PARALLEL AND DISTRIBUTED PROGRAMMING.
Экзаменационная программа
2 год магистратуры, 3 группа, 2017-2018 уч. г., 1 семестр

Составитель: М.Э.Абрамян

Topic 1.1. MPI Technology: a general description. Point-to-point communication

The history of development and the versions of the MPI interface (Message Passing Interface). The basic MPI notions: a process, a communicator, a message sender and a message receiver, MPI runtime. The MPICH system. Six basic functions of MPI. Timing functions. Blocking and non-blocking point-to-point communications. Communication modes: the standard mode, synchronous mode, the buffered mode, the ready mode. Receiving messages and related functions including MPI_Probe and MPI_Get_count. Deadlocks during the blocking communications and how to avoid them. A non-blocking receiving messages and the Wait and Test family of functions. Receiving messages from any source (MPI_ANY_SOURCE) and receiving messages with any tag (MPI_ANE_TAG). Combined send-receive operations and their features.

Topic 1.2. Collective communication. Global reduction operations

Collective communications and their main differences with the point-to-point communications. Barrier synchronization via the MPI_Barrier function. The types of collective communications: broadcasting communication MPI_Bcast, gathering data to one member MPI_Gather and MPI_Gatherv, gathering data where all members receive the result MPI_Allgather and MPI_Allgatherv, scattering data to all members MPI_Scatter and MPI_Scatterv, combined scattering/gathering data from all to all members MPI_Alltoall and MPI_Alltoallv. Collective communications with a varying count of data for each process, their features. The predefined reduction operations, the specific features of the MPI_MINLOC and MPI_MAXLOC operations. The types of possible reduction operations: receiving the result data in one process MPI_Reduce, receiving the result data in all processes MPI_Allreduce, combined reduction and scattering the results data to all processes MPI_Reduce_scatter, the prefix reduction MPI_Scan.

Topic 1.3. Global reduction operation. Derived datatypes. Packing and unpacking

Derived MPI datatypes and their characteristics: the extent (the MPI_Type_extent function) and the size (the MPI_Type_size function). Datatype constructors: replication of a datatype into contiguous locations (MPI_Type_contiguous), replication of a datatype into locations that consist of equally spaced blocks (MPI_Type_vector and MPI_Type_hvector); replication of a datatype into a sequence of blocks, where each block can contain a different number of copies and have a different displacement (MPI_Type_indexed and MPI_Type_hindexed); replication of a datatype into a sequence of blocks, where each block consists of different datatypes (MPI_Type_struct). Derived datatypes committing and releasing. Using the derived datatypes for the efficient sending of two-dimensional data blocks. Packaging and unpackaging data via the MPI_Pack and MPI_Unpack functions, their features. Sending of packed data (the MPI_PACKED datatype).

Topic 1.4. Groups of processes and communicators

Predefined groups of processes; group constructors: create a new groups of processes on the basis of an existing ones or on the basis of the communicator, using a set operations for groups. Predefined communicators; communicator constructors: create a new communicator on the basis of existing groups and using the MPI_Comm_split function for splitting of the existing communicator. Communicators' comparing. The use of new communicators for execution of collective operations within the part of existing processes.

Topic 1.5. Virtual topologies

Communicators with the topology. Types of topologies: the Cartesian topology and the graph topology. Constructors for communicators with the topology. Cartesian topology with periodicity in some dimensions. Additional features of communicators with the Cartesian topology: rank-to-coordinates and coordinates-to-rank translation, the data shift for a given coordinate (non-periodic or periodic), partitioning of Cartesian communicator into communicators with low-dimensional Cartesian topology. Additional features of communicators with the graph topology: providing the information about the neighbors of a certain process (adjacency information for a graph topology).

Topic 1.6. Parallel input-output

The basic notions of the MPI parallel IO: file, file handle, file pointer, file displacement. Opening and closing a file: the MPI_File_open and MPI_File_close functions, file access modes. Size of file, the MPI_File_set_size and MPI_File_get_size functions. File view and its components: file offset, etype, filetype, data representation; the MPI_File_set_view function. Data access routines, three aspects to data access: positioning, synchronism, coordination. Two types of file pointers. Samples of MPI functions related to various types of data access.

Section 2. Developing multi-threaded applications using OpenMP technologies

Topic 2.1. OpenMP Programming Interface: a general description

OpenMP interface and its versions. Model of the OpenMP program. The compiler mode with the OpenMP support. OpenMP directives and routines (functions). Parallel parts of the OpenMP-program. Shared and local variables in the OpenMP-program. Setting the parameters of the OpenMP-program via environment variables, OpenMP functions, and clauses of the OpenMP directives. Timing functions.

Topic 2.2. The basic means of parallelization in OpenMP

Low-level parallelization via omp_get_thread_num () and omp_get_num_threads (). Parallel sections, their features. Directives for single-thread execution. Parallelization of loops, its features. The schedule clause and schedule types for load balancing (static block distribution, dynamic block distribution with fixed-size blocks and guided block distribution with variable-size blocks).

Topic 2.3. The synchronization means in OpenMP

The importance of synchronization for parallel programs with shared memory. Critical sections and their usage (the critical directive). The barrier synchronization (the barrier directive). The assignment synchronization (the atomic directive). An example of ineffective and effective synchronization of the implementation of the parallel algorithm for finding the maximum or minimum value.