2次元反応拡散方程式

$$\frac{\partial u}{\partial t} = D_u \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} \right) + f(u,v)$$

$$\frac{\partial v}{\partial t} = D_v \left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} \right) + g(u,v)$$

Schnakenberg kinetics

$$f(u,v) = k_1 - k_2 u + k_3 u^2 v$$

$$g(u,v) = k_4 - k_3 u^2 v$$

区間 0≤ x ≤ 100, 0 ≤ y ≤ 100を

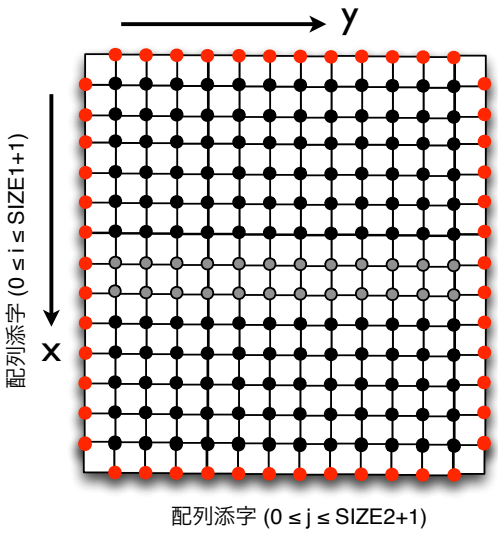
1）適当な分割数 N の下で1ノードで計算

2）2ノードを用いて並列計算

陽的差分法

1ノードで実行する場合
（これまでのやり方）

```
SIZE1 = 101
SIEZ2 = 101
dx = 1.0
```



赤は境界条件用の配列要素

```
double u[SIZE1+2][SIZE2+2];

void go_ahead() /* 時刻を dt だけ進める */
{

  for(i=1; i<=SIZE1; i++)
    for(j=1; j<=SIZE2; j++){
      ud[i][j] = u[i][j] + .....
  }

  境界条件の設定

}
```

配列 (i, j) 成分の座標は ((j-1)*DX, (i-1)*DX) とする

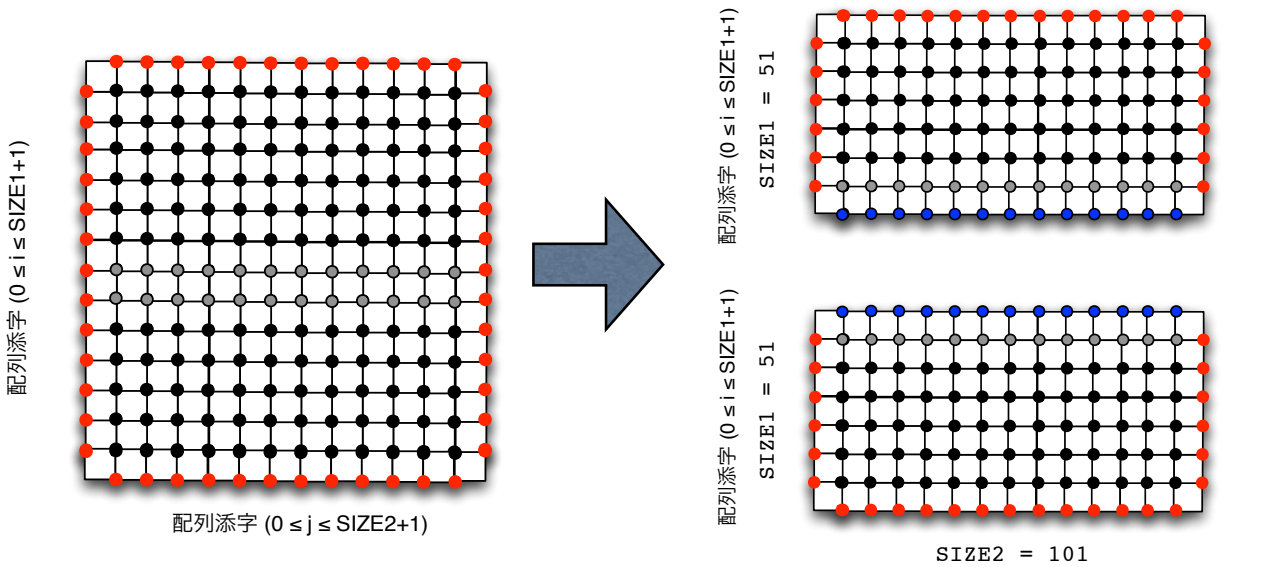
陽的差分法（2次元空間）

- 計算量は空間分割数の二乗に比例 N = SIZE1 x SIZE2
- 1次元の場合と同じく、空間領域を2分割することにより2つのノードで並列計算が可能となる
- 2分割した領域の境界を接続する必要がある

