分散環境における構造体管理

六沢 一昭*　　近山 隆*　　中島 浩**　　瀧 和男*
市吉 伸行*　　中島 克人*　　稲村 雄*

*（財）新世代コンピュータ技術開発機構　　**三菱電機(株)

1．はじめに

分散環境では，論理的には同一な構造体を同じものであると認識することがむずかしい。例えば，ＰＥが異なると同じ構造体も異なったものに見えてしまう。

ところが「同じであること」を検出できないと，同じ構造体を別々に割付けてしまいメモリをいくらでも使ってしまう恐れがある。また同じ構造体を何度も転送してしまいネットワークトラヒックの増大を招く恐れもある。構造体のサイズが大きい場合，これは大きな問題である。

本稿では，ユニークなｉｄを用いて構造体が同じものであることを検出し無駄なメモリ消費や不要な転送を防ぐ方式について述べる。この方式は Multi-PSI/V2 上のＫＬ１処理系[1,2] に適用している。

2．計算モデル

以下に示す計算モデルを仮定する。

- 局所メモリを持つ有限個のプロセッサ（ＰＥ）がネットワークで結合されている。
- 共有メモリはなく，メッセージ通信によってのみＰＥ間処理が行なわれる。
- ＰＥは自分の局所メモリ中のデータを他のＰＥから参照可能にすることがある。このため他のＰＥにあるデータへの参照ポインタ("外部参照ポインタ" と呼ぶ)をＰＥは持ちうる(図１)。
- 外部参照ポインタをもつＰＥは read要求を送信しポインタが指すデータを読みだすことができる。これを "データの輸入" と呼ぶ(図１)。

1) PEi には PEj のデータを指す外部参照ポインタがある。

2) PEi は PEj へ read要求を送信し，データを読みだした。

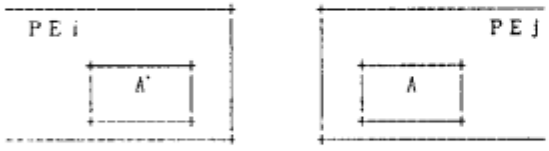


図１．外部参照ポインタとデータの輸入

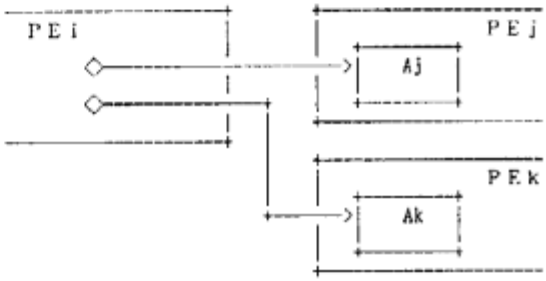
3．構造体管理の必要性

構造体の管理を行なわないと，「同じであること」を検出できず無駄なメモリ消費や不要な転送を行なってしまうことがある。

(a) 異なる輸出元からの輸入

論理的には同じ構造体であっても異なったＰＥに存在すると異なったものに見えてしまう。図２1) の２つの外部参照ポインタは論理的に同じ構造体を指すが PEi にはわからない。このため read要求を２回送信してしまい，不要な転送が起きてしまう。そして輸入した２つの構造体 Aj',Ak' は同じものであることがわからないため，別々に割付けてしまいメモリを無駄に使ってしまう。

1) Aj と Ak は論理的に同じものであるが PEi にはわからない。



2) PEj は PEj,PEk へ read要求を送信し Aj,Ak を読みだした。Aj' と Ak' は論理的には同じものであるが別々にメモリに割付けられてしまう。

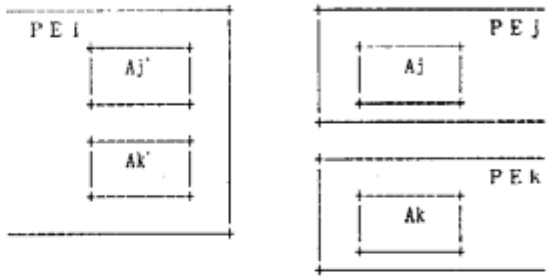


図２．異なるＰＥからの輸入

(b) ループ構造をもつ構造体の輸入

輸出元が同じであってもループ構造をもつ構造体を輸入する場合はやはり同じものであることが認識できない。図３に例を示す。図３2) において輸入した Y' が指す X は前回輸入した X' と同じものであるが PEi にはわからない。このため再度 read要求を送信してしまい構造体の不要な転送が起きてしまう。また輸入した構造体X',X'',... 及び Y',Y'',... はすべて異なったものと認識されるため別々に割付けてしまいメモリを無駄に使ってしまう。

(a),(b) どちらも無駄なメモリ消費と不要な転送が起きているが (a) はあまり深刻ではない。それぞれの輸入処理は独立しているので，ｎ個の構造体を輸入するにはｎ個

Structure Management for Distributed Processing Systems

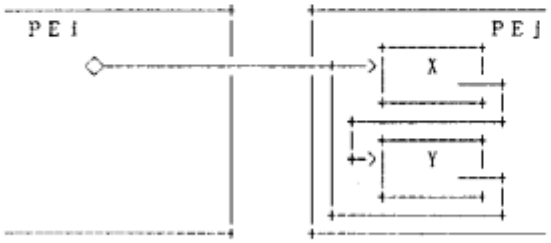
Kazuaki ROKUSAWA *　　Takashi CHIKAYAMA *
Hiroshi NAKASHIMA **　　Kazuo TAKI *
Nobuyuki ICHIYOSHI *　　Katsuto NAKAJIMA *
Yu INAMURA *　　(* ICOT　　** MELCO)

