MPI Send and Recv Semantics

• These are “standard mode”, blocking calls.
• Send returns when buf may be re-used; Recv returns when data in buf is available.
• Count consecutive items of type datatype, beginning at buf, are sent to process with rank dest.
• Tag can be used to distinguish among messages from the same source.
• Messages are non-overtaking.
• Buffering policies depend on implementation.
Inside MPI_Send and MPI_Recv
MPI Communicators

• A *communicator* can be thought of as a set of processes; every communication event takes place within a particular communicator.

• MPI_COMM_WORLD is the initial set of processes (note the *static* process model).

• Why do communicators exist?
  – Collective operations over subsets of processes
  – Can define special *topologies* for sets of processes
  – Separate communication contexts for libraries.
MPI Collective Operations

• An operation over an entire communicator
• Must be called by every member of the communicator
• Three classes of collective operations:
  – Synchronization (MPI_Barrier)
  – Data movement
  – Collective computation
Collective Patterns (Gropp)
Collective Computation Patterns (Gropp)
Collective Routines (Gropp)

- Many routines:
  - Allgather
  - Allgatherv
  - Allreduce
  - Alltoall
  - Alltoallv
  - Bcast
  - Gather
  - Gatherv
  - Reduce
  - ReduceScatter
  - Scan
  - Scatter
  - Scatterv

- ‘All’ versions deliver results to all participating processes.

- ‘V’ versions (e.g., Scatterv) allow chunks to have different sizes.

- Allreduce, Reduce, ReduceScatter, and Scan take both built-in and user-defined combination functions.