



XcalableMPの仕様について

2013/11/01

富士通

岩下 英俊

XMPプログラムの例(at a glance)

- FortranまたはCがベース。指示文(ディレクティブ)拡張
 - ディレクティブだから、逐次プログラムから徐々に並列化できる。

```
module defs
  !$xmp nodes p(4)
  !$xmp template t(YMAX)
  !$xmp distribute t(block) on p

  ineger a(XMAX, YMAX)
  !$xmp align a(*, j) with t(j)
end module

use defs
integer i, j, res
res = 0

!$xmp loop on t(j)
do j = 1, YMAX
  do i = 1, XMAX
    a(i, j) = foo(i, j)
    res = res + a(i, j)
  end do
end do

!$xmp reduction(+: res)
...
```

```
#pragma xmp nodes p(4)
#pragma xmp template t(YMAX)
#pragma xmp distribute t(block) on p

int a[YMAX][XMAX];
#pragma xmp align a[j][*] with t(j+1)

main() {
  int i, j, res;
  res = 0;

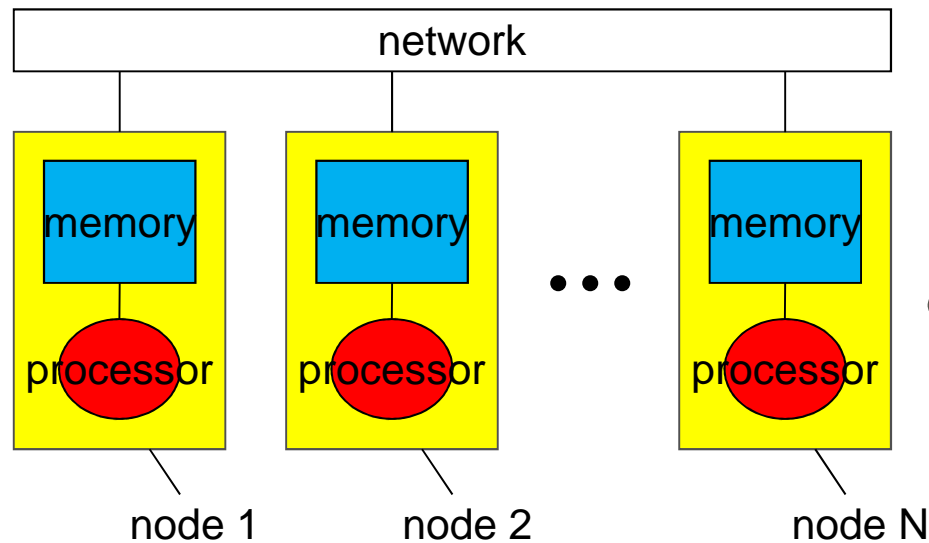
  #pragma xmp loop on t(j+1)
  for (j = 0; j < YMAX; j++) {
    for (i = 0; i < XMAX; i++) {
      a[j][i] = foo(i, j);
      res += a[j][i];
    }
  }

  #pragma xmp reduction(+: res)
  ...
}
```

- XcalableMP (XMP)
 - ハードウェアモデル
 - 実行モデル
 - 言語仕様の体系
- 主な仕様のご紹介、例を交えて
 1. テンプレートの分散
 2. データのマッピング
 3. 計算のマッピング
 4. 通信・同期
- まとめ

