environment, efficiency of memory resource utilization isn't good because text area of commonly used program among virtual machines are stored on memory independently.

This paper addresses evaluation of efficiency of memory utilization using DCSS, which is a memory sharing scheme available on z/VM hypervisor on a mainframe system. Memory sharing among Linux kernel reduced only 2.7% memory utilization, because Linux kernel isn't optimized for memory sharing. On the other hand, Linux must executes many daemons and sharing daemon's text and read only area reduced almost 25% memory utilization. It is also shown that maximum memory reduction of user applications such as Open Office reaches almost 40%.

1. はじめに

計算機システムが多様なサービスに用いられるようになる中で，複数のOS(Operating System)を同時に一台の物理サーバの上で稼働させる仮想化技術が注目されている．仮想化技術は，OSよりもハードウェアに近いベース・ソフトウェアによって，ハードウェア資源を仮想化し，複数の仮想計算機(VM; Virtual Machine)として提供する技術であり[1)~4)]，それぞれのVM上で各種のOS(ゲストOS)を稼働させることができる．代表的な実装例としては，メインフレームのz/VM[4)]やVMware[5)]などがある．物理的には一つのサーバを複数のVMで共有できるので，ハードウェア資源の効率的な利用が可能となると同時に，VMごとに完全に独立した計算機環境を構築できる．

VMによる主なサービス提供上の利点は，複数のサービスを1台のサーバ上に統合する際に，セキュリティや管理の独立性を提供することが可能な点である[6)~8)]．

仮想化技術のシステム的なメリットは，(1)VM機構を提供するソフトウェア(VMハイパーバイザ)の構造がOSに比べて極めて単純なため，VMハイパー

† 北陸先端科学技術大学院大学
Japan Advanced Institute of Science and Technology.
†† (株) 日本アイ・ビー・エム
IBM Japan

バイザそのものが障害となって複数のOSによるサービス全体が異常停止する可能性が少ないこと，(2)OSによるプロセス間の分離に比べて格段に高いセキュリティが確保できること，などがある．また，(3)VMごとに違った運用ポリシーやパッチの適用を行ないやすいことや，(4) 通常ならば複数のOSに対してCPU，メモリ，ディスクから成る物理的なサーバを複数台用意すべきところを，1台のシステムに集約できるため，管理コストの削減や低消費電力化 (CO_2 排出量の低減化) が可能となること，なども有用な利点である．

一方，システムの効率的利用という観点からは，あるVMから別のVM上のメモリ空間は全く関知できないため，メモリの利用効率が低下する問題がある．OSによる同一プログラムの複数同時実行であれば，OSがプログラムのテキストページをプロセス間で共有させ，RO(Read Only) ページの利用効率を向上させることができる．しかしながら，同一サーバ上の異なるVMで同じプログラムが実行される場合には，互いのメモリ空間は全く関知できないため，VMごとにプログラムのバイナリがロードされる結果となり，OSによるプロセス共有よりもメモリ利用効率が低下する．

また，現在の計算機システムはマルチコア化が発展し，CPU処理能力は向上している反面，メモリの搭載量には物理的に限りがある．極端に多くのメモリを搭載することは，メモリスロットの増加を意味し，その結果メモリのアクセス速度の低下につながるため，システム全体での使用メモリ量を削減することは，特に多くのVMが動作する場合には重要性が高い．

そこで本論文では，IBM z/VM[4),9),10)] が提供するVM間のメモリ共有機構DCSS (DisContinuous Saved Segment) を用いて，従来広く使われているLinuxでの物理メモリ使用量の節減を試み，その効果を評価した．OSカーネル部分のテキスト共有によるメモリ節減は古くから実装されているが[4)]，VMを前堤としていないLinuxでのメモリ共有の試みは殆んど行なわれていない．また，Linuxでは多くのdaemonが動作しているが，これらについてのメモリ共有による物理メモリ使用量の節減効果は明白ではない．

本論文の貢献は，メモリ共有を前堤としないLinuxのカーネルについてVM間メモリ共有による物理メモリ使用量の節減効果を定量的に明らかにしたこと，Linuxの動作に必要なdaemon類の実行イメージを共有し，Linuxシステムとしてのメモリ使用効率を高めたこと，およびそのメモリ利用効率についてサーバ用途とクライアント用途の代表的なアプリケーションについて評価し，効率を定量的に明かにしたことである．

