オブジェクト並列プログラミング言語 INADA/MPP の設計

2U－5

福見 幸一　小野 剛　今崎 憲児　牧之内 顕文
（九州大学 工学部 情報工学科）

1. はじめに

現在我々はオブジェクト並列プログラミング言語 INADA/MPP を設計・実装している．この言語は数値計算だけでなく，データベースなどの応用も記述できるような言語を目指している．INADA/MPP は我々がここ数年研究開発してきたオブジェクト指向永続プログラミング言語 INADA[AA93] にオブジェクト並列の概念を導入することで，並列処理の機能を実現している．オブジェクト並列とは，データ並列を拡張した概念で，並列計算機の各プロセッサに分散したオブジェクトが並列にメソッドを実行するという考え方である．

本稿では，INADA/MPP で導入された各種オブジェクトおよび並列実行についての概要と富士通社製並列計算機 AP1000 上での実装方法について述べる．

2. ヒープ管理オブジェクト

INADA/MPP には，プロセッサエレメント (PE) 上のヒープ領域を管理する揮発ヒープ管理オブジェクト (VHMO) というシステム提供のオブジェクトがある．VHMO は次の 2 つの仕事を行っている．

- オブジェクトを生成，消去する．
- 自分が管理しているオブジェクト宛に来たメッセージを解釈し，メソッドを実行させる．

ヒープ領域に生成されたオブジェクトの管理法としては，オブジェクト参照テーブル (ORT) による間接参照方式を採用しており (図 1参照)，並列実行やオブジェクトの永続化に役立つようになっている．

3. 集合オブジェクト

INADA/MPP には INADA と同様に集合オブジェクトが用意されている．集合オブジェクトは任意の PE 上に生成することができ，任意の型のオブジェクトを要素として任意の個数保持することができる．

Design and Impremementation of Object Parallel Programming Language INADA/MPP
Koichi Fukumi, Tsuyoshi Ono, Kenji Imasaki, and Akifumi Makinouchi
Department of Computer Science and Communication Engineering,
Kyushu University

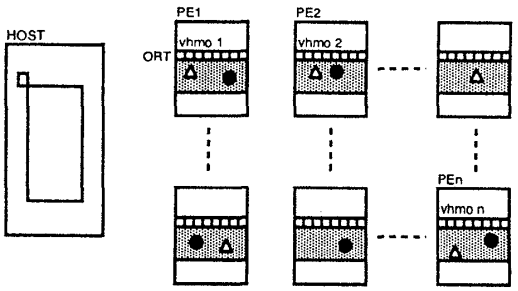

図 1: ヒープオブジェクトの概念図

集合オブジェクトおよび要素オブジェクトを生成するには次のように記述する．

```
Set<Element>* set;
set = new(vhmo[i])Set<Element>();
new(set)Element(argument);
```

これは PE_i のヒープ領域に集合オブジェクト *set* を生成し，*set* の中に要素オブジェクトを 1 つ生成することを意味する．(図 2参照)

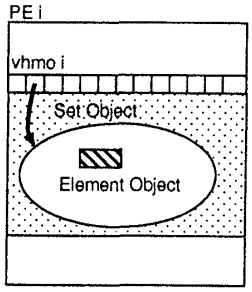

図 2: 集合オブジェクトおよび要素オブジェクトの生成

4. 和集合オブジェクト

要素オブジェクトのメソッドを並列に実行する (= オブジェクト並列) ためには，各 PE 上の集合オブジェクトを 1 つの集合と見る必要がある．INADA/MPP では，このために和集合オブジェクトという概念を導入している．

和集合オブジェクトの生成は以下のように行う．

```
Union<Element>* union;
union = new Union<Element>(processors);
```