XMPのハードウェアモデル

- 複数の“ノード”が、ネットワークで結合されたもの

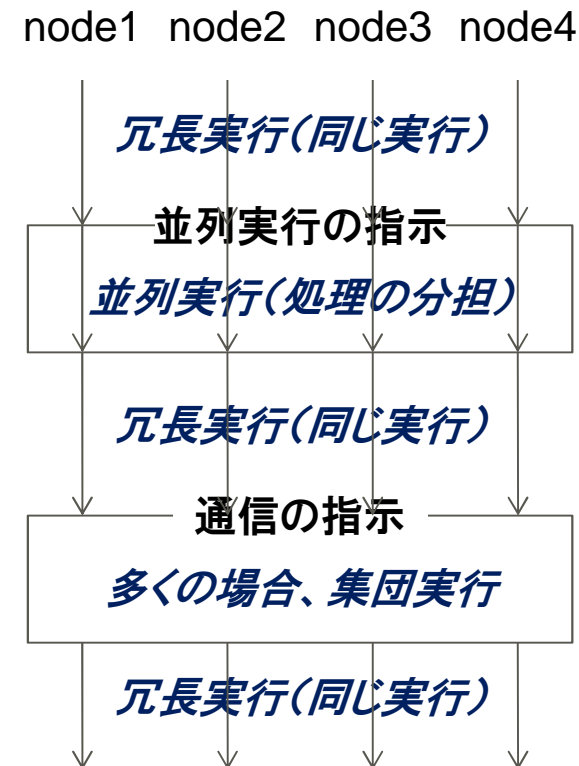


XMP 並列プログラミングは、
1. データをノードにマッピング
2. 計算をノードにマッピング
3. ノード間通信を指示
の3つ

- マルチコア、アクセラレータとのハイブリッド並列も視野に
 - OpenMPとの混在が可能
 - GPUとの連携も検討中

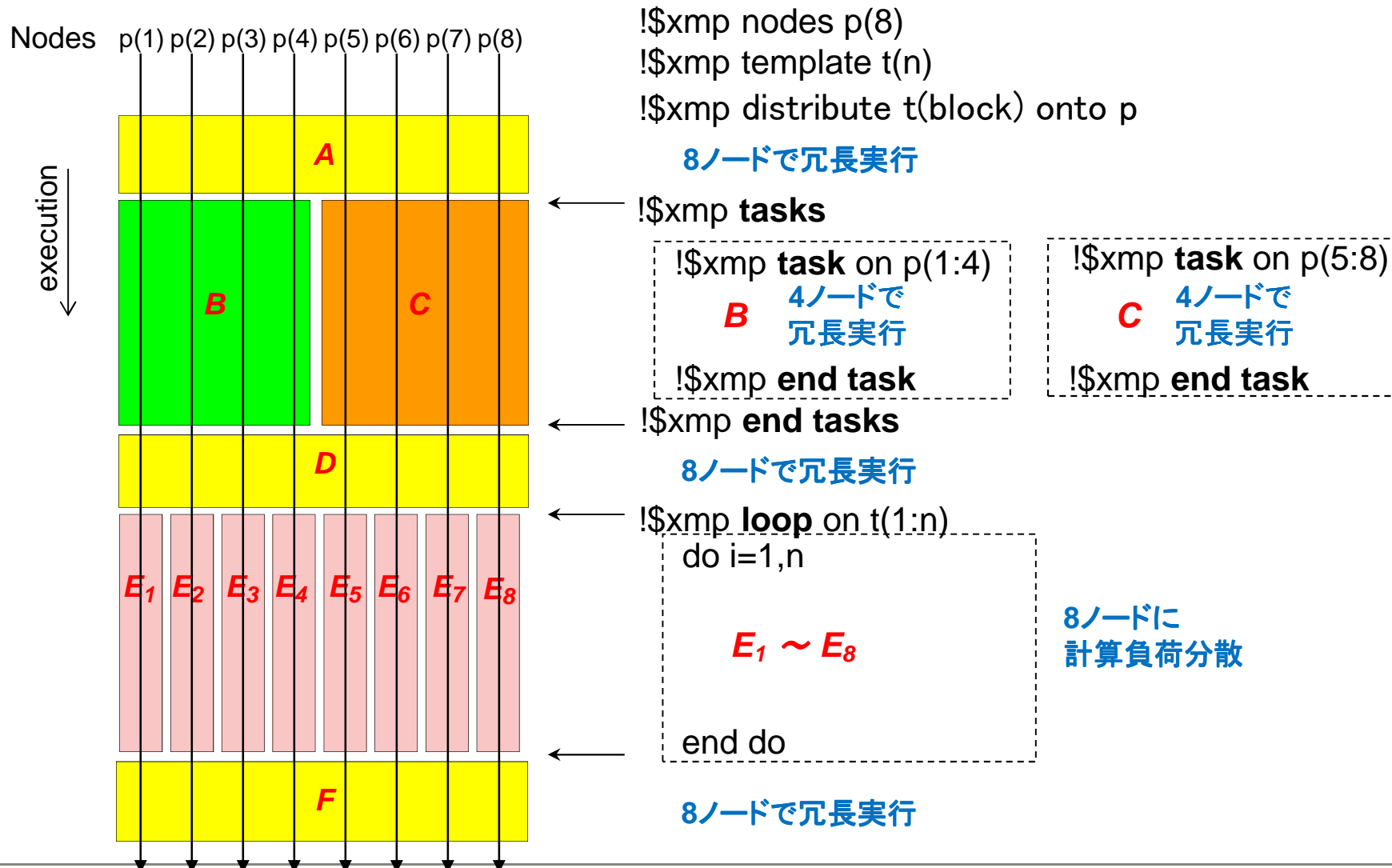
■ 基本的には、SPMD実行モデル

- “冗長実行部”では各ノードが独立に実行。
- HPFと違い、“ローカル変数”が存在する。
- 並列実行の指示があれば、並列実行を行う。
 - LOOP指示構文 ……ループ並列
 - ARRAY指示構文 ……配列代入文の並列実行
 - TASK指示構文 ……タスク並列
- 通信／同期の指示があれば、それを実行する。
 - 多くの場合、集団(collective)実行