のプロセスが必要である。そして対応するプロセスがなくなれば輸入した構造体は解放可能になる。

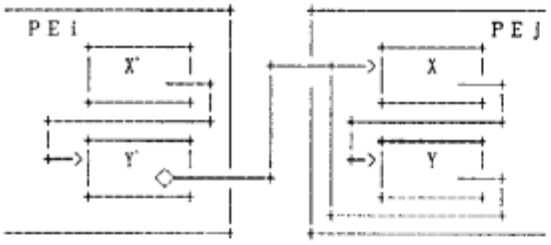
ところが (b) は深刻である。ある輸入によって新しい外部参照ポインタが生まれそれが次の輸入を招く形で輸入処理が進むため，プロセスが１つあるだけで何回でも輸入を行なってしまう。また輸入の結果作られる各構造体はチェインしているので，大本と末端の構造体(図３における X' と Y''') を指す２つのプロセスが存在する限り輸入した構造体はすべて解放できない。

(b) のようなことは決して珍しくない。特にコードモジュールではいくらでも起こる普通のことである。

1) 互いに指し合う構造体 X, Y が PEj にある。



2) PEi はまず X を輸入し(X' とする)，次に X' を読んで Y を輸入した(Y' とする)。Y' は PEj の X を指してしまう。



3) 2) を繰り返すと，同じ構造体を何度でも輸入し，輸入した構造体はすべて新しく割付けてしまう。

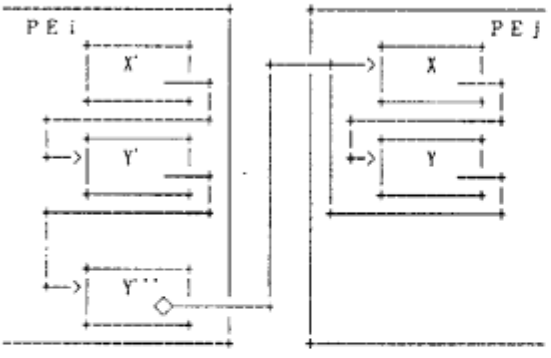


図３．ループ構造をもつ構造体の輸入

４．ユニークなid を用いた構造体管理

ユニークなid を用いた構造体管理方式について述べる。この管理方式では「同じであること」を検出し，無駄なメモリ消費や不要な転送を防ぐことができる。

４－１．構造体id

一度でも輸出入にからんだことのある構造体には "構造体id" を付ける。このid は完全にユニークなid で，同じid を持つ構造体は論理的には全く同じものであるようにする。id の生成は，例えばＰＥごとにカウンタを持ち必要に応じて生成すればよい。カウンタ値とＰＥ番号の組み合せでユニークなid が構成できる。

同じid をもつ構造体が１ＰＥ内で１つしかないように管理することは容易であるので，構造体への read要求に対していつも構造体本体とこの構造体id を送ることにより無駄なメモリ消費は防ぐことができる。id を受信した時，同じid を持つ構造体がＰＥ内に存在するかどうか調べ存在したならば受信した構造体を棄ててしまえばよいからである。しかしこれでは構造体の不要な転送は防げない。

４－２．構造体本体を転送するタイミング

構造体への read要求を受信した際，構造体本体は転送せず構造体id のみを送る。構造体id を受信したらそのid を持つ構造体あるいは外部参照ポインタがＰＥ内に存在するかどうか調べる。存在する場合は何もしない。存在しない場合は外部参照ポインタにid を付加し，再度read要求(id既知のread要求) を送信する。id既知のread要求を受信した場合は構造体本体を送信する。構造体本体を転送するのはこの時だけである(図４)。

同じ構造体を指すポインタが図２のように複数あってもid既知のread要求は１つのid に対して１回しか送信されない。このため不要な転送を防ぐことができる。図３の場合は X に対して２度目の read要求を送信すると付与されているid が返り，そのid から自ＰＥ内の X' を指すことがわかるため，もはや read要求は送信しない。

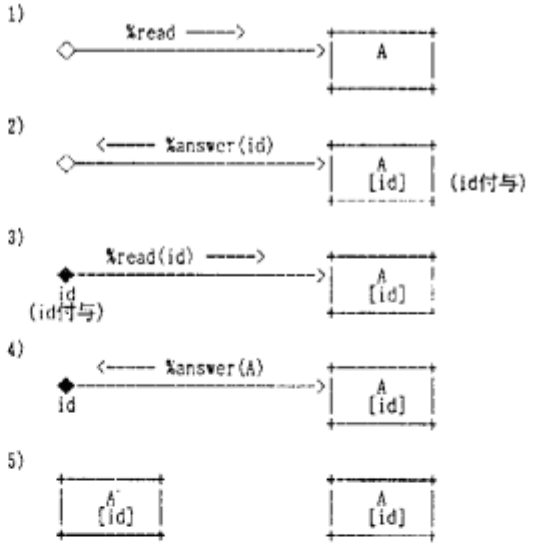


図４．id管理下における構造体本体の転送

５．おわりに

ユニークなid を用いた構造体管理方式について述べた。本方式では，同一であることを分散環境においても検出し無駄なメモリ消費や不要な転送を防ぐことができる。

<参考文献>

[1] K.Taki, "The Parallel Software Research and Development Tool: Multi-PSI system," Technical Report TR-237, ICOT, 1987.

[2] K.Nakajima et al, "Distributed Implementation of KL1 on the Multi-PSI/V2," ICLP, 1988.