本論文の構成は以下の通りである．第2節で仮想計算機の概略とDCSS，及び本論文で実装したDCSSファイルシステムによるVM間メモリ共有について解説する．第3節でLinux環境におけるOSとアプリケーションを用いた物理メモリ使用量節減の定量的評価を示す．第4節では本手法をz/VM以外のシステムに適用する場合の効果について考察する．第5節では関連研究について述べる．第6節はまとめである．

2. DCSSを用いたメモリの共有

2.1 仮想計算機システム

仮想計算機 (VM) システムは，システムの持つ資源を仮想的にOSに提供することによって，通常1システムあたり1つのOSが実行されるところを，1システムを複数のOSで共有する機構であり，近年のセキュリティ需要の高まりやシステムの利用効率化の観点から注目されている．技術自体は古くからあり，代表的な実装例としては，IBM System z[9)] 上のz/VM[4),11)] やVMware[5)] などがある．いずれの方式も，VM間は完全に分離され，互いのメモリ空間にアクセスすることは，原則的には不可能である．

2.2 DCSS

DCSS (DisContinuous Saved Segment) は，z/VMが提供するVM間でのメモリ共有の機構であり，パフォーマンスデータの収集等で利用されてきた．z/VM上で稼働するLinux OSからもdcss blockデバイスドライバを使用してアクセス可能であり，ゲストOSとして稼働する複数のLinuxから同一のメモリ領域がアクセスできる．

図1にDCSSを用いたVM間メモリの割り当て状況の例を示す．通常，VMのメモリ空間 (Virtual Memory) は完全に分離されており，物理メモリ上では異なる領域が割り当てられるのに対し，DCSSは，複数のVMのメモリ空間から参照可能であり，これは物理メモリ (Physical Memory) 上で1つのメモリ空間としてマップされる．各VMのメモリ空間は，DCSSのメモリ空間と固有のメモリ空間が結合されて提供される．DCSSは1つのVM内で複数設定でき，名前を付加することによってセグメントを管理する．

当然ながら，複数のVMで共有されるDCSSは，同時に唯一つのVMがDCSSへの書き込みを行なう場合を除き，各VMで稼働するLinuxにROモードのメモリ領域として割り当てる必要がある．

2.3 NSS

NSS(Named Saved System) は特殊なDCSSであり，仮想アドレスが0番地から始まる．アドレスが0

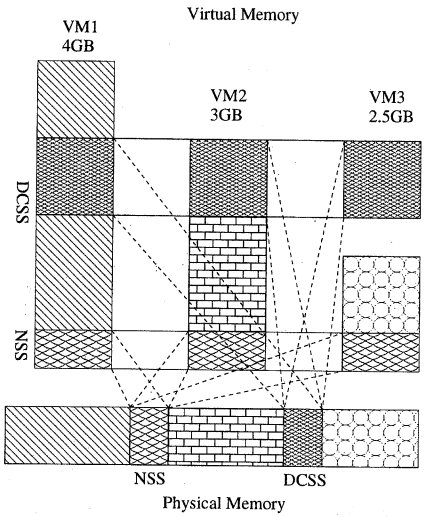

図 1 仮想メモリ空間と DCSS 領域
Fig. 1 Virtual memory space and DCSS

番地から始まるため，OS のカーネルが格納される領域が VM 間で共有される．

2.4 z/VM と仮想記憶

VM ハイパーバイザ環境下では，仮想記憶の管理に VM レベルと OS レベルの 2 つの機構が関与する．OS レベルでは長時間使用していないページをディスクにページアウトするが，VM もまた使用していないページをページアウトする．仮想計算機システムにおいて，ゲスト OS と VM が連携を取らず勝手にページングを行なう問題をダブルページングと呼び，次の様な弊害がある．

(1) ゲスト OS が使用しないページと判定したのに，VM がページを物理メモリに置くことによるメモリ利用効率の低下，

(2) ゲスト OS が必要とするため物理メモリ上にページを置きたいのに，VM がページアウトすることによるシステムの不安定化．

この問題について，本稿で取り扱う z/VM 上の Linux では，VM ハイパーバイザとゲスト OS である Linux が互いに連携をしながら，矛盾が生じないような機構を有している[12]．DCSS においても，必要が無い場合には z/VM によってページアウトされ，物理メモリを浪費しない機構であることに注意されたい．

2.5 DCSS ファイルシステム

z/VM における Linux での DCSS の利用法として特殊なものに，ファイルシステムを DCSS に設定することが可能である点があげられる．この様子を図 2 に示す．DCSS にファイルを格納することにより，あたかも RAM disk のように利用することができるが，書き込み権限を持つ VM 以外からは RO としてアクセスされる．本稿では，DCSS 上に設けられたファイルシステムを DCSS ファイルシステムと呼ぶ．