冗長実行と並列実行

■ TASKS/TASK指示構文、LOOP指示構文



■ グローバルビュー関連

■ テンプレートの分散

- NODES指示文
- TEMPLATE指示文
- DISTRIBUTE指示文
- TEMPLATE_FIX指示文

■ データマッピング

- ALIGN指示文
- SHADOW指示文

■ 計算マッピング

- LOOP指示構文
- ARRAY指示構文
- TASKS/TASK指示構文

■ 通信と同期

- REFLECT指示文
- GMOVE指示構文
- BARRIER指示文
- REDUCTION指示文
- BCAST指示文

■ ローカルビュー関連

■ 言語拡張(F,Cとも)

- Coarray記法

■ Coarray関連の指示文／組込み

- COARRAY指示文
- IMAGE指示文
- LOCAL_ALIAS指示文(F)
- POST/WAIT指示文
- LOCK/UNLOCK指示文(C)
- Image Index変換組込み手続

■ XMP/C言語拡張

- 部分配列, 配列式, 配列代入
- 記述子問合せ xmp_desc_of

■ 組込み／ライブラリ手続

- 問合せ関数
 - xmp_node_num, xmp_num_nodes等
 - xmp_dist_blocksize等

■ 付録

■ ノード配列の宣言

- 仮想的に配列状に構成される。ノード数は実行時に決めることもできる。

```
!$xmp nodes P( <整数式> , ... ) [ = *  
                                *      = <ノード参照> ]
```

- 左辺Pの添え字の*は、ノード数を実行時に決める。最終次元のみ出現可。
- 右辺が*のとき、Pは全実行ノードであることを意味する。
- 右辺が <ノード参照> のとき、Pは右辺の全ノードの別名として宣言される。

■ 例

```
!$xmp nodes P(6)
```

P(6) (a) (b) (c) (d) (e) (f)

```
!$xmp nodes Q(2,2)=P
```

Q(2,2) (a) (c)
(b) (d)

- テンプレートの形状を宣言する。

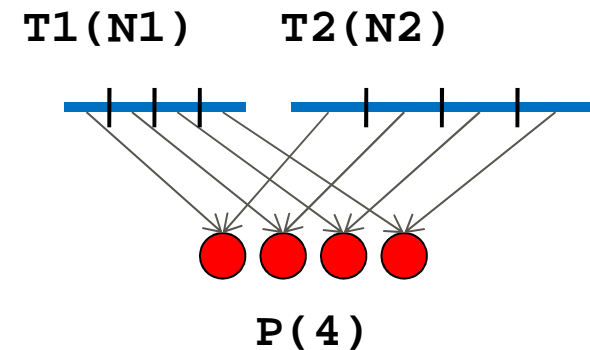
```
!$xmp template T ( <整数式>
                  <整数式> : <整数式>
                  :
                  , ... ) , ...
```

- テンプレートをノード配列に分散する。

```
!$xmp distribute T(<分散種別>, ...) onto P
<分散種別> は, 分散する次元は、block, cyclicなど指定
分散しない次元は *
```

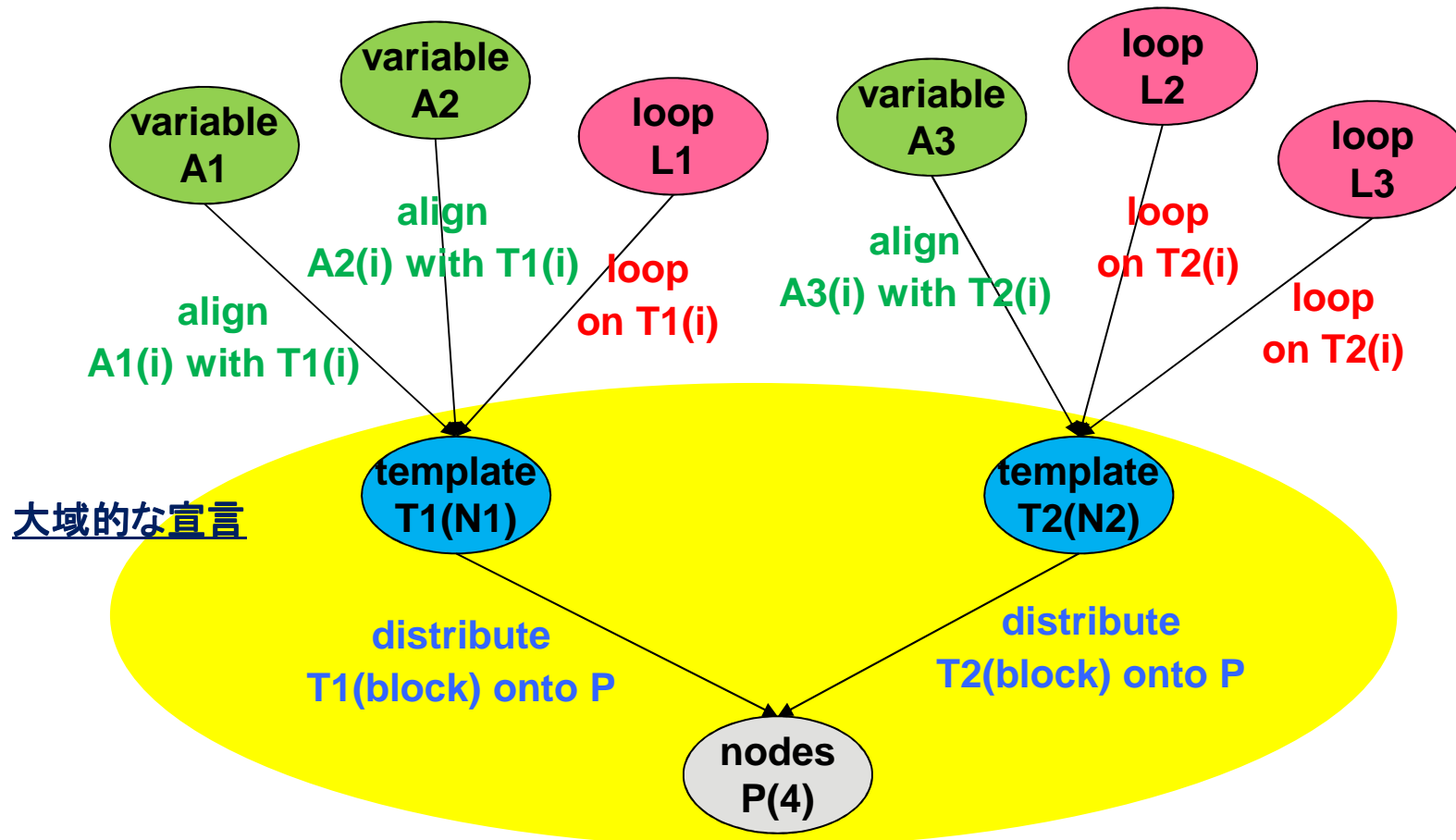
■ 【例】

```
!$xmp nodes P(4)
!$xmp template T1(N1),T2(N2)
!$xmp distribute T1(block) onto P
!$xmp distribute T2(block) onto P
```



