



高性能・高生産プログラミングのための
並列言語

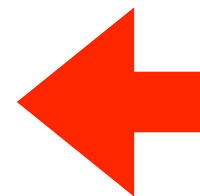
XcalableMP



中尾昌広
筑波大学計算科学研究センター
研究員



- 素粒子物理学研究部門
- 宇宙・原子核物理研究部門
 - 宇宙物理分野
 - 原子核分野
- 量子物性研究部門
- 生命科学研究部門
 - 生命機能情報分野
 - 分子進化的分野
- 地球環境研究部門
- 高性能計算システム研究部門
- 計算情報学研究部門
 - データ基盤分野
 - 計算メディア分野



- 私の研究について
 - 並列計算機を簡易に扱うコンパイラの作成および評価
 - ◆ 計算科学と計算機科学の架け橋に
 - 次世代のHPCプログラミング言語の作成
 - ◆ ノード間並列 + ノード内並列 + アクセラレータの混合プログラミング
- 本発表の目的
 - 筑波大学計算科学研究センターで開発している並列言語 XcalableMP (XMP) の紹介

XcalableMP (XMP)



● 背景

- 並列プログラムはMPIライブラリ+C/Fortranが主流
 - ◆ 生産性が悪いため、並列化のコストが高い

● 目標

- 基本は指示文+C/Fortranで並列化
- 並列システムでのアプリケーション開発を容易にし、生産性を向上させる並列プログラミング言語の設計および開発

現在の並列プログラミング

```
int array[MAX];

main(int argc, char **argv){
    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    MPI_Comm_size(MPI_COMM_WORLD, &size);
    dx = MAX/size;
    llimit = rank * dx;
    if(rank != (size - 1)) ulimit = llimit + dx;
    else ulimit = MAX;

    temp_res = 0;
    for(i=llimit; i < ulimit; i++){
        array[i] = func(i);
        temp_res += array[i];
    }

    MPI_Allreduce(&temp_res, &res, 1, MPI_INT, MPI_SUM, ...);
    MPI_Finalize();
}
```

ソースが長くなるから、メンテナンスしにくいよう



XcalableMP 言語

```
int array[MAX];
#pragma xmp template t(0:MAX-1)
#pragma xmp nodes p(*)
#pragma xmp distribute t(block) onto p
#pragma xmp align array[i] with t(i)

main(){
    #pragma xmp loop on t(i) reduction (+:res)
    for(i = 0; i < MAX; i++){
        array[i] = func(i);
        res += array[i];
    }
}
```

