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

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

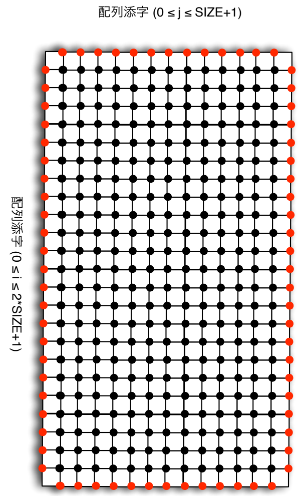
$$g(u,v) = k_4 - k_3 u^2 v$$

区間 $0 \leq x \leq 1, 0 \leq y \leq 2$ 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



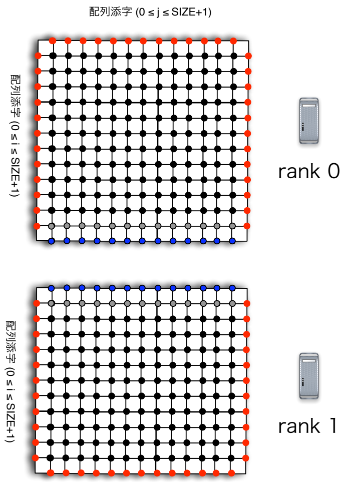
赤は境界条件用の配列要素

1ノードで実行する場合
（これまでのやり方）

```
double u[2*SIZE+2][SIZE+2];

void go_ahead() /* 時刻を dt だけ進める */
{

  for(i=1; i<=2*SIZE; i++)
    for(j=1; j<=SIZE; 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

2ノードによる並列処理



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

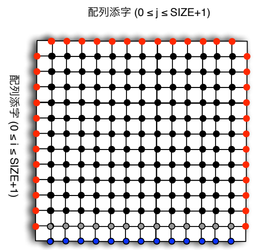
灰色は、2つに分割した空間を接続するために相手ノードへ送る必要がある配列要素

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

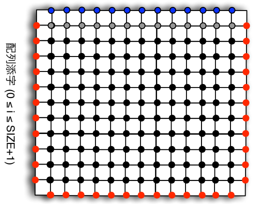
```
void go_ahead(int rank)
{

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

2ノードによる並列処理実装



2次元配列 double u[2*SIZE+2][2*SIZE+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）

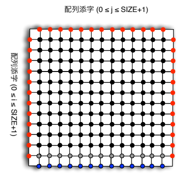
MPI_Send, MPI_Recv で複数個の double 型変数を
一度に送受信する



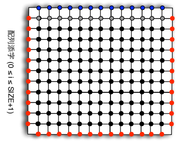

```
In rank 0 // Rank 0 の下端の境界条件を rank 1 から取得
 src = 1;
 MPI_Recv(&u[SIZE+1][0], SIZE+2, MPI_DOUBLE, src, ...);

In rank 1 // Rank 1 が持つ上端の状態を送信
 dest = 0;
 MPI_Send(&u[1][0], SIZE+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。
- 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, j ≤ SIZE



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

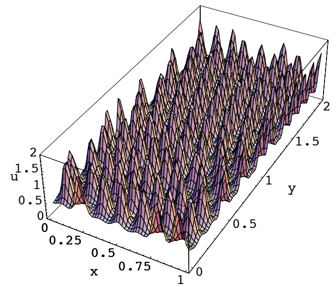
課題

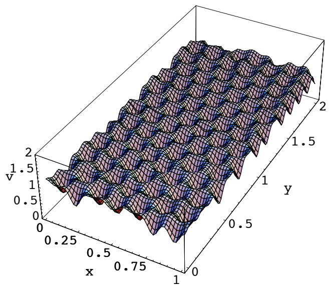
- Schnakenberg kinetics を持つ 2次元反応拡散方程式を 2 つのノードを用いて並列計算するプログラムを作成せよ。
- 実行時間を測定し、並列化の効果（2ノードを用いると 2倍速くなるのか？）について調べよ。特にノード当たりの分割数 SIZE を変化させると並列化の効果はどうなるかに注目せよ。
- パラメータは次の値を用いること。

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

$$D_u = 1.0 \times 10^{-4}, D_v = 1.4 \times 10^{-3}$$

$$0 \le x \le 2 \qquad 0 \le y \le 1$$






```
% 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.
```