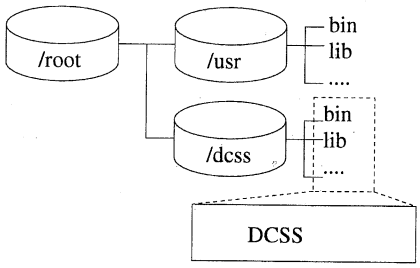

図 2 DCSS ファイルシステム
Fig. 2 DCSS file system

本論文で構築した DCSS ファイルシステムの設定手順を，以下に例と共に示す．

(1) VM ユーザ上で DCSS を作成
defseg jaistseg 20000-3ffff sr
saveseg jaistseg

(2) DCSS のメモリをオーバーラップしないように VM ユーザのメモリ定義を変更
define storage 512M

(3) Linux ブート後，dcss block ドライバをロードし，sysfs を使用して DCSS を認識させる．
modprobe dcssblk
echo jaistseg 〉 /sys/devices/dcssblk/add

(4) /proc/devices でメジャー番号を確認し，mknod コマンドで/dev/dcssblk0 デバイスノードを作成する．

(5) mke2fs コマンドで ext2 でファイルシステムを作成する．

DCSS ファイルシステムに置かれたファイルは，現実には VM が管理するメモリに格納される．ゲスト OS である Linux が DCSS に格納されたファイルを読み込む動作は，実際にはファイルハンドルのポインタを mmap 関数によって DCSS の該当メモリアドレスに設定することを意味するので，z/VM では RAM disk 領域から読み込むためのメモリコピーなどの動作は，特別な操作無しに標準として不要である．

DCSS ファイルシステムに格納されている実行形式ファイルを実行する場合，CPU は単純に DCSS に置かれた実行形式ファイルの実行開始アドレスにジャンプするのみであり，やはりメモリコピーやページ キャッシュなどの効率を低下させる動作は行なわれない．プログラムの実行にあたり，R/W(Read/Write) メモリ

領域は随時 malloc 等で確保されるが，これらの R/W 領域は DCSS とは別の，個々の VM がローカルに持つメモリ領域から確保される．

また，DCSS の中のアクセスされないページは，VM のメモリ管理機構によってページアウトされるので，不必要に物理メモリを占有することは無い．ファイルのアクセスがあれば，VM がページをロードするが，このページはどのみちゲスト OS からアクセスされるので，ページインは 1 回で済む．

これらの点は DCSS ファイルシステムが単なる RAM disk とは異なる，優れた点である．今回定量的な評価は行っていないが，バイナリファイルではアプリケーションの起動時間が大幅に短縮できる．一方で Shell Script のようなテキストファイルを読み込んでインタープリタが実行する様な場合は，ロード時間の削減効果は少ない．また，DCSS ファイルシステムは複数の VM で共有することが一般的なので，ディスクスペースの節減も利点の一つである．

2.6 DCSS ファイルシステムを用いた VM 間メモリ共有

複数の VM で多用されるアプリケーションを DCSS ファイルシステムに格納することによって，(1) バイナリの物理メモリ上での共有によるメモリ利用効率の向上，および (2) メモリからのゼロコピー読み出しによるアプリケーションの起動・実行の高速化，の 2 点が期待できる．DCSS ファイルシステムに置かれたファイルを参照する場合，ファイルへのポインタを VM 間共有メモリに置かれたファイルの先頭番地に付け替えることで行なわれる．従って多くの VM が参照するファイルであれば，通常複数の VM が個別にファイルを読み込み物理メモリに格納されるところが，1 つの物理メモリの領域を共通して参照できるため，メモリの利用効率を向上できる．

具体的に DCSS ファイルシステムに格納するファイルごとのメモリ利用効率の向上について，次節で詳しく評価する．

3. DCSS を用いた物理メモリ使用量の評価

3.1 Linux の OS 部分でのメモリ共有効率

OS は各 VM で必ず起動されるため，VM 間メモリ共有の効果が高く期待できる．このため，CMS など z/VM を前堤として開発されたオペレーティングシステムでは，カーネルの主要なメモリが極力 RO モードとなるように設計されており，VM 間メモリ共有によるメモリ使用効率が高い．一方，Linux は設計時点で VM を想定しておらず，共有できる RO メモリ領域は必ずしも多くはない．