簡単にかけて、読みやすい！！



開発体制について

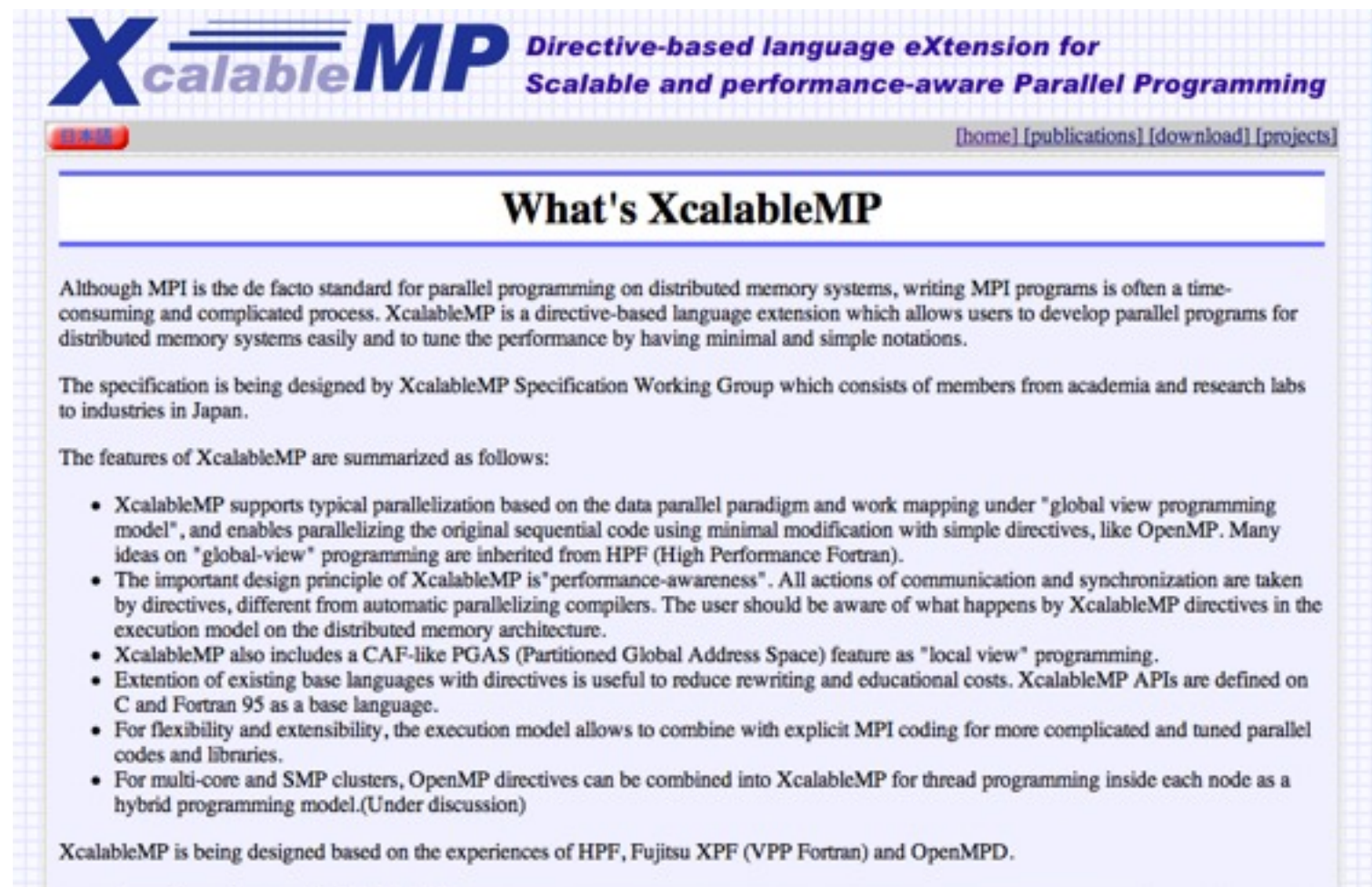


- 大学・企業・研究所のメンバで構成される、次世代並列プログラミング言語検討委員会を作成
 - コミュニティの経験（特にHigh Performance Fortran）と意向を取り入れた仕様検討
 - 開発後の普及体制までを考慮（実用化を重視）
 - 標準化を目指して、world-wide communityに提案
- メンバー
 - 大学：筑波大、東大、京都大、九州大
 - 企業：富士通、NEC、日立
 - 研究所：理研、NIFS、JAXA、JAMSTEC/ES

XMP仕様とコンパイラ



- 仕様書version 1.0をリリース (2011/11/14)
- 筑波大からOmni XMP Compilerがリリース
(オープンソース・現在はC言語のみの対応)
- マニュアル・チュートリアルも公開



<http://www.xcalablemp.org>

プログラム例 (1/3)



```
int a[100];
#pragma xmp nodes p(4)
#pragma xmp template t(0:99)
#pragma xmp distribute t(block) onto p
#pragma xmp align a[i] with t(i)

main(){
    int i, res = 0;
    #pragma xmp loop on t(i) reduction(+:res)
    for(i = 0; i < 100; i++){
        a[i] = func(i);
        res += a[i];
    }
}
```

