15.1

1. 进程0发送一个消息，并执行函数parallel_fft(...),进程1收到该消息后，执行parallel_fft(...).
2. 进程0会发送一个消息，由于使用的消息标签为tag2，即使没有通信域，也不会被进程1的语句 `MPI_Recv(...)` 接收，故对程序的计算结果没有影响。

15.3

代码段1

```
float data[1024];
MPI_Datatype floattype;
MPI_Type_vector(10,1,32,MPI_FLOAT,&floattype);
MPI_Type_commit(&floattype);
MPI_Send(data,1,floattype,dest,tag,MPI_COMM_WORLD);
MPI_Type_free(&floattype);
```

代码段2

```
float data[1024], buff[10];
for(int i=0;i<10;i++)buff[i]=data[32*i];
MPI_Send(buff,10,MPI_FLOAT,dest,tag,MPI_COMM_WORLD);
```

15.13

代码如下：

```
// hw5-15-13.cpp: 定义控制台应用程序的入口点。
//

#include "stdafx.h"

#define PI 3.141592654f
double randf() {
   return (double)rand() / RAND_MAX;
}
double buffon_laplace() {
   unsigned int in_rect = 0;
   unsigned int total = 0;
```

```
    //假设a=b=l=1
    for (int i = 0; i < 1e6; i++) {
        //针尖位置
        double p0_x = randf();
        double p0_y = randf();
        //角度
        double theta = randf() * 2 * PI;
        //针尾位置
        double p1_x = p0_x + cos(theta);
        double p1_y = p0_y + sin(theta);
        if (p1_x >= 0.0f && p1_x <= 1.0f && p1_y >= 0.0f && p1_y <=
1.0f)in_rect++;
        total++;
    }
    double pro = 1 - (double)in_rect / total;
    return 3.0f / pro;
}
int main()
{
    srand((unsigned int)time(NULL));

    for (int i = 0; i < 10; i++) {
        clock_t start = clock();
        double pi = buffon_laplace();
        clock_t t = clock() - start;
        printf("|%d|%lf|\n", t, pi);
    }

    return 0;
}
```

时间(毫秒)	Π
114	3.142260
136	3.140487
114	3.142378
112	3.142743
110	3.142197
112	3.141522
119	3.142055
126	3.140875
119	3.141220
111	3.141542

可以发现，程序给出的π值的正确位数只有2-3位。