3R-5

並列オブジェクト指向言語DinnerBellの領域管理

河野真治　田中英彦
東大　工学部

はじめに

DinnerBellは，オブジェクト指向にデータフロー・単一代入則を導入した言語である。ここでオブジェクト指向のシステムとはは，データとその手続きを統一的に扱う意図を持つシステムを意味する。DinnerBellでは，単一代入則を採用することにより，通常のメッセージパッシングには，副作用が存在しない。ここでは，実際の並列実装の際のDinnerBellのデータ構造について考察する。

DinnerBellのオブジェクト

もちろん，DinnerBellのオブジェクトは，クラス定義から生成されるインスタンスであり，これが，各プロセッサ間に分配される。DinnerBellのインスタンスは，次のような要素からなる。

①itBlock variable　(インスタンス変数)
　side effect variable
　pure variable
②method variable　(一時変数)

これは，それぞれ次のような特徴をもっている。

①メソッドコールの後も値を保持する。
②メソッドコールと同じ寿命をもつ。

これらの変数は，ＯＯＰ (Object Oriented Pointer) を通して，特定のオブジェクトを指す。まだ値が決まっていない場合は，その変数セルへのポインタが入る。（図1）

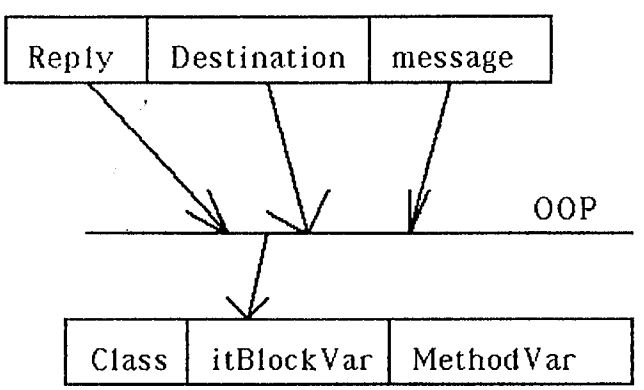

図1　DinnerBellのデータ構造とＯＯＰ

並列実装

並列実装は，現在，Shared Memory BaseとNetwork Base の二つを考えている。どちらの場合も，自分のローカルな領域にあるオブジェクト以外のオブジェクトへのアクセスを減らす必要がある。オブジェクトは，インスタンスなので，各プロセッサに対してローカルであり，通常固定されている。従ってＯＯＰの上位をプロセッサ番号に当てることにより，大域的なＯＯＰをつくることができる。自分の領域内のＯＯＰに対しては，通常のアクセス（代入と，dereference ）　で良いが，自分の領域外のOOP に関しては，特別な工夫が必要である。