**data
distribution**

**work mapping
& reduction**

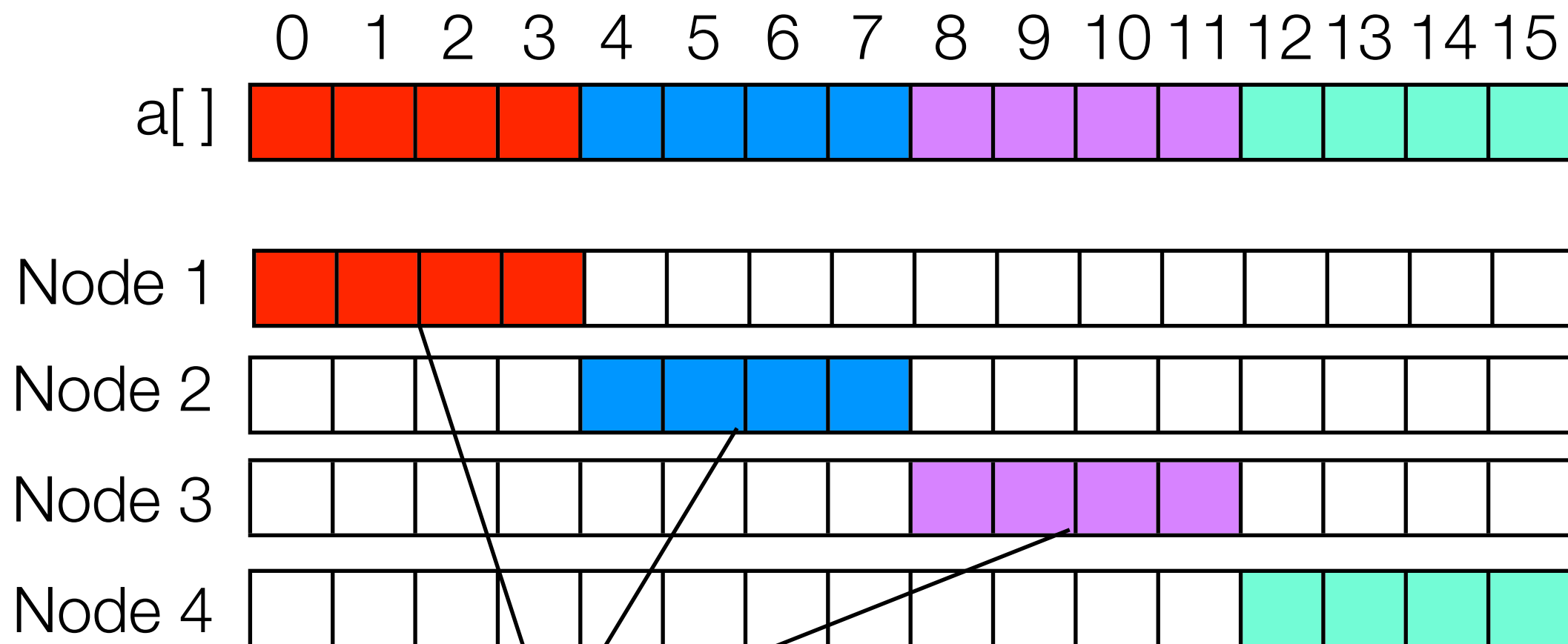
逐次コードに指示文を追加することで並列化が可能
もちろん、並列アルゴリズムを最初から記述も可能

Data Distribution



- 指示文でデータ分散を定義

```
#pragma xmp nodes p(4)  
#pragma xmp template t(0:15)  
#pragma xmp distribute t(block) on p  
#pragma xmp align a[i] with t(i)
```