表 1 Linux の OS 部分でのメモリ共有効率
Table 1 Memory Sharing Efficiency of Linux-OS

	Method	WSS (MB)	Reduce (MB)	Efficiency (%)
Linux	DASD	236	-	-
	NSS	230	6	2.7
	DCSS	183	53	22.5
	NSS+DCSS	177	60	25.2
CMS	DASD	11	-	-
	NSS	6	4	40.6

そこで本節では，Linux カーネルのみのメモリ共有，DCSS ファイルシステムのみを用いた OS コマンドの実行イメージによるメモリ共有，更に両者のメモリ共有を行なった場合について，メモリ使用効率について評価実験を行なった．

実験条件として，用いた計算機システムは IBM System z 990, 1 LPAR, VM ハイパーバイザは z/VM 5.2, ゲスト Linux は z/VM 用の Linux であり，Novel SuSE Enterprise Server 10 SP1 (Kernel 2.6.16, developerWorks パッチ)．測定は，z/VM Performance Toolkit にて WSS (Working Storage Set) として示されるページ数からメモリ使用量を算出した．WSS は物理メモリ上に置かれているページ数に相当し，z/VM では 1 ページは 4kB，これから各 VM(=Linux) ローカルの (共有部分を含まない) 物理メモリの消費量を知ることができる．本評価では 1 台の System z 上にゲスト OS である Linux を複数起動し，安定した状態で 1 Linux あたりの物理メモリ消費量を測定した．

3.1.1 NSS による Linux カーネルの共有

表 1 に Linux の OS 部分でのメモリ共有効率を示す．WSS は先に述べた通り物理メモリの使用量を示す．Reduce は DASD(Direct Access Storage Device, 通常は HDD に相当) と比べた際の物理メモリの節減量であり，Efficency は DCSS や NSS 無しの場合に対する節減割合である．

表中，DASD は VM 間メモリの共有を一切しない方法である．通常の DASD に Linux システムを格納し，ここから Linux を起動した場合の物理メモリ消費を示している．NSS は，仮想アドレス 0 番地から始まる NSS のみがメモリ共有領域とした場合のメモリ消費である．通常この領域には Linux カーネルが置かれるため，OS 付属の各コマンドや daemon 類を含まない，Linux カーネルのみのメモリ共有効率を知ることができる．表によれば，Linux カーネルのメモリ使用

量は約 236MB であり，NSS によってカーネルのメモリ消費量は約 230MB に縮減し，割合として 2.7%節減できた．

3.1.2 DCSS ファイルシステムによる Linux コマンド共有

Linux はマルチタスク OS であるため，OS 起動後にカーネルばかりでなく，/etc/init.d/配下の各種 OS コマンドや daemon が実行される．これらの実行ファイルを DCSS ファイルシステムに格納することによって，バイナリ部分や静的データ部が VM 間で共有され，システム全体のメモリ使用効率が向上することが期待できる．そこで，root ファイルシステム全体を DCSS ファイルシステムとして実装した場合のメモリ使用効率を表 1 の DCSS 行に示す．本評価で root ファイルシステムに格納したディレクトリは，/lib, /lib64, /bin, /sbin, /usr/lib, /usr/lib64, /usr/bin, /usr/sbin, /opt である．

表中 DASD は通常の実装であり，root ファイルシステムは DASD 中に格納され，コマンド実行の際にはそれぞれの VM でメモリを消費する．一方，DCSS では，root ファイルシステムのコマンドは，DCSS 領域に格納されているので，実行時には実際にはファイルの読み込みは行なわれず，DCSS 上のコマンドの実行イメージがそのまま主記憶上で実行される．その時，他の VM で同じコマンドが実行中ならば，そのコマンドを含むページは物理メモリにロードされているはずであり，同じ物理メモリのページを複数の VM で共有できるのでメモリの使用効率が向上する．

DCSS ファイルシステムにあるコマンドであっても，どの VM からも長時間参照されていない場合，そのページは VM によってページアウトされるので，物理メモリを不要に占有することは少ない．

表 1 から分かるように，Linux-VM 当り約 22%程度の物理メモリの節減効果があり，これは Linux カーネルによる物理メモリの節減効果よりも極めて大きい．

3.1.3 Linux システムの共有と CMS との比較

表 1 の NSS+DCSS は，NSS による Linux カーネルの共有と DCSS ファイルシステムによる Linux コマンド共有の両方を行なった場合のメモリ使用量である．NSS と DCSS の VM 上でのメモリ領域は独立しているので，節減できるメモリ量は両者の合計である．Linux が稼働している状態ではおよそ 25%の物理メモリ節減効果があることが分かった．

