

分散環境における構造体管理

1Q-3

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フィ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。

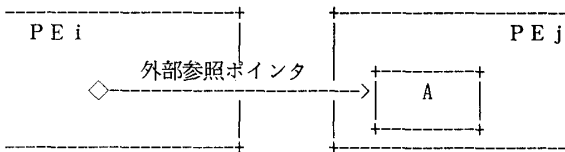
本稿では、ユニークなidを用いて構造体が同じものであることを検出し無駄なメモリ消費や不要な転送を防ぐ方式について述べる。この方式はMulti-PSI/V2上のKL1処理系[1,2]に適用している。

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。

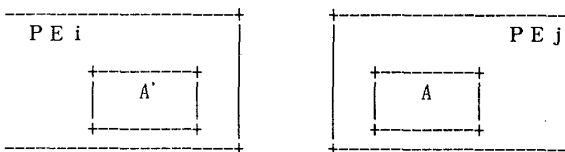


図1. 外部参照ポインタとデータの輸入

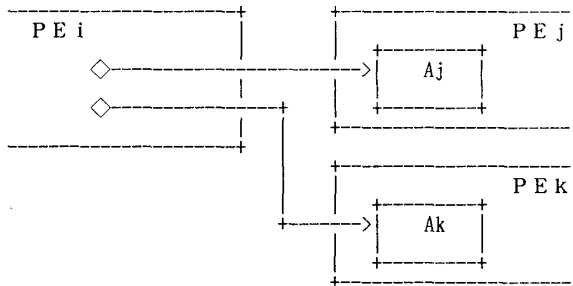
3. 構造体管理の必要性

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) PEi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。

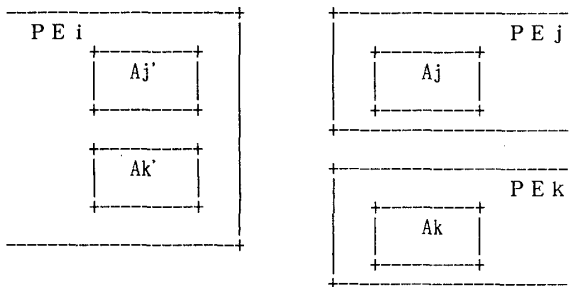


図2. 異なるPEからの輸入

(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。

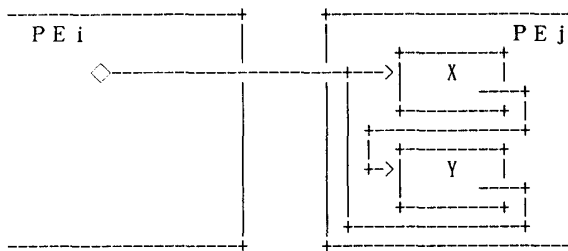
(a), (b) どちらも無駄なメモリ消費と不要な転送が起きているが(a)はあまり深刻ではない。それぞれの輸入処理は独立しているため、n個の構造体を輸入するにはn個

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。

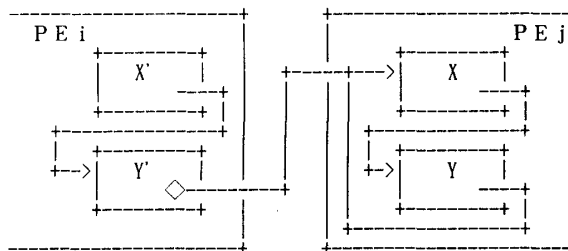
ところが (b) は深刻である。ある入力によって新しい外部参照ポインタが生まれそれが次の入力を招く形で輸入処理が進むため、プロセスが1つあるだけで何回でも輸入を行なってしまう。また輸入の結果作られる各構造体はチェーンしているため、大本と末端の構造体(図3におけるX'とY'')を指す2つのプロセスが存在する限り輸入した構造体はすべて解放できない。

(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 を入力し(X'とする)、次に X'を読んで Y を入力した(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。

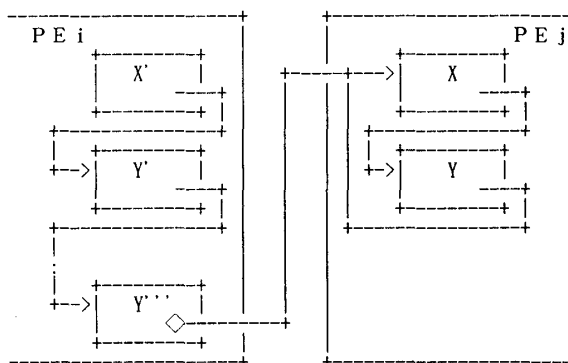


図3. ループ構造をもつ構造体の輸入

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番号の組み合わせでユニークな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。

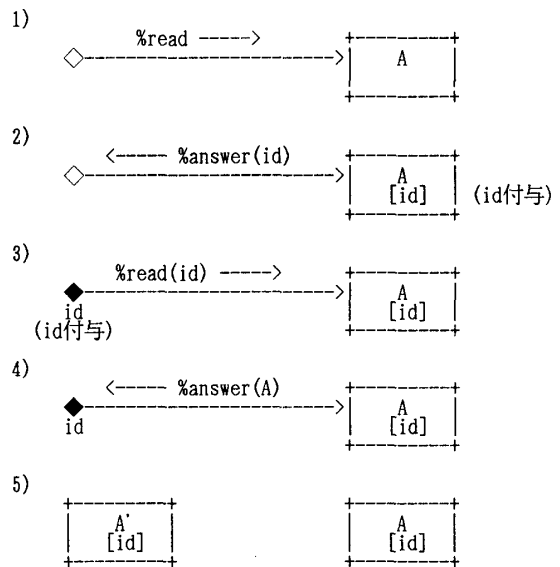


図4. id管理下における構造体本体の転送

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.