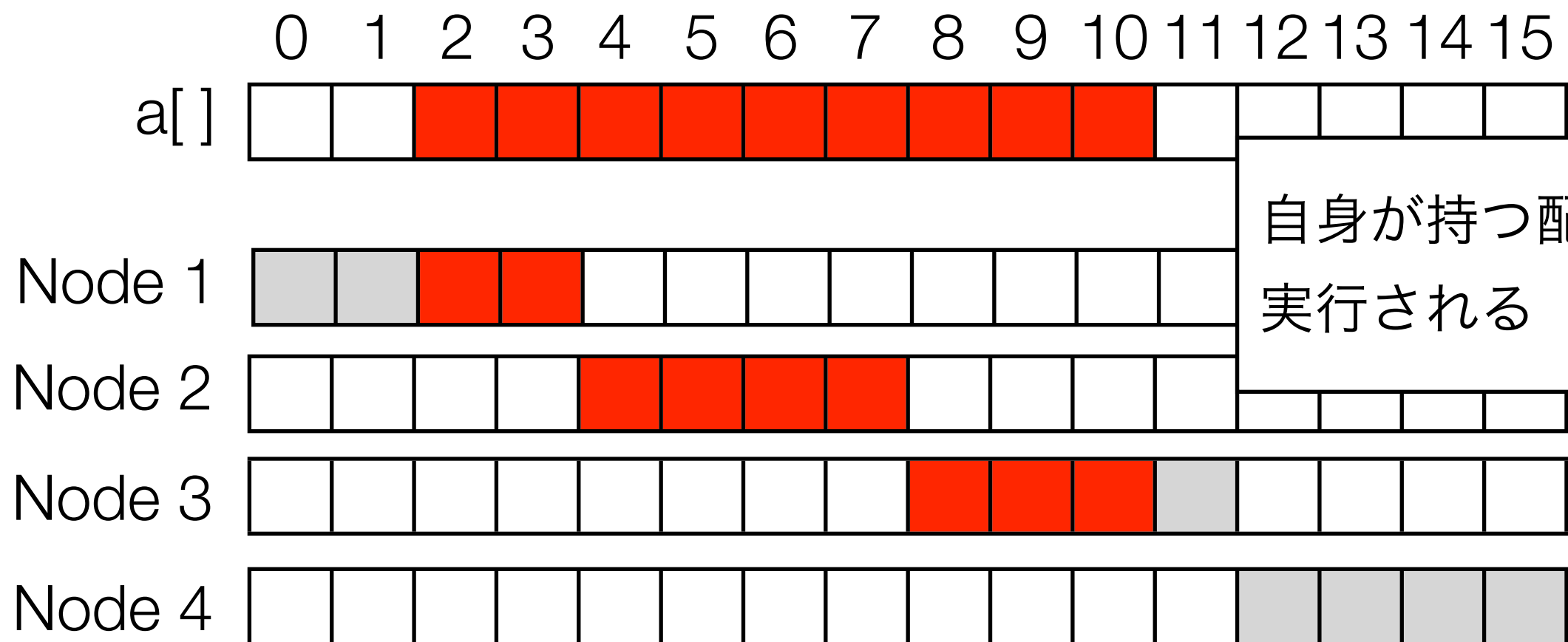
分散配列

Parallel Execution of loop **X_{calable}MP**

- ループ指示文で並列処理

```
#pragma xmp loop on t(i)  
for(i=2;i<=10;i++){...}
```

```
#pragma xmp nodes p(4)  
#pragma xmp template t(0:15)  
#pragma xmp distribute t(block) on p  
#pragma xmp align a[i] with t(i)
```



自身が持つ配列のみ
実行される

各ノードで赤色の要素のみが並列に実行される

Gmove (Global View Communication)

