3U-8

稠密処理システムＧＡＬＡＸＹの アドレススペースの考え方

河内谷清久仁、前川　守、太田昌孝、濱野　純
(東京大学)

１．はじめに

稠密処理(Holonic Processing)は、バッチ処理、ＴＳＳ処理、分散処理と発展してきた計算機の利用形態の次世代を担うものとして我々が提唱しているものである。これは、ネットワーク全体が協調して働く生物体のように有機的なコンピュータシステムを実現しようというものである。

ＧＡＬＡＸＹは現在我々が設計中の稠密処理システムである。今回は、地理的に広い範囲にばらまかれた情報及び資源を統合して扱うためにＧＡＬＡＸＹが用いている方法について説明する。この際に目標となるのは、ユーザが全ての情報や資源を自分のワークステーション上にローカルに存在しているかのように利用できる「トランスペアレント・ネットワークシステム」の構築である。オブジェクトの管理及びネーミングのメカニズムに関しては他の二稿(３Ｕ-７,９)で述べられている。本稿では、プログラムレベルでのインタフェースであるアドレススペースの考え方と、そのメカニズムについて説明する。

２．ＧＡＬＡＸＹのアドレススペースのイメージ

アドレススペースは、プログラムからみたときのハードウェアとの最終的なインタフェースである。ＧＡＬＡＸＹのアドレススペースは図１のようなイメージで表わされる。ＧＡＬＡＸＹの世界は、オブジェクトがちりばめられた一つの広大なインフォメーションスペースとして与えられ、プロセスはその一部分を自分のインフォメーションスペース（アドレススペース）として用いる。このアドレススペースは必要に応じて、ＧＡＬＡＸＹスペース内を移動したり、他のプロセスのアドレススペースと重なったりすることが可能である。

このようなイメージでアドレススペースをとらえることは、次のような意味で好都合である。

a)システム中の情報や資源をシングルレベルのインフォメーションスペースを通して扱うことにより、プログラムが単純になる。あらゆるデータに対するアクセスは、それが各自のワークステーションのメモリ内にあるかのごときイメージで行える。

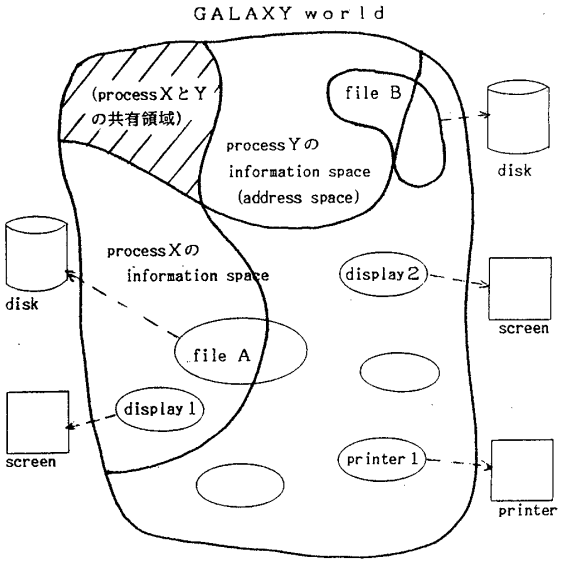

図１　アドレススペースのイメージ

b)ファイルへのアクセスは、Ｉ／Ｏ命令を用いるよりもＬＯＡＤ、ＳＴＯＲＥのようなメモリアクセス命令で行なえる方が効率がよい。

c)ＧＡＬＡＸＹシステム中に存在する全ての情報や資源に対しアクセスすることができる。また、システムの成長（ワークステーションの追加など）に対応してアドレススペースからアクセスできる範囲も広くなる。

d)次で述べる各種のトランスペアレンシの要求にマッチしている。

３．ＧＡＬＡＸＹのアドレススペースに対する要求

次に、上で示したイメージの実現に際して満たすべき条件を考える。まず第一に、ＧＡＬＡＸＹの設計目標のひとつである「トランスペアレント」ネットワークシステムの構築のため、アドレススペースのメカニズムは以下のような要求を満たしている必要がある。

1)位置のトランスペアレンシ----アドレススペースはシステムワイド（ネットワークワイド）なものでなければならない。即ち、ＧＡＬＡＸＹ内の全ての情

Address Space of ＧＡＬＡＸＹ Holonic Processing System
Kiyokuni Kawachiya , Mamoru Maekawa , Masataka Ohta , Jun Hamano
University of Tokyo

報及び資源を、アドレススペースを通して、物理的位置とは無関係な統一的な形で取り扱えなければならない。

2)リプリケーションのトランスペアレンシ----他稿で述べたように、ＧＡＬＡＸＹはオブジェクトのリプリケートを許している。しかしプログラムからはリプリケーションを意識することなくオブジェクトにアクセスできなければならない。

3)並列性のトランスペアレンシ----ＧＡＬＡＸＹはネットワーク上で動くシステムであり、同時に多くのプロセスが実行される。これらのプロセス間でコンシステンシを失うことなくデータのアクセスが行なえなければならない。