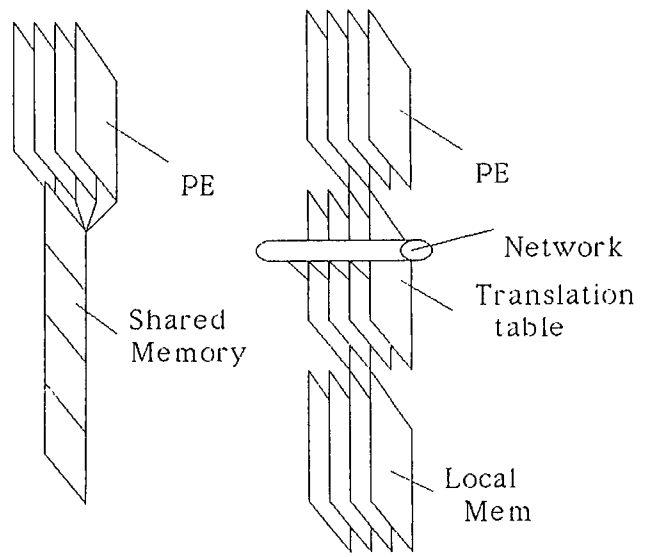

図2　Shared Memory 版の構成

図4　Network 版の構成

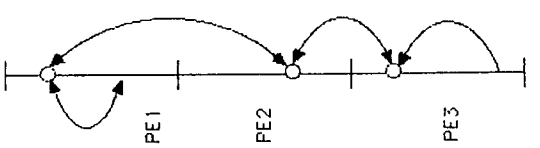

図3　Shared Memory 版での変数のリンク

Parallel Object Oriented Language DinnerBell - Data Management

Shinji KONO, Hidehiko TANAKA
The Tokyo University

①外部を参照する変数と，OOP は，メッセージ送信であるコンテクストを外部に転送することにより生じる。②DinnerBellでは，メッセージの受信は，必ずそのインスタンスのあるプロセッサでおこなわれる。これは，インスタンス変数へのアクセスのローカル性を確保するための実装上の制約である。したがって，③送信先が変数，または，相手のプロセッサのインスタンスでない限り転送は，おこなわれない。これは，シミュレータから得られた結果による制約である。また，④変数のアクセスは，インヘリタンスを通しても，外部の変数に直接アクセスすることはできず，メッセージ送信を通す必要がある。これは，言語上の制約である。これにより，オブジェクトのプロセッサ間にまたがる分配を単純にすることができる。

以下Shared Memory，Network双方のOOP の管理の方法について議論するが，両方とも，プロセッサ間のdereference を除去する所に狙いがある。これにより，dereference によるループ，プロセッサ間通信のオーバヘッドをとりのぞくことができ，OOP の管理も簡単になる。これが可能なのは，DinnerBellの持つ単一代入則による制限と，上記の①—④の制約の為である。

Shared Memory の場合のOOP

この構成の場合は，OOP は，一つの連続したshared Memory 上にとられらる。shared Memory は，プロセッサ単位に分割され，各々が自分の領域をもつ。したがってプロセッサの区別は，物理アドレスによっておこなわれる。コンテクストの転送は，sharedMemory上で直接におこなわれる。（図2）

転送によって生じた他のプロセッサの変数への参照は，生じた時点で，互いを指すように，doubly link を張る。（図3）以後は，このリンクしたローカルな変数にのみアクセスする。この変数へのメッセージ送信だった場合には，そのメッセージ送信をこの変数の待ちキューに接続する。この変数の値が決定した時には，doubly link を通して値が放送される。この時に，変数に接続された待ちコンテクストがアクティブキューに接続され，コンテクストの内容にしたがって，転送される。DinnerBellの単一代入則により，この放送は，衝突することがない。

参照が他のプロセッサのOOP だった場合は，そのポインタをそのままコピーする。

Network の場合のOOP

この場合は，OOP は，外部参照の領域と，自分の領域の二つに分割される。（図4）ローカルなOOP には，自分のプロセッサ番号が付加される。外部参照の領域では，Shared Memory の場合と同様に，doubly link がおこなわれる。（図5）

本方法の特徴

この方法では，記号処理向きであるにもかかわらず，変数のdereference が生じない。全体の実行は，データ駆動となる。オブジェクトへのアクセスは，OOP を通しておこなうため，プロセッサ間のオブジェクトのコピーが可能である。OOP とローカルなオブジェクトの間が間接参照なために，一つのプロセッサ内部での独立したGCが可能である。もちろんローカルなGCでは，プロセッサ間にまたがる環状のごみはとりきれないので，別な方法をもちいる必要がある。この方法は，まだ検討中である。

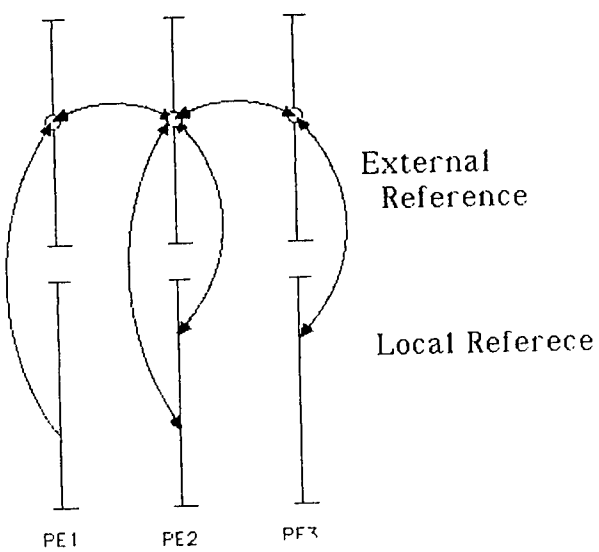

図5　Network 版での変数のリンク

参考文献

〔5〕情報処理学会第30回全国大会，3R-6（1985）