テンプレートとは？

- テンプレートは、分散を表現するための、実体のない仮想的な配列
 - 変数は、テンプレートを仲介としてノードにマッピング
 - ループの反復も、テンプレートを介してノードにマッピング
- テンプレートの形状・分散は、プログラム全体の並列化のスキーム



■ グローバルビュー関連

- テンプレートの分散
 - NODES指示文
 - TEMPLATE指示文
 - DISTRIBUTE指示文
 - TEMPLATE_FIX指示文

■ データマッピング

-  ■ ALIGN指示文
- SHADOW指示文

■ 計算マッピング

-  ■ LOOP指示構文
- ARRAY指示構文
- TASKS/TASK指示構文

■ 通信と同期

- REFLECT指示文
- GMOVE指示構文
- BARRIER指示文
- REDUCTION指示文
- BCAST指示文

■ ローカルビュー関連

- 言語拡張(F,Cとも)
 - Coarray記法
- Coarray関連の指示文／組込み
 - COARRAY指示文
 - IMAGE指示文
 - LOCAL_ALIAS指示文(F)
 - POST/WAIT指示文
 - LOCK/UNLOCK指示文(C)
 - Image Index変換組込み手続

■ XMP/C言語拡張

- 部分配列, 配列式, 配列代入
- 記述子問合せ xmp_desc_of

■ 組込み／ライブラリ手続

- 問合せ関数
 - xmp_node_num, xmp_num_nodes等
 - xmp_dist_blocksize等

■ 付録

ALIGN宣言指示文

- 配列変数をテンプレートに整列させる。

```
!$xmp align A ( | <変数名> | , ... ) T ( | <変数名> [ { + | - } <整数式> ] | , ... )  
                *                               *  
                :                               :  
                |                               |
```

■ 【例】

!\$xmp nodes P(4)

!\$xmp template T1(N1),T2(N2)