2ノードによる並列処理

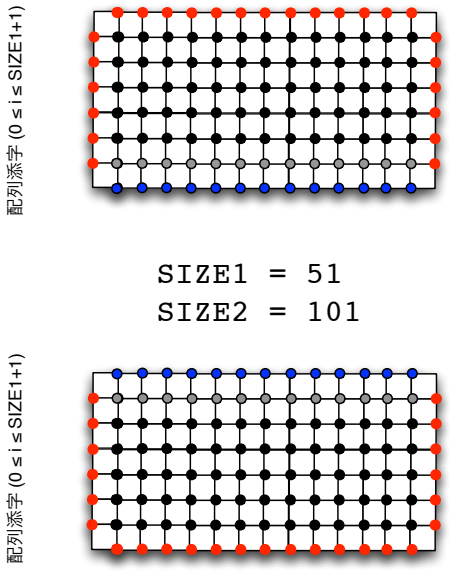
- 空間を二つに分割する

上側の青色の配列要素の値は、下側の配列要素の値と同じ。
下側の青色の配列要素の値は、上側の灰色の要素の値と同じ。




```
SIZE1 = 102, SIZE2 = 101, dx = 1.0
```

2ノードによる並列処理



rank 0

rank 1

2ノードで実行する場合
（空間を2つに分割）

2つに分割した空間を接続する必要がある（ノード間通信）

青は、2つに分割した空間を接続するための境界条件用の配列要素

灰色は、2つに分割した空間を接続するために相手ノードへ送る必要がある配列要素

rank 0 の青色の配列要素の値は、rank 1 の配列要素の値と同じ。
rank 1 の青色の配列要素の値は、rank0 の灰色の要素の値と同じ。

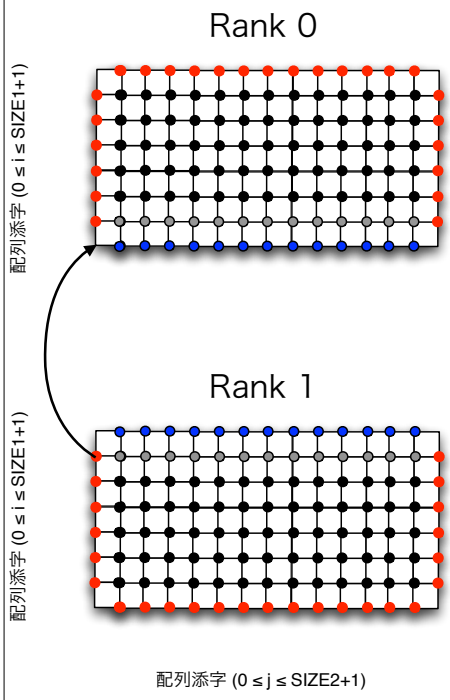
奈良女子大学理学部情報科学科
H25 計算機実験2

2ノードによる並列処理実装

- 時刻を dt 進める関数 process(int rank) を定義
- rank 0 は上半分の区画、rank 1は下半分の区画を担当

```
void go_ahead(int rank)
{
  if( rank == 0 ) {
    MPI により下端の境界条件を取得（send/receive）
    陽的差分
    境界条件の設定
  }
  if( rank == 1 ){
    MPI により上端の境界条件を取得（send/receive）
    陽的差分
    境界条件の設定
  }
}
```

2ノードによる並列処理実装



2次元配列 double u[SIZE1+2][SIZE2+2]の
第 i 行は、一次元配列 u[i] として指定可能
（アドレス &u[i][0] から連続するメモリ領域）

MPI_Send, MPI_Recv で複数個の double 型変数を
一度に送受信できる

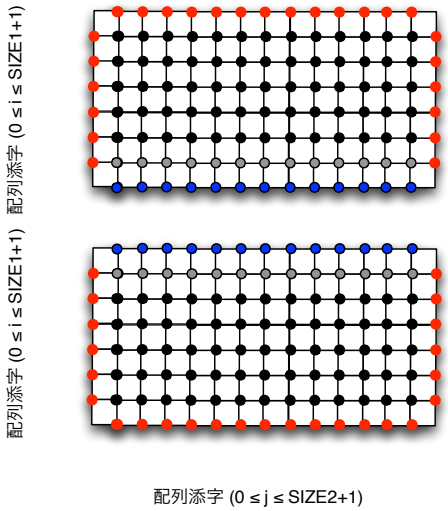
Rank 1 の配列第1行を rank 0 の 配列第SIZE1+1行へ送信

```
// In rank 0: Rank 0 の下端の境界条件を rank 1 から取得
src = 1;
MPI_Recv(&u[SIZE1+1][0], SIZE2+2, MPI_DOUBLE, src, ...)

// In rank 1: Rank 1 が持つ上端の状態を送信
dest = 0;
MPI_Send(&u[1][0], SIZE2+2, MPI_DOUBLE, dest, ...);
```