上記のように記述すると，各プロセッサ上に *Element*

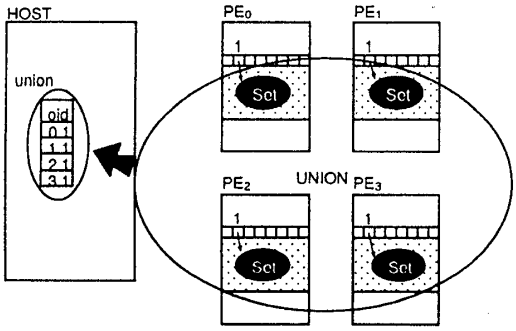

図 3: 和集合オブジェクト

型のオブジェクトを要素に持つ集合オブジェクトが生成され，ホスト上の和集合オブジェクトに各集合オブジェクトの OID(= PE 番号 +ORT 番号) 等の情報が送られる．(図 3参照)

5. 並列実行

INADA/MPP では，和集合の全要素オブジェクトがメソッドを並列に実行することでオブジェクト並列の概念を実現している．このための構文として，for all 文がある．

```
for all x in union do{
    x.method();
}
```

上のように記述すると，和集合の全要素オブジェクト x がメソッド method を並列に実行する．for all 文中では並列に実行するメソッドの呼び出しのみを記述する．

また，要素オブジェクトが別の要素オブジェクト (同じ和集合の要素オブジェクトでも異なる和集合でも構わない) と通信するためには，メソッドの中に *for all* 文を記述すればよい．但し，メソッドの引数として和集合オブジェクトが必要である．

for all 文の実装は次のような手順で行われる．
(図 4参照)

i) ホストが和集合オブジェクトを参照し，各 PE 上の集合オブジェクトにメソッド起動要求を出す．ホストは各 PE でのメソッドの実行が終了するまで待ち状態になる．

ii) 各 PE 上で，VHMO がメッセージを受け取り，指定された集合オブジェクトにメソッドのエミュレーションを実行させる．メソッドに戻り値がある場合は，戻り値をパックしてホストに送る．

iii) 全 PE 上でのメソッドのエミュレーションが終了すると，ホストに制御が戻る．メソッドに戻り値がある場合，セルから送られてきたメッセージをアンパックして集合に挿入する．

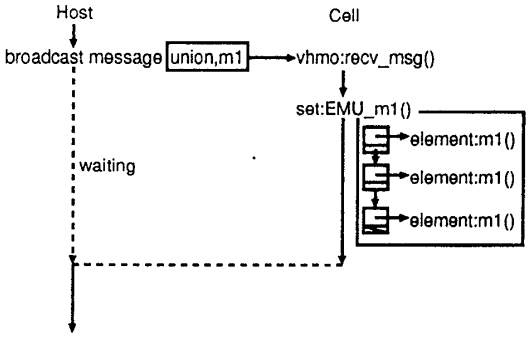

図 4: 並列実行の様子

6. リダクション

for all 文内でリダクションを行うためには，システムが提供するリダクション関数を用いなければならない．例えば，全ての要素オブジェクトの x の和 sum を求める時は *for all* 文内で呼ばれるメソッドに次の様に記述する．

```
Element::method(){
    sum_reduction(&sum,x)
}
```

実装方法としては，まず同じ PE 内に存在する要素オブジェクトに対して和を計算した後，全体の和を AP1000 が提供する関数を用いて求める．

7. おわりに

本稿では，オブジェクト並列プログラミング言語 INADA/MPP の概要と実装方法について述べた．我々は既に様々な例題に対する実装の予備的評価を終えており [AI94]，現在は C++ へのトランスレータについて検討中である．今後は永続オブジェクトも取り扱えるようにするなどの拡張を行い，データベースへの応用を検討していきたいと思う．

参考文献

[AA93] M. Aritsugi and H. Amano:“ View in an Object-Oriented Persistent Programming Language”, *Proc. of the International Symposium on Next Generation Database Systems and Their Applications*, pp.18-25, Sep.1993.

[AI94] 天野, 今崎, 福見, 牧之内：“オブジェクト並列プログラミング言語 INADA/MPP の設計方針と予備的評価について”，文部省重点領域研究第 4 回シンポジウム予稿集,2-214 ～ 2-220, 平 6.3.