Announcements

- Starting next week, there will be paper reading assignments
- Assignment 1 on MPI will be out on September 9 and due on September 23
Summary of last lecture

- Parallel architectures and programming models
- Message passing and MPI

Basic MPI routines:
- MPI_Init, MPI_Finalize
- MPI_Comm_rank, MPI_Comm_size
- MPI_Send, MPI_Recv

Delivery order: only guaranteed between a pair of processes
Collective operations

- **int MPI_Barrier( MPI_Comm comm)**
  - Blocks until all processes in the communicator have reached this routine

- **int MPI_Bcast( void *buffer, int count, MPI_Datatype datatype, int root, MPI_Comm comm)**
  - Send data from root to all processes

- **int MPI_Reduce( const void *sendbuf, void *recvbuf, int count, MPI_Datatype datatype, MPI_Op op, int root, MPI_Comm comm)**
  - Reduce data from all processes to the root
Collective operations

- `int MPI_Scatter( const void *sendbuf, int sendcount, MPI_Datatype sendtype, void *recvbuf, int recvcount, MPI_Datatype recvtype, int root, MPI_Comm comm)`
  - Send data from root to all processes

- `int MPI_Gather( const void *sendbuf, int sendcount, MPI_Datatype sendtype, void *recvbuf, int recvcount, MPI_Datatype recvtype, int root, MPI_Comm comm)`
  - Gather data from all processes to the root

- `MPI_Scan`
Calculate the value of \[ \pi = \sqrt{\int_0^1 \frac{4}{1 + x^2}} \]

```c
int main(int argc, char *argv[]) {
    ...

    n = 10000;

    h   = 1.0 / (double) n;
    sum = 0.0;

    for (i = 1; i <= n; i += 1) {
        x = h * ((double)i - 0.5);
        sum += (4.0 / (1.0 + x * x));
    }
    pi = h * sum;

    ...
}
```
Calculate the value of \[ \pi = \int_0^1 \frac{4}{1 + x^2} \]

```c
int main(int argc, char *argv[]) {
    ...

    n = 10000;
    MPI_Bcast(&n, 1, MPI_INT, 0, MPI_COMM_WORLD);

    h   = 1.0 / (double) n;
    sum = 0.0;

    for (i = myrank + 1; i <= n; i += numranks) {
        x = h * ((double)i - 0.5);
        sum += (4.0 / (1.0 + x * x));
    }
    pi = h * sum;

    MPI_Reduce(&pi, &globalpi, 1, MPI_DOUBLE, MPI_SUM, 0, MPI_COMM_WORLD);

    ...
}
```
MPI communicators

- Communicator is a group or set of processes numbered 0, … , n-1
- Every program starts with MPI_COMM_WORLD
- Several MPI routines to create sub-communicators
  - MPI_Comm_split
  - MPI_Cart_create
  - MPI_Group_incl
Non-blocking point-to-point calls

- MPI_Isend and MPI_Irecv
- Two parts:
  - post the operation
  - Wait for results: need to call MPI_Wait or MPI_Test
- Can help with overlapping computation with communication
Other MPI Calls

- MPI_Wtime
- MPI profiling interface: PMPI_*
Questions?

Abhinav Bhatele
5218 Brendan Iribe Center (IRB) / College Park, MD 20742
phone: 301.405.4507 / e-mail: bhatele@cs.umd.edu