表の下段には，比較のため CMS(Conversational Monitoring System) でのメモリ使用量を示した．CMS は VM のためのシングルタスク会話モニタであり，VM 間のメモリ共有を強く意識した設計となっている．シングルタスクであるため，Linux にあるような daemon は起動されず，起動時はモニタのみが動作する．NSS を用いて CMS の実行ファイルを共有する場合，物理メモリの節減率は約 40%である．

Linux カーネルでの物理メモリの節減効果は 2.7%と少なかったが，daemon も含めた Linux-VM 当りの使用メモリの節減効果は 25%であり，CMS には届かないものの，十分な物理メモリの節減効果が得られた．

表 2 DCSS ファイルシステム共有の内訳
Table 2 Detail of shared DCSS file system

Directory	WSS (MB)	delta (MB)	delta (%)	dir size (MB)	delta / size
no share	216.7	-	-	-	
/bin	214.4	2.25	1.0%	8.5	26.5%
/sbin	214.4	2.23	1.0%	15.7	14.2%
/lib	215.4	1.28	0.6%	54.4	2.3%
/lib64	212.7	3.99	1.8%	10.7	37.3%
/opt	216.1	0.61	0.3%	51.2	1.2%
/usr/bin	213.7	2.96	1.4%	48.6	6.1%
/usr/sbin	215.3	1.40	0.6%	13.3	10.5%
/usr/lib	209.5	7.20	3.3%	354.7	2.0%
/usr/lib64	188.3	28.36	13.1%	146.2	19.4%

3.1.4 共有するファイルシステムの選択

DCSS ファイルシステムは，仮想記憶を含めた主記憶の一部をファイルシステムに使用するため，高コストなファイルシステムである．このため，複数 VM 間で共通して使用するファイルを選択し，共通するファイルを DCSS ファイルシステムに格納すべきである．

そこで，**表 2** に root ファイルシステム下の各ディレクトリごとの DCSS ファイルシステムによる物理メモリの節減効率を示す．それぞれのディレクトリのみを DCSS ファイルシステムに格納した場合の WSS 数を得ることによって，各ディレクトリのメモリ節減への貢献度を測定した．表中，no share はファイルシステム共有を一切行なわない場合の 1 Linux 当りの WSS 数，ファイルシステムごとの数値は，それぞれのファイルシステムを DCSS によって共有した場合の 1 Linux 当り WSS 数を示す．delta は，それぞれのファイルシステム共有での no share に対する差分である．表より /lib64, /usr/lib64, /bin のメモリ節減効率が高いことが分かる．この結果より，/lib64, /usr/lib64 に格納されているシェアードライブラリがメモリ使用量の節減に大きく貢献していることが推測できる．/bin がメモリ使用量の節減に貢献しているのは，OS の動作に必要な使用頻度の高いコマンドの多くが/bin に格納されているからである．

今回，ファイルシステムごとに共有効率を調べたが，

表 3　Apache2 でのメモリ共有効率
Table 3　Memory Sharing Efficiency of Apache2

	Before apache2 started (MB)	After apache2 started (MB)	Memory comsumption by apache2 (MB)	Efficiency (%)
DASD	152.8	160.3	7.5	
DCSS	128.0	132.9	4.9	34.8%

厳密にはファイル単位での調査ができれば，共有効率が更に向上する．しかしながら，DCSS はファイルシステム単位なので，ファイル単位の共有を実現するためには，シンボリックリンクの自動生成など，別途の実装を必要とするので，ファイル単位共有については今後の課題とする．

3.2　アプリケーション・ソフトウェアのメモリ共有効率

3.2.1　サーバ向けソフトウェアによる評価

表 3 に代表的なサーバ向けアプリケーションである apache2 による VM 間メモリ共有の効率を示す．表中，DASD は apache2 関連のファイル (バイナリを含む) を通常のディスクに置き，そこから読み込んだ場合，DCSS は apache2 関連のファイルを DCSS ファイルシステムに格納し，VM 間で共有した場合のシステムが使用した物理メモリ量を示す．apache2 の消費メモリ量は，apache2 の起動前後の WSS の差分を MB 単位に換算した．本評価で用いた Linux は SuSE Enterprise Server 11 GM (64bit) である．

