1B－3

Tender のメモリ管理機能
－ヘテロ仮想記憶 (HVS)－

長嶋 直希　谷口 秀夫　牛島 和夫
九州大学大学院システム情報科学研究科

1　はじめに

我々はプログラム構造に重点をおいたオペレーティングシステム ***Tender*** を開発している [1]。

Tender では、プロセスと仮想メモリ空間を独立して管理している。これにより、一つの仮想メモリ空間上に複数のプロセスが存在する複数プロセス仮想メモリ空間 (MPVMS: Multiple Process Virtual Memory Space) と、一つの仮想メモリ空間上に一つだけプロセスが存在する単一プロセス仮想メモリ空間 (SPVMS: Single Process Virtual Memory Space) を混在させることができる [2]。これをヘテロ仮想記憶 (HVS: Heterogeneous Virtual Storage) と呼ぶ。

本稿では、HVS の特徴を示し、実現時の問題と対処を述べる。さらに、***Tender*** への実装方式を説明する。

2　ヘテロ仮想記憶

2.1　従来の仮想記憶の特徴

仮想記憶のモデルとして、多重仮想記憶 (MVS: Multiple Virtual Storage) と単一仮想記憶 (SVS: Single Virtual Storage) がある。MVS は、SPVMS が複数存在する仮想記憶モデルである。これに対し、SVS は、MPVMS が一つだけ存在する仮想記憶モデルである。以下に MVS と SVS の特徴を示す。

MVS では、各プロセスは固有の仮想メモリ空間を持ち、それぞれの仮想メモリ空間でアクセス保護がなされている。このため、各プロセスのメモリ操作に関する保護は強い。しかし、プロセス間通信やプロセスディスパッチには、カーネルを介した処理が必要となり、その分、オーバーヘッドが大きい。また、プロセスディスパッチには、仮想メモリ空間の切替を伴うため、それまでの論理キャッシュや TLB (Translation Look-aside Buffer) の内容が無効になり、メモリアクセス速度が遅くなるという欠点もある。

SVS では、全てのプロセスは一つの仮想メモリ空間上で走行する。このため、プロセス間通信やプロセスディスパッチの処理が MVS に比べ、簡単である。さらに、プロセスディスパッチ時に仮想メモリ空間の切替が必要ないため、切替前の論理キャッシュや TLB を有効に用いることができる。しかし、プロセスのメモリ操作に関する保護が弱い。また、アドレス空間の大きさは有限であるため、プロセス一つあたりの大きさが、MVS の場合と比較して小さくなるという欠点もある。

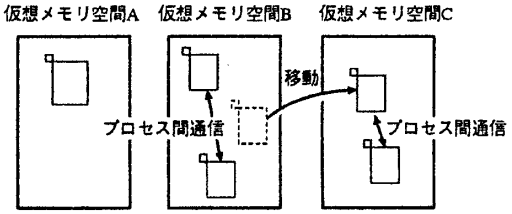

図 1　ヘテロ仮想記憶

2.2　HVS の特徴と利用法

HVS とは、SPVMS と MPVMS が融合した仮想記憶モデルである。SPVMS は MPVMS 上にプロセスが一つしか存在していない仮想メモリ空間と考えられるので、HVS は MPVMS が複数存在する仮想記憶モデルとみなすことができる。

HVS の様子を図 1に示す。HVS の特徴は、プロセスの処理内容に合わせて、適切な仮想メモリ空間を提供することにより、SVS と MVS の長所を生かすことができる点である。具体的には、プロセス間通信を頻繁に行なうプロセスやディスパッチを頻繁に行なうプロセスは同一の仮想メモリ空間上に置くことにより、SVS の長所を得ることができる。図 1 の仮想メモリ空間 B 上のプロセスがこれにあたる。逆に、強い保護を要求するプロセスや暴走を起こして他のプロセスを破壊する危険のあるプロセスは別の仮想メモリ空間上に配置することにより、プロセス間の保護を強くすることができる。図 1 の仮想メモリ空間 A 上のプロセスがこれにあたる。

さらに、HVS の機能として、プロセスが現在動作している仮想メモリ空間上から、別の仮想メモリ空間に移動できれば、以下のような利点を得ることができる。

(1) 他の仮想メモリ空間上に存在するプロセスとプロセス間通信を頻繁に行なう場合、プロセス間通信を行なうプロセスを同じ仮想メモリ空間上へ移動して、プロセス間通信にかかるオーバーヘッドを削減することが可能となる。

Memory Management on *Tender*.
－ Heterogeneous Virtual Storage －
Naoki NAGASHIMA, Hideo TANIGUCHI, and Kazuo USHIJIMA
Graduate School of Information Science and Electrical Engineering, Kyushu University
Email:nagasima,tani,ushijima@csce.kyushu-u.ac.jp

図1では、仮想メモリ空間B上のプロセスが仮想メモリ空間C上のプロセスと通信を行なうために仮想メモリ空間C上に移動する様子を示す。

(2) MPVMSでは、プロセスの大きさが同じ仮想メモリ空間上に存在する他のプロセスによって制限されるため、プロセス実行途中で広いメモリ空間を必要とした場合、確保できないことがある。この場合、メモリ空間の確保が可能な他の仮想メモリ空間にプロセスを移動させることにより、この問題を解決できる。