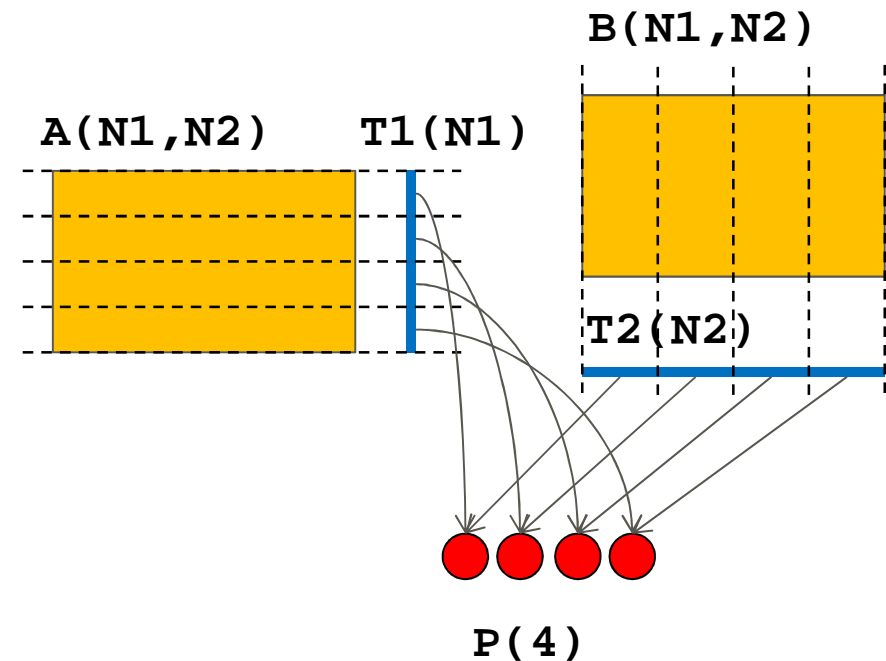
!\$xmp distribute T1(block) onto P

!\$xmp distribute T2(block) onto P

real A(N1,N2), B(N1,N2)

!\$xmp align A(i,*) with T1(i)

!\$xmp align B(*,j) with T2(j)



- DOループ、forループのiterationをテンプレートに整列させる。

- 単一ループに対する指示

```
!$xmp loop on T( <変数名> [ {+|-} <整数式> ] ) [ <reduction節> ]
                *
                :
do i = ...
...
end do
```

- 多重ループに対する指示

```
!$xmp loop (<i1>, ... ,<in>) on T( <変数名> [ {+|-} <整数式> ] , ... ) [ <reduction節> ]
                *
                :
do <i1> = ...
...do <in> = ...
...
end do
end do
```

XMPプログラムの例(再掲)

- ノードとテンプレートは、大域的に宣言するのがよい。
- 共通のテンプレートに対して、データと計算をマッピングしていく。

```
module defs
  !$xmp nodes p(4)
  !$xmp template t(YMAX)
  !$xmp distribute t(block) on p

  ineger a(XMAX, YMAX)
  !$xmp align a(*, j) with t(j)
end module

use defs
integer i, j, res
res = 0

!$xmp loop on t(j)
do j = 1, YMAX
  do i = 1, XMAX
    a(i, j) = foo(i, j)
    res = res + a(i, j)
  end do
end do

!$xmp reduction(+: res)
...
```

```
#pragma xmp nodes p(4)
#pragma xmp template t(YMAX)
#pragma xmp distribute t(block) on p

int a[YMAX][XMAX];
#pragma xmp align a[j][*] with t(j+1)

main() {
  int i, j, res;
  res = 0;

  #pragma xmp loop on t(j+1)
  for (j = 0; j < YMAX; j++) {
    for (i = 0; i < XMAX; i++) {
      a[j][i] = foo(i, j);
      res += a[j][i];
    }
  }

  #pragma xmp reduction(+: res)
  ...
```

テンプレート
の分散

データ
マッピング

計算
マッピング

通信・同期

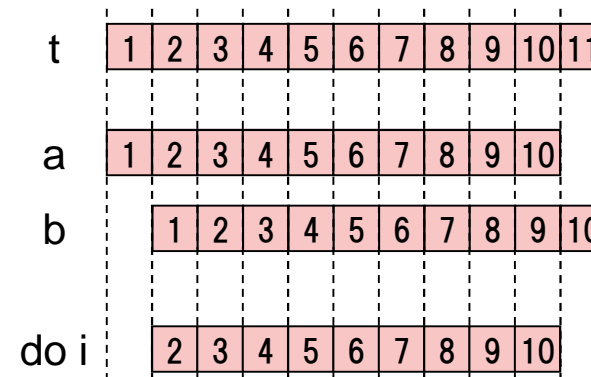
整列(alignment)による通信の削減・削除

- データのALIGN指示と、データが出現するループのLOOP指示で、alignmentを合わせ、通信が起こらないようにする。

```
!$xmp nodes p(2)
!$xmp template t(11)
!$xmp distribute t(block) onto p

      real a(10),b(10),c(10)
!$xmp align a(i) with t(i)
!$xmp align b(i) with t(i+1)

!$xmp loop on t(i)
  do i=2,10
    a(i)=a(i)+1.0
    b(i-1)=a(i)
  end do
```



整列していれば通信は発生しない。

- 分散種別(block, cyclic, ...)には依存しない！

■ グローバルビュー関連

- テンプレートの分散
 - NODES指示文
 - TEMPLATE指示文
 - DISTRIBUTE指示文
 - TEMPLATE_FIX指示文

■ データマッピング

- ALIGN指示文
- SHADOW指示文



■ 計算マッピング

- LOOP指示構文
- ARRAY指示構文
- TASKS/TASK指示構文

■ 通信と同期

- REFLECT指示文
- GMOVE指示構文
- BARRIER指示文
- REDUCTION指示文
- BCAST指示文



■ ローカルビュー関連

- 言語拡張(F,Cとも)
 - Coarray記法
- Coarray関連の指示文／組込み
 - COARRAY指示文
 - IMAGE指示文
 - LOCAL_ALIAS指示文(F)
 - POST/WAIT指示文
 - LOCK/UNLOCK指示文(C)
 - Image Index変換組込み手続