表から分かるように，DASD から直接読み込む場合は apache2 の実行のために 7.5MB の物理メモリを消費していたが，apache2 関連のファイル一式を DCSS ファイルシステム中に格納することにより，4.9MB の物理メモリの消費で済む結果となった．つまり，apache2 でのメモリ節減効率はおよそ 34.8%となり，これは Linux OS の VM 間メモリ共有による物理メモリ使用量の節減効果よりも大きい．

3.2.2　クライアント向けソフトウェアによる評価

本節では，セキュリティを高めるため，VM 間で完全に空間を分離しつつも，多くのユーザーが使用するアプリケーションを共有した場合について評価する．実際の使用例としては，例えば給与計算や人事，サービスセンターにおける機密業務など，各オペレーターが専ら Web を通じたインターフェースでシステムにアクセスしつつも，オペレーター間の秘密性を厳格に管理したい用途などが想定される．

図 3 に，システム内に複数の Linux VM を起動し，各アプリケーションの動作を含む稼働 VM 数と，システム全体で消費される物理メモリ量の関係を示した．

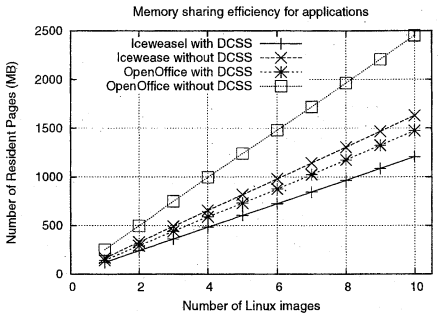

図 3　複数 Linux 起動時におけるアプリケーションのメモリ消費
Fig. 3　Memory Consumputin by applications under multi Linux

Iceweasel は Mozilla Firefox とほぼ同等の Web ブラウザである．測定環境としては，z/VM 5.4, OS は z Linux, Debian ver. 5.3, Iceweasel ver. 3.0.12 を用いた．グラフより，Icewasel は VM が一つ増えるごとに，DCSS 無しの場合で 164MB, DCSS 有の場合で 121MB の物理メモリが消費されていることが分かる．各 VM 間は基本的に同じ条件で起動しているため，VM の数が増えるごとに物理メモリの消費は直線的に増加する．DCSS の有無の差 (約 43MB) が DCSS によって VM 間で共有されたメモリ量であり，これはテキストや静的データなど RO メモリ量に相当する．DCSS 有の 121MB はヒープやスタックなど R/W メモリ量の増加を意味する．Iceweasel の物理メモリ使用量の節減効果は 1 Linux 当たり 26%であり，複数の Linux の場合でも同率の削減効果が得られた．

次に Open Office の Impress でのメモリ節減効率を調べた．Open Office は Microsoft Office に近い機能を有するフリーのオフィスソフトウェアであり，本評価ではその中の Impress(プレゼンテーションソフト) を用いた．測定環境は，z/VM 5.4, OS は z Linux, Debian ver. 5.3, Open Office ver. 2.4 を用いた．

Iceweasel の場合と同様に，稼働 VM 数の増加に従って物理メモリの消費量が線型に増加する．VM 一つあたりの Open Offoce による物理メモリ消費は DCSS 無しの場合で 252MB, DCSS 有の場合で 142MB である．DCSS の有無の差が DCSS によって VM 間で共有されたメモリ量であり，Open Offoce の物理メモリ使用量の節減効果はおよそ 40%と大きい．

4.　z/VM 以外の VM 機構での可能性

本稿では IBM System z 上で稼働する z/VM での測定結果を示したが，同様の手法を VMware など他

表 4 各アプリケーションでのメモリモードと RSS
Table 4 Memory Mode and RSS of Applications

		Iceweasel			Open Office (scalc)		
	mode	Size	RSS	%	Size	RSS	%
z/VM x86	rw-p	4.06	3.98	38.9	2.16	2.07	13.2
	rwxp	13.2	1.20	11.8	11.5	1.14	9.2
	rwxs				0.01	0.01	0.0
	r-xp	10.6	5.02	49.1	28.7	10.8	69.1
	---p	0.22	0.00	0.0	0.00	0.00	0.0
	r-xs	0.03	0.03	0.2	2.28	1.32	8.4
	rw-p	19.5	5.37	56.7	13.8	3.64	25.0
	rw-s				0.01	0.01	0.0
	r-xp	6.73	3.98	42.0	22.2	9.54	65.5
	r--p	0.68	0.11	1.1	4.15	0.06	0.4
	r--s	0.02	0.02	0.2	2.28	1.31	9.0
	---p	0.23	0.00	0.0	0.01	0.00	0.0
	r-ws				0.01	0.01	0.0