2ノードによる並列処理実装

- それぞれのノードの計算結果を個別のファイルに書き出す。
- (x, y, z) の形でファイルに書き出す。z は (x, y)における物質濃度
- Mathematica で2つのファイルを連結して図示



Rank 0 の配列 (i, j) 成分の座標は
((j-1)*DX, (i-1)*DX)　1 ≤ i ≤ SIZE1, 1 ≤ j ≤ SIZE2

Rank 1 の配列 (i, j) 成分の座標は
((j-1)*DX, SIZE1*DX + (i-1)*DX)
1 ≤ i ≤ SIZE1, 1 ≤ j ≤ SIZE2

ファイルへの書き出し

```
void write_data(int rank)
{
   int i, j;

   if( rank == 0){

     for(i=1; i<=SIZE1; i+=INTV_THIN)
        for(j=1; j<=SIZE2; j+=INTV_THIN){
           fprintf(fp_u, "%f %f %f ", (j-1)*DX, (i-1)*DX, u[i]);
           fprintf(fp_v, "%f %f %f ", (j-1)*DX, (i-1)*DX, v[i]);
        }
     fprintf(fp_u, "\n"); fprintf(fp_v, "\n");

   }

   if( rank == 1 ){

     for(i=1; i<=SIZE1; i+=INTV_THIN)
        for(j=1; j<=SIZE2; j+=INTV_THIN){
           fprintf(fp_u, "%f %f %f ", (j-1)*DX, SIZE1*DX + (i-1)*DX, u[i]);
           fprintf(fp_v, "%f %f %f ", (j-1)*DX, SIZE1*DX + (i-1)*DX, v[i]);
        }
     fprintf(fp_u, "\n"); fprintf(fp_v, "\n");

   }
}
```


課題

- Schnakenberg kinetics を持つ 2次元反応拡散方程式を 2 つのノードを用いて並列計算するプログラムを作成せよ。
- 実行時間を測定し、並列化の効果（2ノードを用いると 2倍速くなるのか？）について調べよ。特にノード当たりの分割数 SIZE1, SIZE2 を変化させると並列化の効果はどうなるかに注目せよ。
- パラメータは次の値を用いること。

$k_1 = 0.2, k_2 = 1.0, k_3 = 1.0, k_4 = 0.5$

$D_u = 1.0, D_v = 14.0$

$0 \leq x \leq 100 \quad 0 \leq y \leq 100$

1ノード：SIZE1 = SIZE2 = 101, DX = 1.0

2ノード：SIZE1 = 51, SIZE2 = 101, DX = 1.0

並列処理の効果

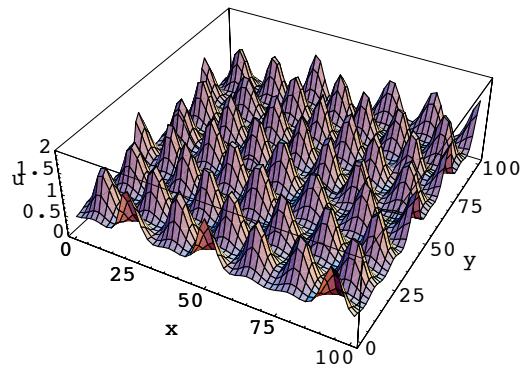
- 時刻を dt 進めるために必要な計算量は分割数 Nの二乗に比例
- ノード間通信に要する時間 Tc（通信には時間がかかる）
- 陽的差分の処理時間＋通信時間が実時間

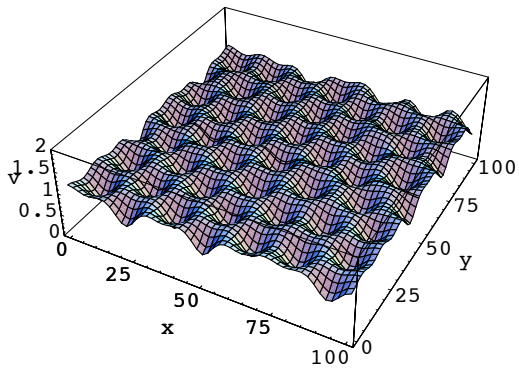
1ノード処理　　2ノード処理

$T \propto N^2$　　$T \propto N^2/2 + Tc$

分割数 N が大きいほど 2ノード並列処理が効果的

奈良女子大学理学部情報科学科
H25 計算機実験2






```
% mpirun -np 2 ./a.out
[Session started at 2009-06-11 16:55:03 +0900.]
Process 0 of 2 on node1.ics.nara-wu.ac.jp
Process 1 of 2 on node1.ics.nara-wu.ac.jp
0/1000000
2000/1000000
4000/1000000
...
996000/1000000
998000/1000000
It took 331.436240 seconds
The Debugger has exited with status 0.
```