■ XMP/C言語拡張

- 部分配列, 配列式, 配列代入
- 記述子問合せ xmp_desc_of

■ 組込み／ライブラリ手続

- 問合せ関数
 - xmp_node_num, xmp_num_nodes等
 - xmp_dist_blocksize等

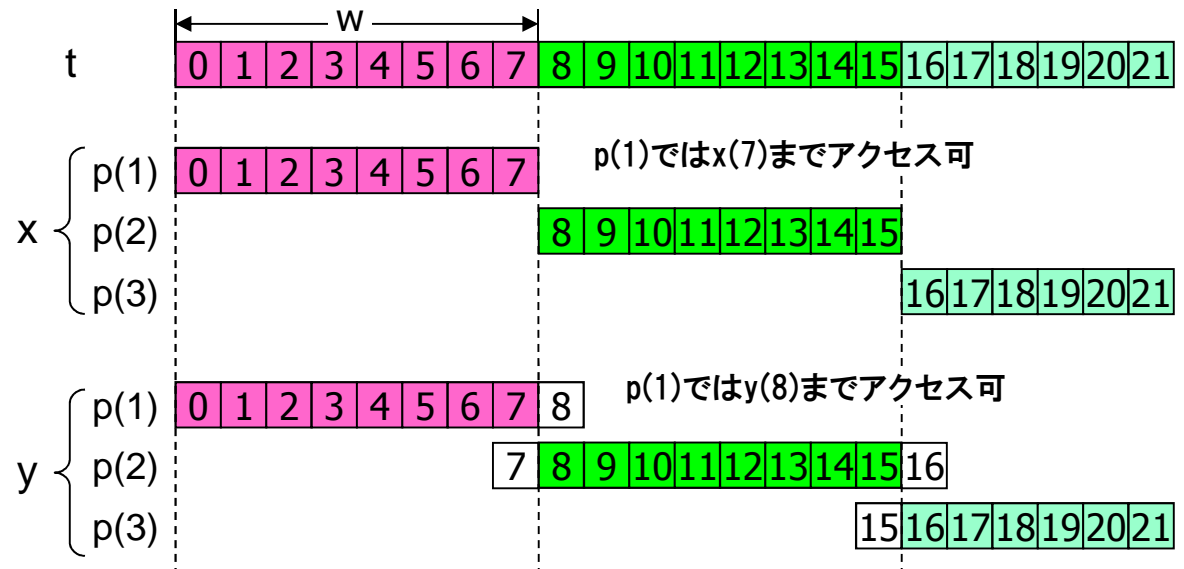
■ 付録

SHADOW宣言指示文

- block分散または不均等block分散で、分割範囲に「袖」を加える。
 - 通常 of 分割範囲に加えて、袖の領域までローカルにアクセスできる。
 - 隣接ノードとの値の一貫性は、利用者の責任。

```
!$xmp shadow A ( <整数式> , ... ) , ...  
                <整数式> : <整数式>  
                :
```

```
!$xmp nodes p(3)  
!$xmp template t(0:21)  
!$xmp distribute t(W) onto p  
  
dimension x(0:21)  
!$xmp align x(i) with t(i)  
  
dimension y(0:21)  
!$xmp align y(i) with t(i)  
!$xmp shadow y(1)
```