の VM 機構で適用すれば，有効に機能することが期待できる．しかし，DCSS と同様な仕組みを他の VM 機構で実装し評価するのは，工数上容易ではない．そこで，稼働中のアプリケーションについて，OS 管理下のメモリモード (RO と R/W) の割合から，他の VM 機構での有効性について考察する．

表 4 に各アプリケーションでのメモリ・モードと RSS(Resident Set Size) およびそれぞれのモードのページが RSS 上に占める割合を示す．本評価では OS が管理しているメモリを観測しているため，これまでの WSS を用いた測定とは異なる．表中の数値の単位は k page (1page は 4kB) である．表中の z/VM は System z 版 Linux 上，x86 は x86 版 Linux 上での各アプリケーションでのメモリ使用状況である．

測定環境としては，VM については，z/VM 5.4, OS は z Linux, Debian ver. 5.3, PC について，OS は Linux (for x86), Debian ver. 5.3 を用いた．アプリケーションは，両プラットホーム上で Iseweasel および Open Office ver. 3.1.1 の scalc(表計算ソフト) を用いた．メモリ構成はアプリケーションの実行に従って，ユーザーデータが増えると R/W 領域が増加する．このため，本評価ではアプリケーション起動直後のページ状態を示した．

表中，"w" フラグが立っているメモリページは，書き込みが行なわれるため，明かに他の VM と共有できない．"w" フラグの無いページはテキストや静的データが格納される領域であり，理論上共有が可能である．Iceweasel および Open Office の両方ともにメモリの使用状況が z/VM と x86 版 Linux で一致しないが，共有可能な RO ページの割合は両者ともに比較的類似している．共有可能ページの割合は Iceweasel の場合，z/VM 版の 49.3%なのに対し，x86 版だと 43.4%，Open Office の場合，z/VM 版の 77.5%に対し，x86 版は 74.9%という結果が得られた．このことから，x86 版でも DCSS と同様な仕組みを設けた場合には，近いメモリ使用効率を得ることが期待できる．

5. 関連研究

本論文に関連する技術として次のようなものがある．

VMware ESX Server で実装されている手法として，"Memory Sharing" がある．これはハッシュ・アルゴリズムによって 4kB ページ単位で保持している内容が同一のメモリブロックを探し出し，同一内容のメモリブロックを共有することによってメモリの冗長性を排除している．

Memory Sharing の目的は，本論文の目的と極めて近いが実装手法が異なっている．効果としては，文献 13) によれば，最大で 33%ほどのメモリ節約が可能と報告されている．一方，同文献には述べられていないが，Memory Sharing は原理上，ロードする度にハッシュで同一ページを探すためのオーバーヘッドが発生する．これに対し，提案手法は事前に共有できるメモリを指定しているため，ページ比較のようなオーバーヘッドは全く無い．

一方，別のアプローチとして，メモリを動的に圧縮することにより，物理メモリの使用量を節減しようという試みもなされている[14),15)]．本手法は，長時間使用していないページをデータ圧縮し，読み出し時に伸張する方法である．ハードウェアによる支援なども行なわれ，伸張の高速化も試みられている．原理から明かなように，ページ読み出し時のペナルティが問題となる反面，メモリから読み出すデータ量も削減できるため，実質のメモリバンド幅が増大する利点もある．文献 15) では，別付けハードウェアによる圧縮伸張を用いて，ベンチマークの結果では圧縮伸張無しの場合とほぼ同等の性能を得ている．

仮想計算機におけるメモリの効率的利用法として，z/VM および VMware ESX Server で "Memory Overcommitment" が実装されている[11),13)]．本手法は，実ハードウェアとして搭載されているメモリ量よりも多くのメモリ容量を VM に提供するものである．この実装の狙いは，それぞれの VM が必ずしも自分に割り当てられたメモリ量を使い切る訳ではないことを利用し，メモリを使っていない VM に割り当てたハードウェア・メモリを，本当にメモリを必要とする VM に割り当てることによって，システム全体のメモリ利用効率の向上を図っている．

6. まとめ

本論文では，z/VM 仮想計算機で利用可能な DCSS を用いて VM 間メモリ共有によるメモリ使用効率を向上させ，その効率を評価した．本来 VM 間メモリ共有を考慮されていない Linux では，カーネルのメモリ共有では 2.7%の節減しか得られなかったが，DCSS ファイルシステムを用いることによって，主要な daemon についても RO メモリ領域を VM 間で共有することができ，Linux 全体では約 25%の節減が可能であることが分かった．また，Open Office などのアプリケーションでは，最大で約 40%の節減が可能であることが明かとなった．