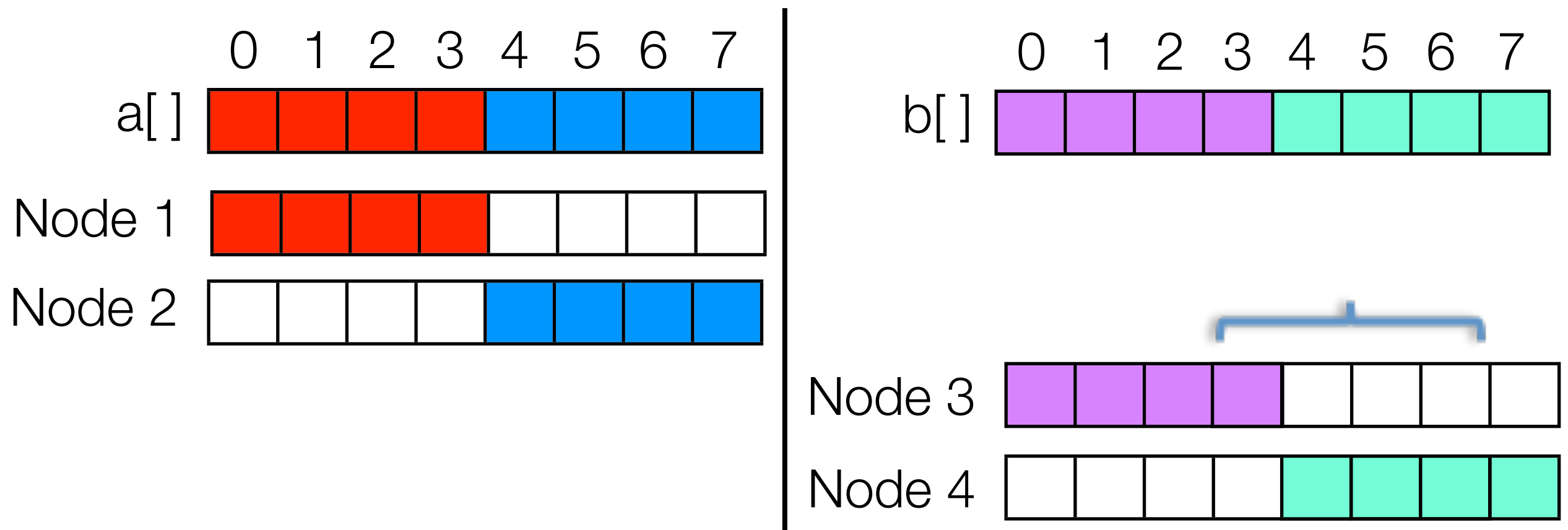
- 分散配列間 or ローカル配列 - 分散配列 間

- C言語を拡張し、Array Sectionを採用

- ユーザは、どのノードがどの要素を持つかを意識する必要がない

base
length

```
#pragma xmp gmove  
a[2:4] = b[3:4];
```



その他の指示文（一部）



以下のような指示文で通信を記述

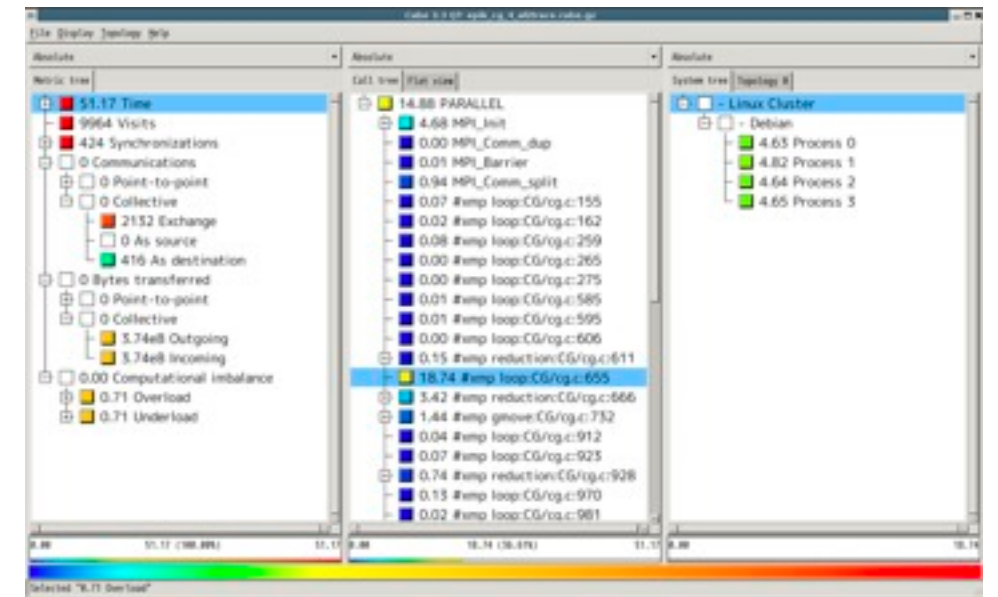
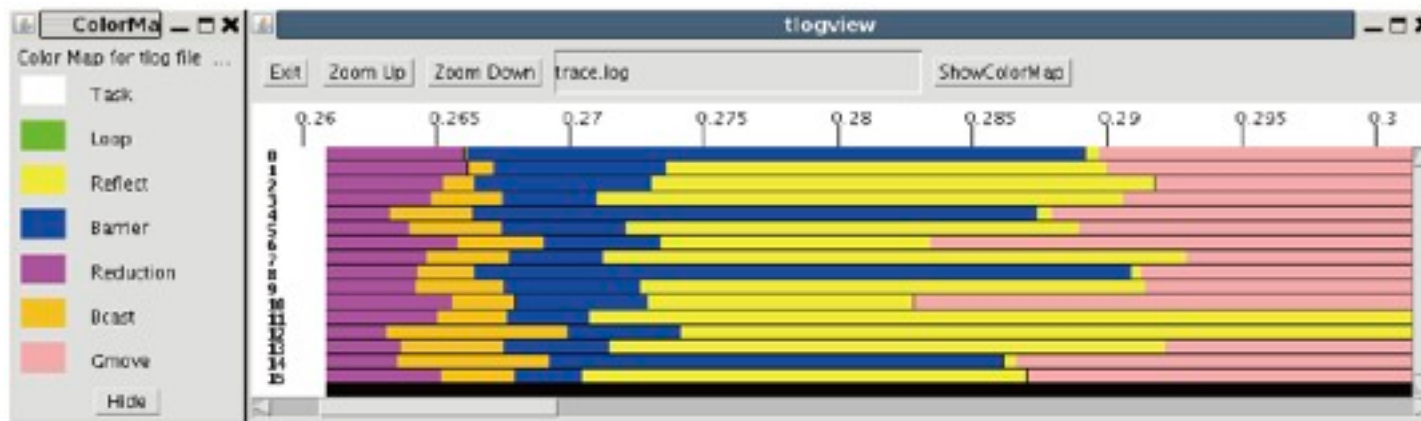
- #pragma xmp bcast (*var*) on node
 - データ（*var*）のブロードキャスト
- #pragma xmp barrier
 - バリア同期
- #pragma xmp reduction (+: *var*)
 - データ（*var*）の集約計算

他にも、片側通信の簡易記法やステンシル計算を簡易に記述できる指示文など

その他の情報



- プロファイリングツールインタフェース
 - 性能最適化ツールScalasca
 - 軽量プロファイリングツールtlog



- 開発中の機能
 - MPIライブラリインタフェース
 - XMP-IO（並列I/O）
 - スレッド並列
 - XMP-dev（アクセラレータ対応）

まとめと今後の活動



- XMP Projectの目的・目標
 - 並列環境のプログラミング環境の改善
 - 仕様書 ver. 1.0および Omni XMP Cコンパイラ
 - ◆ <http://www.xcalablemp.org>
 - XMPの開発体制
 - XMPの文法説明
 - XMPの開発中の機能
- 普及に向けて
 - 既存の大規模アプリケーション（例：気象など）の開発を行い、実用的な大規模アプリケーションが開発可能であることを示す
 - 京コンピュータで検証
 - Omni XMP Fortranコンパイラの作成