REFLECT実行指示文

- すべての「袖」領域に、それぞれ隣接ノードの値をコピーする。

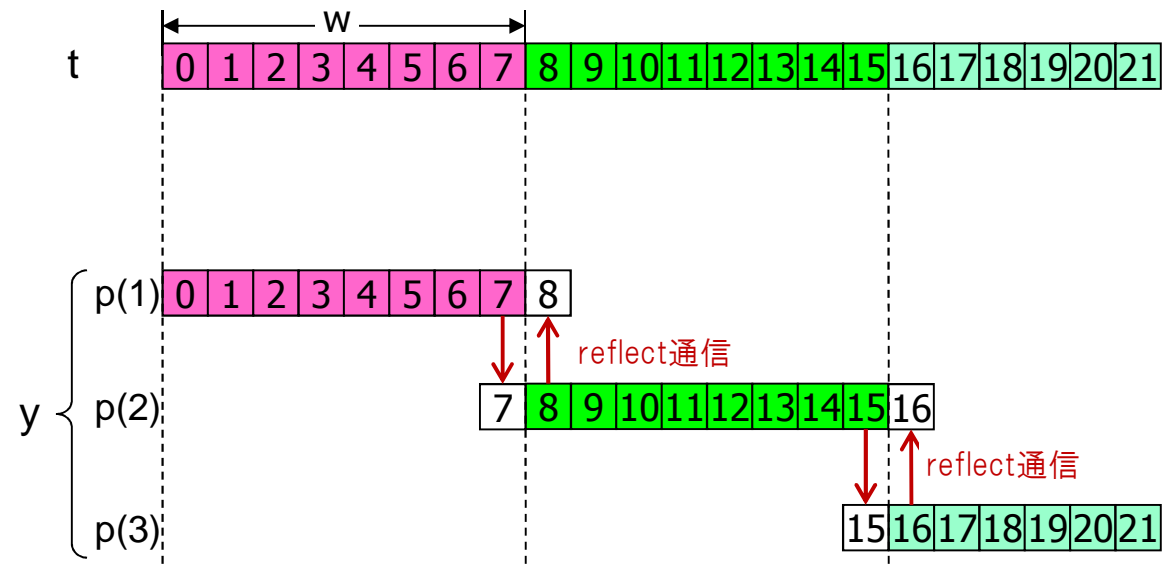
```
!$xmp reflect (A) [ <width節> ] [ <async節> ]
```

```
!$xmp nodes p(3)  
!$xmp template t(0:21)  
!$xmp distribute t(W) onto p
```

```
dimension x(0:21)  
!$xmp align x(i) with t(i)
```

```
dimension y(0:21)  
!$xmp align y(i) with t(i)  
!$xmp shadow y(1)
```

```
!$xmp reflect (y)  
!$xmp loop t(i)  
do i = 1, 20  
  x(i) = 0.25 * (y(i-1) + 2*y(i) + y(i+1))  
end do
```



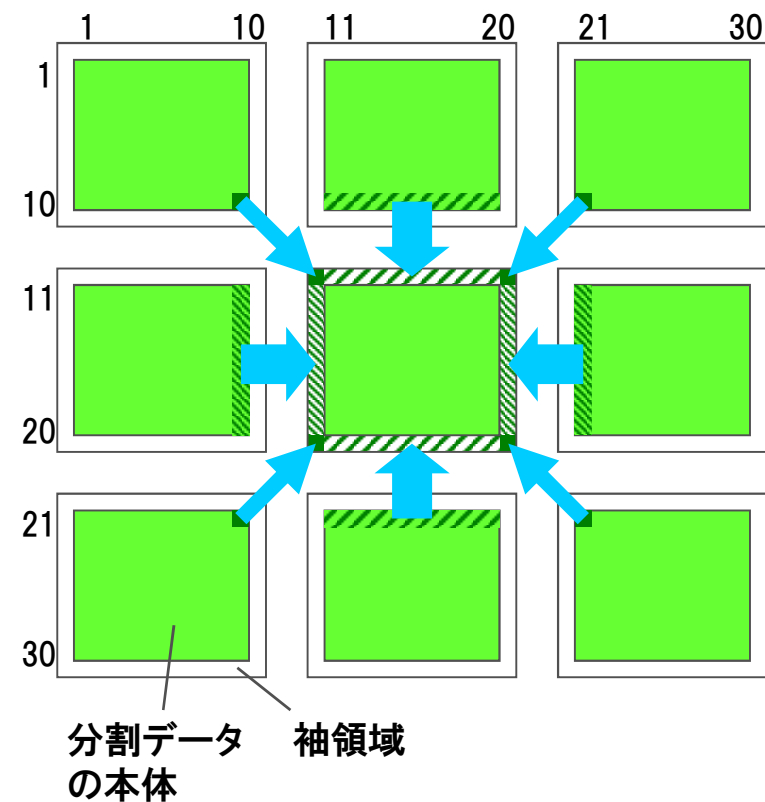
SHADOW/REFLECTの例

- 2次元分割、隣接8ノードと袖通信がある例
 - 通信の記述はREFLECT指示文1つだけ

```
!$xmp nodes p(3,3)
!$xmp template t(30,30)
!$xmp distribute t(block,block) onto p
  real a(30,30)
!$xmp align a(i,j) with t(i,j)
!$xmp shadow a(1,1)

! *** 袖領域に値を設定 ***

!$xmp reflect a
!$xmp loop (i,j) on p(i,j)
  do j=1,100
    do i=1,100
      a(i,j) = a(i-1, j-1) + a(i, j-1) + a(i+1, j-1) + &
               a(i-1, j ) + a(i, j ) + a(i+1, j ) + &
               a(i-1, j+1) + a(i, j+1) + a(i+1, j+1)
    enddo
  enddo
```



