单边通信 (One-Sided Communication)

单边通信指的是通信通过一个进程直接读取或者写入另一个进程的内存区完成的通信，这个操作只和一个进程有关系，所以叫做单边通信。

单边通信的问题在于在多个进程同时操作同一块内存区时需要考虑同步和协作的问题。

MPI 的单边通信通过所谓“窗口”来完成。某个进程将自己的一块内存区设为一个窗口，然后其他的进程直接来读写这个窗口中的内容。

这种方式也叫“远程内存访问” (Remote Memory Access) 。

创建窗口

```
int MPI_Win_create(void * base, MPI_Aint size,
                   int disp_unit, MPI_Info info,
                   MPI_Comm comm, MPI_Win * win);
```

聚合操作;
窗口是一个建立在通信器上的概念，不由某个进程所私有;

释放窗口

```
int MPI_Win_free(MPI_Win *win);
```

三种操作

- 读 (Get);
- 写 (Put);
- 累加 (Accumulate);

三种同步管理模式

- Fence;
- 握手?
- Lock;

Get

```
int MPI_Get(void * origin_addr, int origin_count,
        MPI_Datatype origin_datatype, int target_rank,
        MPI_Aint target_disp, int target_count,
        MPI_Datatype target_datatype, MPI_Win win);
```

Put

```
int MPI_Put(void *origin_addr, int origin_count,
        MPI_Datatype origin_datatype, int target_rank,
        MPI_Aint target_disp, int target_count,
        MPI_Datatype target_datatype, MPI_Win win)
```

Accumulate

```
int MPI_Accumulate(void *origin_addr, int origin_count,
          MPI_Datatype origin_datatype, int target_rank,
          MPI_Aint target_disp, int target_count,
          MPI_Datatype target_datatype,
          MPI_Op op, MPI_Win win)
```

Fence

int MPI_Win_fence(int assert, MPI_Win win);

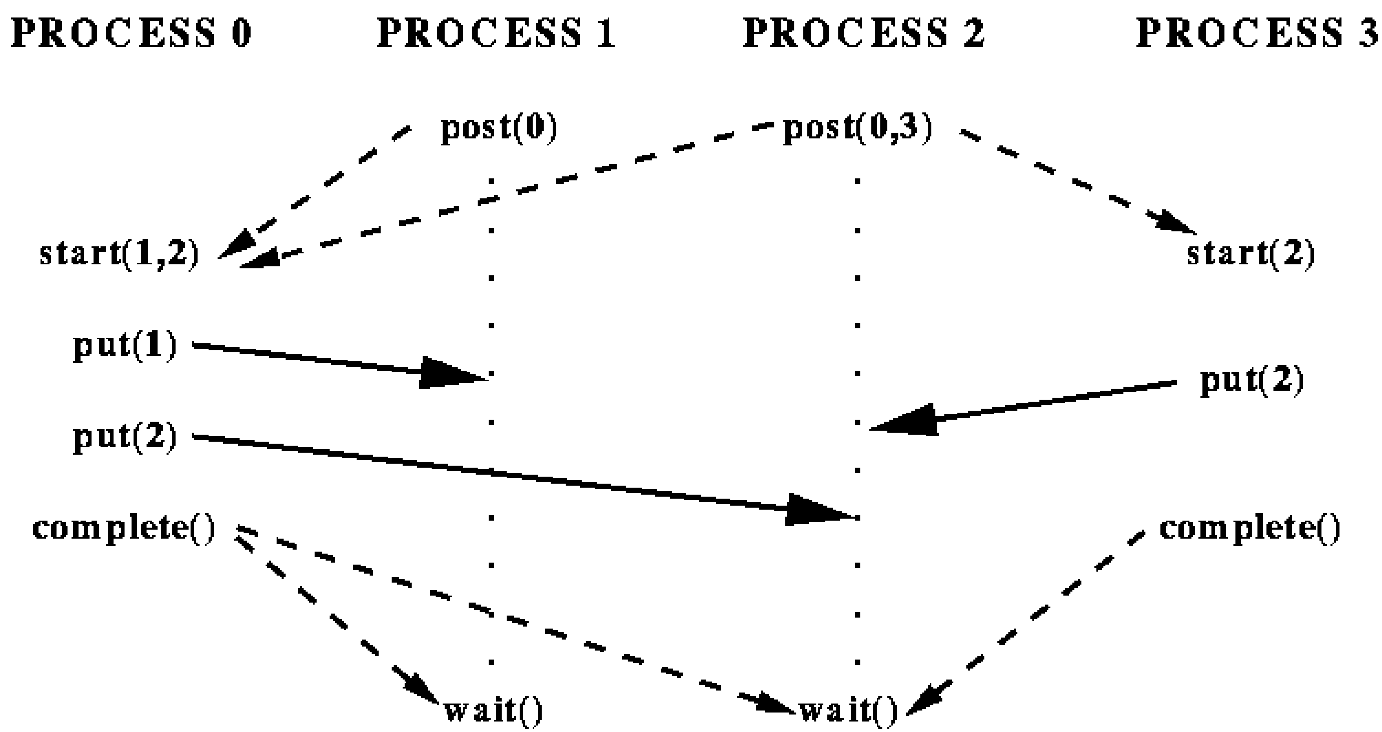
assert: MPI_MODE_NOSTORE,
MPI_MODE_NOPUT,
MPI_MODE_NOPRECEDE
MPI_MODE_NOSECCEEDED