またその他にも、アドレススペースは次のような機能を持っていなければならない。

4)モジュール化と性能向上のためにセグメンテーション及びページングのメカニズムを備えていること。

5)画像データのような大容量のデータも扱えること。

４．ＧＡＬＡＸＹのアドレススペースの構造

２及び３で述べたことをふまえて、ＧＡＬＡＸＹではネットワークワイドなセグメンテーション／ページング機能を備えたアドレススペースを提供している。図２にそのイメージを示す。プロセスのアドレススペースはセグメントに分けられており、各セグメントにＧＡＬＡＸＹオブジェクト（の一部分）をマッピングすることによってオブジェクトに対するアクセスが行なわれる。セグメントからオブジェクトへのマッピングはオブジェクトのＩＤを使ってロケーティング関数によって行なわれる（他稿を参照）。セグメントの内部はさらにページに分けられており、必要なページのみがメモリ中にフェッチされてくる。

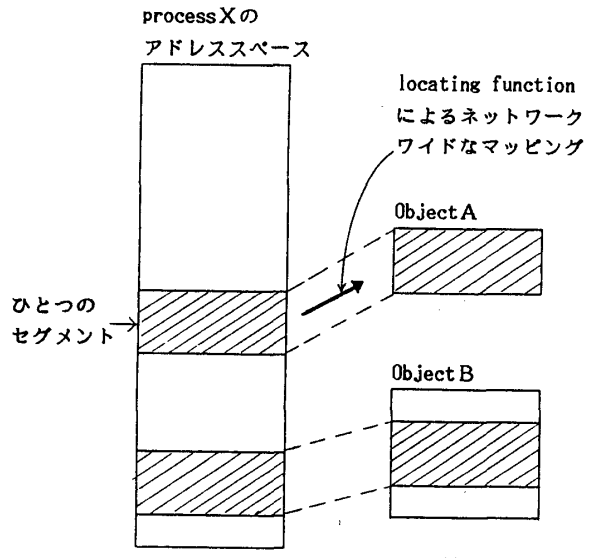

図２　セグメントとオブジェクトの対応

図３にアドレスから実体へのアクセスのメカニズムを示す。アドレスはセグメント番号ｓ、ページ番号ｐ、ページ内でのオフセットｗの３つのフィールドから成る。まずｓを用いてセグメントテーブルから対応するセグメントのエントリがフェッチされる。セグメントテーブルはプロセスごとに存在し、プロセスオブジェクトの実体情報(entity information)としてＩＤテーブル中で管理されている。セグメントテーブルの各エントリは通常のセグメント管理情報の他に、そのセグメントにマッピングされているＧＡＬＡＸＹオブジェクトのＩＤを含んでいる。ページフォールトが起こった場合、ロケーティング関数が起動され、そのＩＤに対応する実体の１ページがメモリ中にフェッチされてくる。ここで重要なのはロケーティング関数はネットワークワイドなものであり、ページフェッチがネットワークを通じて行なわれるかもしれないということである。ページテーブル以下の構造は一般的なページングのものと同様である。

オブジェクトのフェッチにロケーティング関数を用いることによって、３で示した要求1)、2)、3)は自動的に満たされることになる。また5)の要求はセグメントにオブジェクトの「一部分」をマッピングするための情報をセグメントテーブル中に設けることで解決される。

５．まとめ

本稿では、稠密処理システムＧＡＬＡＸＹのプログラムレベルのインタフェースであるアドレススペースについて、その考え方を述べ、実現法としてネットワークワイドなセグメンテーション／ページングのメカニズムを示した。これによってアドレススペースは各種のトランスペアレンシの要求を満たすことができる。

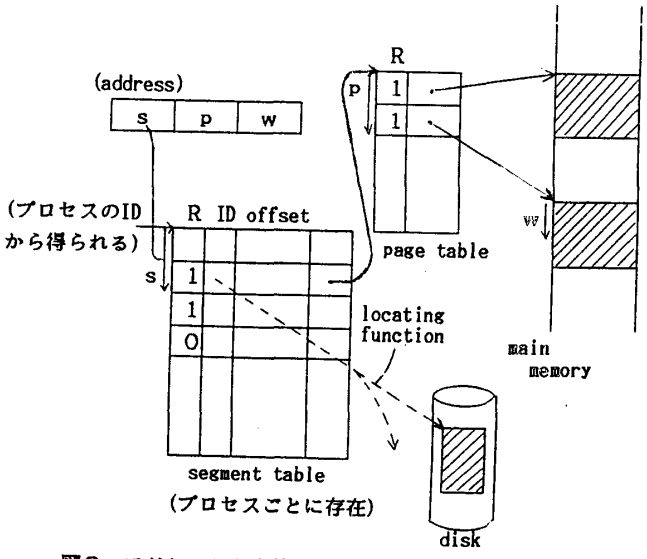

図３　アドレスから実体へのアクセス