■ グローバルビュー関連

- テンプレートの分散
 - NODES指示文
 - TEMPLATE指示文
 - DISTRIBUTE指示文
 - TEMPLATE_FIX指示文

■ データマッピング

- ALIGN指示文
- SHADOW指示文

■ 計算マッピング

- LOOP指示構文
- ARRAY指示構文
- TASKS/TASK指示構文

■ 通信と同期

- REFLECT指示文
- GMOVE指示構文
- BARRIER指示文
- REDUCTION指示文
- BCAST指示文

■ ローカルビュー関連

- 言語拡張(F,Cとも)
 - Coarray記法
- Coarray関連の指示文／組込み
 - COARRAY指示文
 - IMAGE指示文
 - LOCAL_ALIAS指示文(F)
 - POST/WAIT指示文
 - LOCK/UNLOCK指示文(C)
 - Image Index変換組込み手続

■ XMP/C言語拡張

- 部分配列, 配列式, 配列代入
- 記述子問合せ xmp_desc_of

■ 組込み／ライブラリ手続

- 問合せ関数
 - xmp_node_num, xmp_num_nodes等
 - xmp_dist_blocksize等

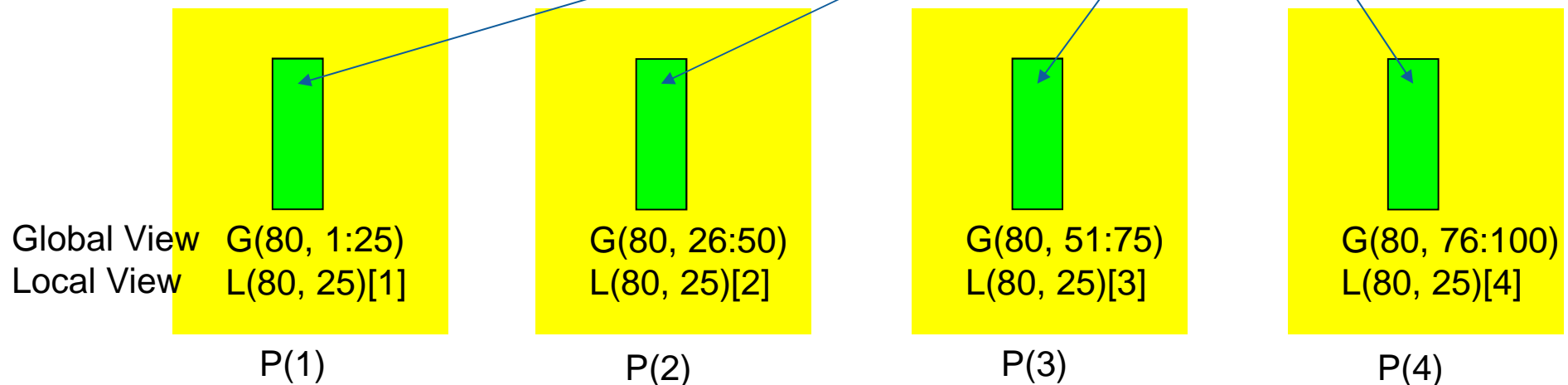
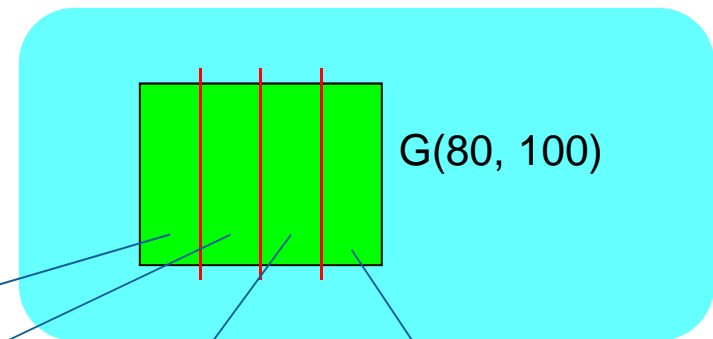
■ 付録

LOCAL_ALIAS宣言指示文

- グローバル配列Gに、ローカルビューの別名Lを付ける

```
!$xmp nodes P(4)  
  
!$xmp template :: T(100)  
!$xmp distribute (block) onto P :: T(100)  
  
  real :: G(80, 100)  
!$xmp align with T :: G  
  
  real :: L(80, 25)  
!$xmp local_alias L to G
```

仮想グローバル空間



■ XcalableMPの特徴

- テンプレートが活躍する言語仕様体系
 - HPFと違い、計算マッピングも通信も手動だから
 - テンプレートは、プログラム全体の並列化の戦略(スキーム)を表現する
 - データもループiterationも、テンプレートに整列させる

- 通信はREFLECTが最重要
 - XMPが確実に性能を出せるのは、ステンシル計算
 - 他の通信パターンでは何が重要だろうか

- グローバルビューとローカルビューとの連携
 - グローバルからローカルへのalias(名前の呼び分け)が提供されている
 - 逆のaliasも必要。要件を具体化したい

■ XcalableMPの今後

- ハイブリッドへ(アクセラレータ、MPI、OpenMP)
- トポロジ対応、片側通信
- まずはOMNI XMPの実装に協力し、利用者を増やしたい