2.3 実現時の問題と対処

(問題1) プロセス移動時のアドレス衝突 プロセスが別の仮想メモリ空間に移動した時に、移動先の仮想メモリ空間に既に存在しているプロセスとアドレスの衝突を起こす可能性がある。この問題への対処法を以下に示す。

(対処A) プロセス生成時にアドレス衝突が生じないように、アドレスをずらす。

(対処B) プロセス移動時に、アドレスを変更する。

(対処A)は実現が容易であるが、プロセスの大きさが制限されるとともに、内部断片化が生じる欠点がある。(対処B)はプロセスのアドレス配置が最適状態に保たれるという利点があるが、実行中のプロセスのアドレスを変更する必要があり実現は難しい。

(問題2) 仮想メモリ空間の外部断片化 一つの仮想メモリ空間上で、プロセスの作成・削除やプロセスの仮想メモリ空間移動を繰り返すと、仮想メモリ空間の空き領域が断片化する。これにより、プロセスの大きさが空き領域の合計よりも小さいにも関わらず、プロセスの大きさよりも大きい連続領域が存在せず、プロセスを配置できないという問題が生じる。この問題の対処法として、上記に示した(対処B)が考えられる。

3 *Tender* での実装方式

3.1 メモリ管理の概要

Tender におけるメモリ管理は、4つの資源「実メモリ」、「仮想領域」、「仮想カーネル空間」、「仮想ユーザ空間」を管理する各資源管理によって実現される。これらの関係を図2に示し、以下に説明する。

資源「実メモリ」は、実メモリ上の領域である。資源「仮想領域」は、メモリイメージを仮想化した領域で、その実体は実メモリ、または、外部記憶装置上に存在する。「仮想領域」は仮想空間に貼り付けることにより、資源「仮想ユーザ空間」、または「仮想カーネル空間」として扱われ、アクセス可能となる。つまり、「仮想領域」に直接アクセスすることはできない。「仮想カーネル空間」はカーネルモードでのみアクセスでき、「仮想ユーザ空間」はカーネルとユーザの両モードでアクセスできる。

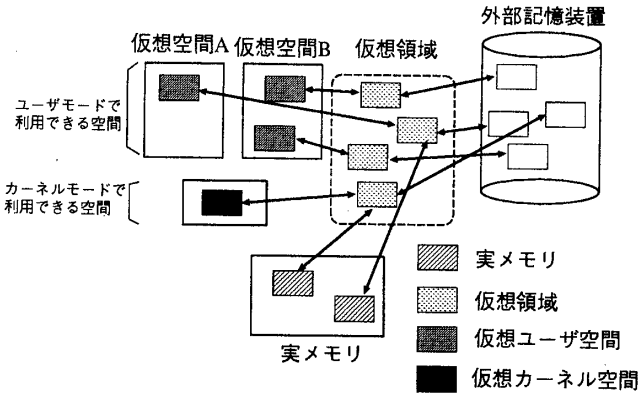

図2 メモリ管理機能を実現する各資源の関係

3.2 実装方式

メモリ管理は、4つの資源管理によって実現され、プロセス管理は、これらの資源管理によって管理される資源を用いてプロセスを生成する。***Tender*** ではプロセスも一つの資源である。各資源の管理は独立しているため、仮想空間は、プロセスの存在に関係なく、複数存在することができる。一つの仮想空間に複数の「仮想領域」を貼り付けて、複数の「仮想ユーザ空間」を生成できる。一つの「仮想ユーザ空間」をプロセスが利用するメモリ空間として対応づける。これにより、HVSを実現できる。

プロセスの生成手順を次に示す。プロセス管理は、まず「仮想領域」を生成する。プロセスを新たな仮想空間上に生成する場合は、新たに仮想空間を作成し、その仮想空間に「仮想領域」を貼り付けて「仮想ユーザ空間」を生成する。プロセスを既存のプロセスと同じ仮想空間に生成する場合は、そのプロセスが存在する仮想空間に「仮想領域」を貼り付けて「仮想ユーザ空間」を生成する。「仮想ユーザ空間」上にプログラムやデータを読み込み、プロセスが生成される。

プロセスの仮想空間移動の手順を次に示す。プロセスの持つ「仮想領域」を移動先の仮想空間に貼り付けて、「仮想ユーザ空間」を作成する。そして、それまでプロセスが走行していた仮想空間の「仮想領域」をはがし、「仮想ユーザ空間」を削除する。

4 今後の課題

今後の課題として、HVSを ***Tender*** へ実装し、評価を行なう。

参考文献

[1] 谷口秀夫：“分散指向永続オペレーティングシステム ***Tender***”, 情報処理学会シンポジウム論文集 Vol.95, No.7, pp.47-54 (1995).

[2] 村上大介, 青木義則, 谷口秀夫, 牛島和夫：“Tenderにおける資源「演算」の扱い”, 情報処理学会第51回全国大会予稿集, 5L-8 (1995).