Technische Universität München
Institut für Informatik
Dr. rer. nat. Tobias Neckel
Matous Sedlacek

WT 2012/2013

Parallel Numerics

Exercise 7: Sparse Matrix–Vector Multiplication & MPI Communicators

1) MPI Communicators

i) Give examples for some network topologies (e.g. cartesian grids, hypercubes, tree). How are the standard mpi ranks mapped onto such topologies?
Topologies: Cartesian grid, Bus, Line, Tree, Fat-Tree, Hypercube. Mapping is provided by virtual topologies.

ii) Which topology fits perfectly to Cannon's algorithm?
Torus topology.

iii) Besides a topology, a mpi communicator holds a group, a communication context (communicator specific tagging) and communicator specific buffers. Thus, e.g. collective operations act only for elements of the same communicator. Make yourself familiar with the communicator concept.
Communicators encapsulate (all derived from `MPI_COMM_WORLD`):

- ***Contexts** of communication: Provide the ability to have seperate safe "universes" of message-passing in MPI. A context is a system-defined object that uniquely identifies a communicator. A message sent in one context can't be received in other contexts. Thus, the communication context is the fundamental methodology for isolating messages in distinct libraries and the user program from one another.*
- ***Groups** of processes: A group is an ordered set of processes, each process in the group is assigned a unique identifier (rank).*
- ***Virtual topologies**: Mapping of the ranks in a group to and from a topology.*
- ***Attribute caching**: Kind of caching mechanism (specific buffers) that allows one to associate new attributes with communicators, for instance to adorn communicators further. Also used by MPI to implement some communicator functions.*

iv) Give an application domain for mpi communicators.
MPI concept to virtually represent topology of underlying application. See 1 i).

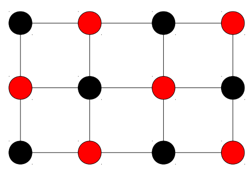
2) Red–Black Colouring

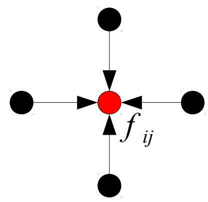
The idea of any colouring approach for an iterative algorithm is to assign evaluations to different computing steps (colours). Afterwards, these steps are executed sequentially.

i) What are typical properties of elements of one colour?
All elements of one colour work with "same old" data. The operations of one colour should be evaluated in parallel.

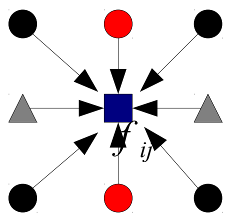
ii) Derive a colouring scheme for a 5–point stencil on a cartesian two–dimensional grid.
Unknowns are arranged in cartesian grid u_{ij} *and are updated via function* f_{ij}*. Function* f_{ij} *depends on left (*f_{ij-1}*), right (*f_{ij+1}*), upper (*f_{i+1j}*), and lower (*f_{i-1j}*) neighbour. Using red-black ordering:*

- *red phase: update all unknowns (evaluate all* f_{ij}*) belonging to red points. The evaluation can be done completely in parallel.*
- *black phase: use new values of red phase.*
- *→ alternate phases until convergence.*





iii) Why is bicolouring for a 9–point stencil on a cartesian two–dimensional grid not well–suited? Derive an improved colouring.
No parallel evaluation possible with two colours → sequential algorithm. Use four colours instead.



3) Sparse Matrix–Vector Multiplication

The solution of the matrix-vector product $A \cdot b$ with a sparse matrix $A \in \mathbb{R}^{n \times n}$ and $b \in \mathbb{R}^n$ should be calculated. Write different procedures for the calculation of the matrix-vector multiplication. Implement the storage schemes for matrix A that have been discussed in the lecture:

1. storage in coordinate form
2. compressed sparse row format (CSR)
3. compressed sparse column format (CSC)
4. CSR with extraction of the main diagonal entries
5. diagonal–wise storage (for band matrices)
6. rectangular, row–wise storage scheme
7. jagged diagonal form

Test your routines for large sparse matrices whose elements are randomly distributed as zeros or as non-zero values. Compare the computation time to the time needed by applying the full matrix storage scheme.
See sourcecode to corresponding tutorial on webpage.