聚合操作，松同步

```
MPI_Win_fence(0, win);
//  读写窗口
MPI_Win_fence(0, win);
```

General Active Target Synchronization(握手)

```
int MPI_Win_start(MPI_Group group, int assert,
                  MPI_Win win);
int MPI_Win_complete(MPI_Win win);
int MPI_Win_post(MPI_Group group, int assert,
                 MPI_Win win);
int MPI_Win_wait(MPI_Win win);
```



Lock

```
int MPI_Win_lock(int lock_type, int rank,
                 int assert, MPI_Win win);
int MPI_Win_unlock(int rank, MPI_Win win);

state: MPI_LOCK_EXCLUSIVE or MPI_LOCK_SHARED

MPI_Win_lock(MPI_LOCK_EXCLUSIVE, rank, assert, win);
//  读写操作
MPI_Win_unlock(rank, win);
```

非重叠区域分解

$$\frac{\partial}{\partial x^i}\left(a^{ij}\frac{\partial u}{\partial x^j}\right)=0, in \quad \Omega$$

$$u\big|_{\partial\Omega}=u_b$$

整个区域分成两个部分

$$\Omega=\Omega_1\cup\Omega_2$$

$$\Gamma_{in}=\overline{\Omega}_1\cap\overline{\Omega}_2$$

$$a^{ij}=\begin{cases} a_1 I, in \quad \Omega_1 \\ a_2 I, in \quad \Omega_2 \end{cases}$$

在子区域的交界面上有

$$\left(a_1 I \nabla u\Big|_{\Gamma_{in}}^{+} - a_2 I \nabla u\Big|_{\Gamma_{in}}^{-}\right)\cdot \vec{\boldsymbol{n}} = 0$$

$$\begin{aligned} 0 &= \int_\Omega \nabla\cdot\left(a^{ij}\nabla u\right) v\,dx,\ \forall v\in C_0^\infty(\Omega) \\ &= \int_{\Gamma_{in}}\left(a_1 I \nabla u\Big|_{\Gamma_{in}}^{+} - a_2 I \nabla u\Big|_{\Gamma_{in}}^{-}\right)\cdot \vec{\boldsymbol{n}}\, v\,ds - \int_\Omega a^{ij}\nabla u \nabla v\,dx \end{aligned}$$

$$\begin{cases} \int_\Omega a^{ij}\nabla u \nabla v\,dx = 0 \\ \int_{\Gamma_{in}}\left(a_1 I \nabla u\Big|_{\Gamma_{in}}^{+} - a_2 I \nabla u\Big|_{\Gamma_{in}}^{-}\right)\cdot \vec{\boldsymbol{n}}\, v\,ds = 0 \end{cases}$$

求解方案

1. 在 Ω_1 上求解 $Dirichlet$ 边值问题

$$\nabla \cdot \left(a^{ij} \nabla u \right) = 0$$

$$u\Big|_{\partial\Omega \cap \overline{\Omega}_1} = u_b$$

$$u\Big|_{\Gamma_{in}}^{+} = u\Big|_{\Gamma_{in}}^{-}$$

2. 在 Ω_2 上求解混合边值问题

$$\nabla \cdot \left(a^{ij} \nabla u \right) = 0$$

$$u\Big|_{\partial\Omega \cap \overline{\Omega}_2} = u_b$$

$$a_2 I \nabla u\Big|_{\Gamma_{in}}^{-} \cdot \vec{\boldsymbol{n}} = a_1 I \nabla u\Big|_{\Gamma_{in}}^{+} \cdot \vec{\boldsymbol{n}}$$

3. 回到1；

$$\Omega=[0,1]\times[0,1]$$

$$\Omega_1=(0,1/2)\times(0,1),\ \Omega_2=(1/2,1)\times(0,1)$$

$$a_1=1, a_2=10$$

$$u_b=0, f=1$$

要求:

- 两个节点并行;
- 使用五点中心差分格式;
- 子区域内部矩阵求解方法自选;

笛卡儿拓扑分区的情况

0,0	0,1	0,2	0,3
1,0	1,1	1,2	1,3
2,0	2,1	2,2	2,3
3,0	3,1	3,2	3,3

i+j= 奇数　　　　　　　　　　i+j= 偶数

交替进行

i+j= 奇数：内边界上使用 Dirichlet 边界条件；
i+j= 偶数：内边界上使用 Neumann 边界条件；

事实上，只需要在每条内边界的两边，分别使用不同的边界条件就可以了。

作业

$$-\Delta u(x,y)+u(x,y)+v(x,y)^3=f$$
$$-\Delta v(x,y)+u(x,y)^3+v(x,y)=g$$

$$\Omega=[-\pi,\pi]\times[-\pi,\pi]$$

双周期边界条件；
使用边界上重叠一层的区域分解方法；
建立具有笛卡儿拓扑的通信器进行通信；
仅仅对拉普拉斯算子隐式进行线性化；
拉普拉斯算子使用 9 点四阶紧致差分格式离散；
每个子区域上使用任意的迭代求解器；

$$f(x,y)=\sin(\pi x)e^{y^2/\pi}$$
$$g(x,y)=\cos(\pi y)e^{(\pi^2-x^2)/\pi}$$