装订线 答题时不要超过此线

中 国 科 学 技 术 大 学

2008－2009 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:_____

一、给出以下循环中的迭代依赖图。（10 分）

```
for(  k = 0; k<N; k++  ) {
  a[ k ][ k ] = sqrt( a[ k ][ k ] );
  for(  i = k+1; i<N; i++ ) {
    a[ i ][ k ] = a[ i ][ k ] / a[ k ][ k ];
    for(  j = k+1; j<N; j++ ) {
      a[ i ][ j ] = a[ i ][ j ] - a[ i ][ k ] * a[ k ][ j ] / a[ k ][ k ];
    }
  }
 }
```

二、分别考查以下循环中的**依赖关系**以及**可向量化和可并行化**情况（40 分）

```
（1）for I = 2 to N – 1 do
        for J = 2 to N – 1 do
   S :      A(I, J) = B( I–1, J ) + C
   T :      B(I, J) = A( I, J–1) * 2
        endfor
   endfor

（2） for I = 1 to N  do
        for J = 1 to N do
   S :      D(I, J)     =  A( I, J ) + C
   T :      A(I+1, J+1) =  B( I, J ) * 2
        endfor
    endfor
```

三、给出以下程序的 **MPI 并行实现**。假设数组 a 的顺序分布于各计算进程（不必给出数据分发、空间分配以及结果收集的代码）。（20 分）

```
for( j=0; j<100000; j++)
     for( i=2;i<10000; i++)
        a[i] = sin(a[i] - a[i-1]) + cos(a[i] + a[i-2]);
```

四、仔细阅读以下程序代码（30 分）

（1） 给出 **OpenMP** 并行化实现（尽可能多地采用多线程并行化）；

（2） 给出 **MPI** 并行化实现；假设：数组 a 采用行连续划分方式顺序分布于各计算进程，数组 x 的初值也被广播到各个进程（不必写数据分布与最后结果的回收代码了）。

注：在上述两种实现里，可以添加适当的变量及操作。

```
#define N 5000
#define epsilon 1e-5  // 10^-5
double x[N], newx[N], a[N][N], b[N], diff;
int i,j;
main(){
  … … // a,b 的初始化；
  for(i=0;i<N;i++) x[i] = b[i] / a[i][i]; //设置数组 x 的初值
  diff = epsilon;
  while (diff >= epsilon) do {
     diff = 0;
     for( i=0;i<N;i++){
           newx[i]= b[i];
           for( j= 0;j<N;j++){
              if (j != i)  newx[i] = newx[i] - a[i][j]*x[j];
           }
            newx[i]  = newx[i] / a[i][i];
     }
     for( i=0; i<N;i++){
           diff = max (diff, fabs(newx[i]-x[i]));
           x[i] = newx[i];
     } // 设 max 为求最大值函数，而 fabs 为求绝对值函数。
  }
  … … //输出 x[i]
}
```

可能用到的 MPI 函数原型：

MPI_Send(buf, count, datatype, dest, tag, comm);

MPI_Recv(buf, count, datatype, source, tag, comm, MPI_Status *status);

MPI_Bcast(buf, count, datatype, root, comm);

MPI_Allgather(sendbuf, sendcount, sendtype, recvbuf, recvcount, recvtype, comm);

MPI_Allreduce (sendbuf, recvbuf, count, datatype, MPI_Op op, comm);