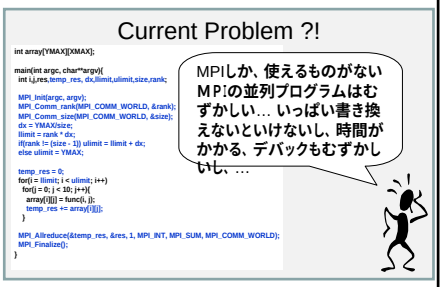
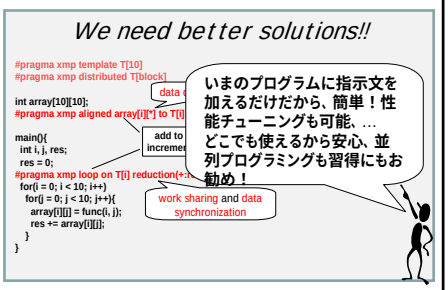
e-science XcalableMP プロジェクト

- 現状と課題
  - 並列プログラムの大半はMPI通信ライブラリによるプログラミング
    - 生産性が悪く、並列化のためのコストが高い。
  - 並列プログラミングの教育のための簡便で標準的な言語がない (MPIでの教育にとどまっている)
  - 研究室のPCクラスタから、センター、ペタコンまでに到るスケーラブルかつポータブルな並列プログラミング言語が求められている
- 目標
  - 既存言語を指示文により拡張し、これからの大規模並列システム (分散メモリシステムと共有メモリノード) でのプログラミングを助け、生産性を向上させる並列プログラミング言語を設計・開発する。
  - 標準化をすることを前提に、ユーザのわかりやすさを第一にどこでも使えるということを重視し、開発ならびに普及活動を進める。





“Petascale” 並列プログラミング WG

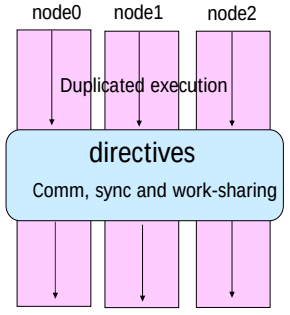
- 目的
  - “標準的な”並列プログラミングのためのペタスケールを目指した並列プログラミング言語の仕様を策定する
  - “標準化”を目指して、“world-wide” communityに提案する。
- Members
  - Academia: M. Sato, T. Boku (compiler and system, U. Tsukuba), K. Nakajima (app. and programming, U. Tokyo), Nanri (system, Kyusyu U.), Okabe (HPF, Kyoto U.)
  - Research Lab.: Watanabe and Yokokawa (RIKEN), Sakagami (app. and HPF, NIFS), Matsuo (app., JAXA), Uehara (app., JAMSTEC/ES)
  - Industries: Iwashita and Hotta (HPF and XPFortran, Fujitsu), Murai and Seo (HPF, NEC), Anzaki and Negishi (Hitachi)
- 2007年12月にkick-off, 現在、e-scienceプロジェクトの並列プログラミング検討委員会に移行
- メーカからのコメント・要望 (活動開始時)
  - 科学技術アプリケーション向けだけでなく、組み込みのマルチコアでも使えるようなものにするべき。
  - 国内の標準化だけでなく、world-wideな標準を目指す戦略を持つべき
  - 新しいものをつくるのであれば、既存の並列言語 (HPF やXPFortranなど) からの移行パスを考えてほしい

並列プログラミング言語：何が問題だったのか

- HPFの教訓 (by 坂上@核融合研、村井@NEC)
  - 並列性とデータ分散を書いて、自動的に生成するという方針は理想的だったが、必ずしも性能は上がらなかった。期待が大きかった分、失望も大きかった。
  - ベース言語としたF90が未熟だった。Fortranだけだった。
  - 必要な情報をユーザで指示文で補ってもらうという方針だったが、どこをどうすれば最適なコードになるかが明らかでなかった。
  - 自動であるがために、通信がどこでおこっているのか、どうやってチューニングすればいいのか、ユーザに手段が与えられていなかった。
  - 完全性を求めるあまり不必要な仕様があり、実装の障害になっていた。
  - レファレンス実装が不在。教育が考慮されていない。
- 90年代の並列プログラミング言語
  - 多くはプログラミング言語の研究が主で、実際のアプリで使われることが少なかった。
  - 組織的な普及活動、標準化、教育活動がない。

