装订线 答题时不要超过此线

中 国 科 学 技 术 大 学

2012－2013 学年第一学期考试试卷

考试科目：并行程序设计　　得分:_____

学生所在系:_____　姓名:_____　学号:_____

本试卷共五个大题！

一、　描述以下循环 1 中的依赖关系及语句依赖图，并尝试向量化变换:（15 分）

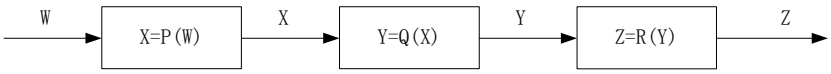
```
for i = 1 to 100 do
S:  A[i+1] = A[i+1] + B[i];
T:  B[i+1] = C[i] + 1;
U:  A[i+1] = A[i] * C[i] + D[i];
endfor    // 循环 1
```

二、判断循环 2 和循环 3 是否等价；并针对循环 3，做出向量化变换和 OpenMP 实现。（20 分）

```
for i = 1 to N do
  A[i] = B[i] +1;
endfor
for i = 1 to N do
  C[i] = A[i] / 2;
endfor
for i = 1 to N do
  D[i] = 1 / C[i+1];
endfor

      //循环 2
```

```
for i = 1 to N do
  A[i] = B[i] +1;
  C[i] = A[i] / 2;
  D[i] = 1 / C[i+1];
endfor




      //循环 3
```

三、以下为一个三进程的流水线。进程 Q 连续地从左边的进程 P 接收一个输入数据流 X，计算一个新的 Y 值，然后将它发送给右边的进程 R。



请设计一个带有双缓冲方式的进程 Q 的流水线 MPI 程序框架，并使用非阻塞消息通信操作实现计算与通信的重叠。（15 分）

四、　将 MPI_COMM_WORLD 中所有进程视为二维 $\sqrt{p}\times\sqrt{p}$ 结构（p 是进程总数且为平方数）。给出进程 $P_{i,j\,(0\le i,j\le\sqrt{p}-1)}$ 将其在 MPI_COMM_WORLD 中编号 rank 分别循环下移 i 步，循环右移 j 步的 MPI 代码片段。（15 分）

五、枚举排序（**Enumeration Sort**）的具体思想是（假设按关键字递增排序），对每一个待排序的元素统计小于它的所有元素的个数，从而得到该元素最终处于序列中的位置。假定待排序的 **n** 个数存在 **a[1]…a[n]**中。首先将 **a[1]** 与 **a[2]…a[n]**比较，记录比其小的数的个数，令其为 **k**，**a[1]**就被存入有序的数组 **b[1]…b[n]**的 **b[k+1]**位置上；然后将 **a[2]**与 **a[1]，a[3]…a[n]** 比较，记录比其小的数的个数，依此类推。

```
输入：a[1]…a[n]
输出：b[1]…b[n]
Begin
for i=1 to n do
(1)   k=1
(2)   for j=1 to n do
            if a[i]>a[j] then
               k=k+1
            end if
      end for
(3)   b[k]= a[i]
end for
End //枚举排序串行算法
```

（1） 给出 MPI 并行实现。假设对一个长为 **n** 的输入序列使用 **n** 个处理器进行排序，只需是每个处理器负责完成对其中一个元素的定位，然后将所有的定位信息集中到主进程中，由主进程负责完成所有元素的最终排位。（**20** 分）

（2） 给出 OpenMP 实现。（**15** 分）

本试卷答题过程中可能用到的函数原型：

MPI_Reduce(void *sendbuf, void *recvbuf, int count, MPI_Datatype datatype, MPI_Op op, int root, MPI_Comm comm)

MPI_Bcast(void *buffer, int count, MPI_Datatype datatype, int root, MPI_Comm comm)

MPI_Gather(void *sendbuf, int sendcnt, MPI_Datatype sendtype, void *recvbuf, int recvcnt, MPI_Datatype recvtype, int root, MPI_Comm comm)

MPI_Allgather(void *sendbuf, int sendcount, MPI_Datatype sendtype,void *recvbuf, int recvcount, MPI_Datatype recvtype, MPI_Comm comm)

MPI_Send(void *buf, int count, MPI_Datatype datatype, int dest, int tag, MPI_Comm comm)

MPI_Isend(void *buf, int count, MPI_Datatype datatype, int dest, int tag, MPI_Comm comm, MPI_Request *request)

MPI_Recv(void *buf, int count, MPI_Datatype datatype, int source, int tag, MPI_Comm comm, MPI_Status *status)

MPI_Irecv(void *buf, int count, MPI_Datatype datatype, int source, int tag, MPI_Comm comm, MPI_Request *request)

MPI_Sendrecv(void *sendbuf, int sendcount, MPI_Datatype sendtype,int dest, int sendtag, void *recvbuf, int recvcount, MPI_Datatype recvtype, int source, int recvtag, MPI_Comm comm, MPI_Status *status)

MPI_Wait(MPI_Request *request, MPI_Status *status)