本手法は apache2 などサーバ常駐型アプリケーションでも有効に機能するので，VM の主な用途であるサーバ統合において効果が期待できる．また，Icewasel(Web ブラウザ) や Open Office などのクライアント向けアプリケーションでも高い効果が認められた．VM をシンクライアントのホストとして利用する場合にも有益である．

DCSS は z/VM 上でしか実装されていないが，アプリケーションのメモリ使用状況を調べた所，x86 版 Linux においてもメモリのモードの割合はおおむね同様であり，x86 版の VM に本機構が実装されれば，本論文に近い効果が期待できる．

本手法の注意点としては，メモリは高価な資源であるので，例え使用されていなければページアウトされるとは言え，DCSS へのファイル配置は計画的に行なう必要がある点である．

今後の課題として，DCSS ファイルシステムはメモリ内の実行イメージに直接ジャンプし実行が開始されるので，アプリケーションの起動時間も大幅に短縮できるが，この定量的評価が挙げられる．

また，DCSS 内の全ファイルが RO モードとなり，OS のアップデート等が困難になる問題に対しては，レイヤーごとにファイルを重ねることができる aufs との併用が考えられる．

謝辞 本研究にて実験機器などの手配にご尽力頂いた山本 直史殿，白垣 英一殿始め，日本アイ・ビー・エムの支援部隊の皆様，ならびに本学情報科学センターの皆様に感謝の意を表します．

参考文献

1) Smith, J. and Nain, R.: *Virtual Machines: Versatile Platforms for Systems and Processes*, The Morgan Kaufmann Publishers (2005).
2) Figueiredo, R., Dinda, P. and Fortes, J.: Resource Virtualization Renaissance, *Computer*, Vol. 38, No. 05, pp. 28–31 (2005).
3) 大町一彦: 仮想マシン道しるべ: 仮想マシン草創期, 情報処理学会誌, Vol. 48, No. 8, pp. 903–905 (2007).
4) 川添 良幸 Gray R. McClain 編, 早川美徳, 小野木 隆監訳: VM アプリケーションハンドブック, 共立出版 (1992).
5) VM Ware: *SPECweb2005 Performance on ESX Server 3.5*, Technical report, Technical Resources (2008).
6) 佐藤幸紀, 上埜元嗣, 宇多仁, 井口寧, 敷田幹文, 松澤照男: 仮想サーバシステムのための環境管理支援ツールの構築, 分散システム・インターネット運用技術シンポジウム 2007 論文集, No. 13, pp. 77–82 (2007).
7) 金田憲二, 大山恵弘, 米澤明憲: 単一システムイメージを提供するための仮想マシンモニタ, 情報処理学会論文誌, Vol. 47, No. SIG 3 (ACS 13), pp. 27–39 (2006).
8) 丸山伸, 最田健一, 小塚真啓, 石橋由子, 池田心, 森幹彦, 喜多一: Virtual Machine を活用した大規模教育用計算機システムの 構築技術と考察, 情報処理学会論文誌, Vol. 46, No. 4, pp. 949–964 (2005).
9) 日本アイ・ビー・エム株式会社: IBM System z Hand Book (2007).
10) IBM: *System z*, http://www-06.ibm.com/systems/jp/z (2007).
11) Creasy, R.: The Origin of the VM/370 Time-Shareing System, *IBM J. Res. Develop*, Vol. 25, No. 5, pp. 483–490 (1981).
12) 北沢強: ゲスト OS とホスト OS のメモリ連携, メインフレーム Linux, http://www.atmarkit.co.jp/flinux/rensai/mf06/mf06b.html (2009).
13) VM Ware and Kingston technology: *The Role of Memory in VMware ESX Server 3*, Informationn Guide, http://www.vmware.com/pdf/esx3_memory.pdf (2008).
14) Benini, L., Bruni, D., Macii, A. and Macii, E.: Hardware-assisted data compression for energy minimization in systems with embedded processors, *Proceedings in Design, Automation and Test in Europe Conference and Exhibition*, pp. 449–453 (2002).
15) Benini, L., Bruni, D., Macii, A. and Macii, E.: Memory energy minimization by data compression: algorithms, architectures and implementation, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol. 12, No. 3, pp. 255–268 (2004).