XcalableMPの概要

XcalableMP : directive-based language eXtension
for Scalable and performance-tunable Parallel Programming



- 基本の実行モデルはSPMDモデル。
  同時に、同じプログラムを各ノードで実行
- 既存の言語をベースに、指示文で言語を拡張。
  書き換えと教育のコストを低減
- “global view”では、データ並列での典型的な並列化をサポート。OpenMPのように簡単な指示文を使って、逐次コードをできるだけ書き換えずに並列化できるようにする。
- “local view”では、CAF-like PGAS (Partitioned Global Address Space) 機能をサポート。
- 通信と同期操作については、明示的に行う。すべての通信・同期は指示文で行われることとし、性能チューニングをやりやすくする。
- 柔軟性と拡張性のために、実行モデルをMPIが混在できるようにし、チューニングされたMPIライブラリを利用可能にする。
- マルチコアあるいはSMPクラスタ向けには、OpenMPとの混在のプログラミングを可能とし、ノード内はOpenMPで記述。

Code Example

```
int array[YMAX][XMAX];

#pragma xmp nodes p(4)
#pragma xmp template t[YMAX]
#pragma xmp distribute t[block] on p
#pragma xmp align array[i][*] to t[i]

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

#pragma xmp loop on t[i]  reduction(+:res)
  for(i = 0; i < 10; i++)
    for(j = 0; j < 10; j++){
      array[i][j] = func(i, j);
      res += array[i][j];
    }
}
```

data distribution

add to the serial code : incremental parallelization

work sharing and data synchronization

The same code written in MPI

```
int array[YMAX][XMAX];

main(int argc, char**argv){
  int i,j,res,temp_res, dx,llimit,ulimit,size,rank;

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

  temp_res = 0;
  for(i = llimit; i < ulimit; i++)
    for(j = 0; j < 10; j++){
      array[i][j] = func(i, j);
      temp_res += array[i][j];
    }

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

XcalableMP コード例 (NPB CG, global view)

```
#pragma xmp nodes p[NPROCS]
#pragma xmp template t[N]
#pragma xmp distributed t[block] on p
...
#pragma xmp aligned [i] to t[i] :: x,z,p,q,r,w
#pragma xmp shadow [*] :: x,z,p,q,r,w
...
```

ノードの形状の定義

Templateの定義とデータ分散を定義

データの分散は、templateにalignデータの同期のためのshadowを定義、この場合はfull shadow

Work sharing ループの分散

データの同期

```
/* code fragment from conj_grad in NPB CG */
sum = 0.0;
#pragma xmp loop on t[j] reduction(+:sum)
    for (j = 1; j <= lastcol-firstcol+1; j++) {
        sum = sum + r[j]*r[j];
    }
    rho = sum;
for (cgit = 1; cgit <= cgitmax; cgit++) {
#pragma xmp reflect p
#pragma xmp loop on t[j]
        for (j = 1; j <= lastrow-firstrow+1; j++) {
            sum = 0.0;
            for (k = rowstr[j]; k <= rowstr[j+1]-1; k+
                sum = sum + a[k]*p[colidx[k]];
            }
            w[j] = sum;
        }
#pragma xmp loop on t[j]
        for (j = 1; j <= lastcol-firstcol+1; j++) {
            q[j] = w[j];
        }
```

XcalableMP コード例 (laplace, global view)

```
#pragma xmp nodes p[NPROCS]
#pragma xmp template t[1:N]
#pragma xmp distribute t[block] on p

double u[XSIZE+2][YSIZE+2],
       uu[XSIZE+2][YSIZE+2];
#pragma xmp aligned u[i][*] to t[i]
#pragma xmp aligned uu[i][*] to t[i]
#pragma xmp shadow uu[1:1]

lap_main()
{
 int x,y,k;
 double sum;
    …
```

ノードの形状の定義

Templateの定義とデータ分散を定義

データの分散は、templateにalignデータの同期のためのshadowを定義、この場合はshadowは袖領域

Work sharing ループの分散

データの同期

```
for(k = 0; k < NITER; k++){
        /* old <- new */
#pragma xmp loop on t[x]
        for(x = 1; x <= XSIZE; x++)
          for(y = 1; y <= YSIZE; y++)
            uu[x][y] = u[x][y];
#pragma xmp reflect uu
#pragma xmp loop on t[x]
        for(x = 1; x <= XSIZE; x++)
          for(y = 1; y <= YSIZE; y++)
            u[x][y] = (uu[x-1][y] + uu[x+1][y
                 uu[x][y-1] + uu[x][y+1])/4.0
    }
    /* check sum */
    sum = 0.0;
#pragma xmp loop on t[x] reduction(+:sum)
    for(x = 1; x <= XSIZE; x++)
        for(y = 1; y <= YSIZE; y++)
          sum += (uu[x][y]-u[x][y]);
#pragma xmp block on master
    printf("sum = %g¥n",sum);
}
```

XcalableMP (local view)

- XcalableMPでは、何も指示をしなければ単なるSPMDのプログラム
- Local viewでは、ノード内のオペレーションを中心に操作。PGAS (Partitioned Global Address Space) 機能により、他ノードのデータを参照できるようにして最適化を支援

Array sectionの導入

- C言語では、array section(部分配列) 記述ができるように言語拡張
- Coarray機能(CAF)を導入、remote memory access機能 (one-sided通信) をサポート

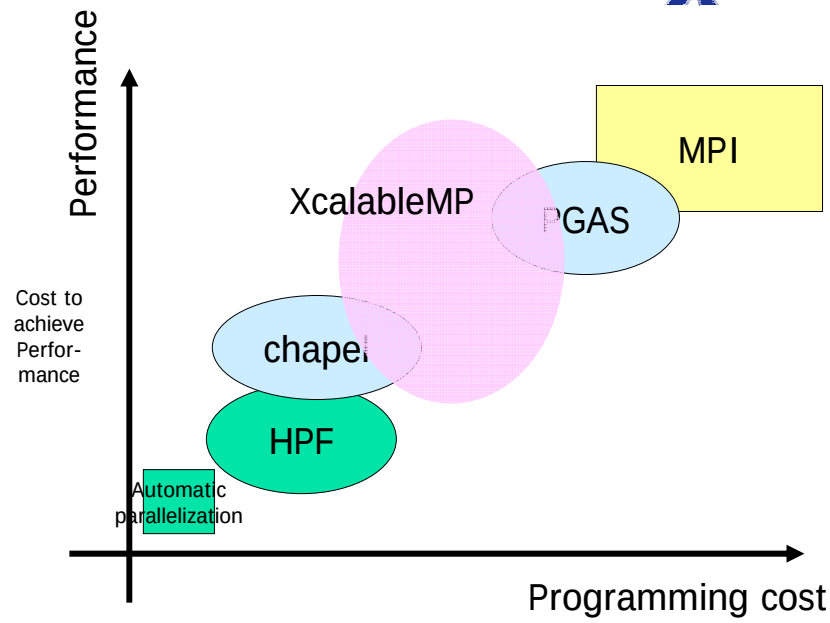
```
int A[10]:
int B[5];

A[4:9] = B[0:4];
```

- Global viewからlocal viewへの変換をサポ

```
int A[10], B[10];
#pragma xmp coarray [*]: A, B
…
A[:] = B[:][10];
```

Position of XcalableMP



おわりに

- XcalableMPの目的・目標
  - 超並列マシンの並列プログラミングにはいろいろな課題はあるが、… 生産性(productivity)をあげることが重要
  - 「MPIよりもましなプログラミング環境を！」

- XcalableMP: これからの計画
  - 文部科学省次世代IT基盤構築のための研究開発「e-サイエンス実現のためのシステム統合・連携ソフトウェアの研究開発」「シームレス高生産・高性能プログラミング環境」プロジェクトで、研究開発を進行中
  - 2009/1Qに初版の仕様を決定
  - C言語版は2009/2Qにβリリース
  - 2009/3QにFortran版
  - 国際的な標準化活動を展開

http://www.xcalablemp.org