Laboratory for Parallel Programming
Prof. Dr. Felix Wolf

TECHNISCHE UNIVERSITÄT DARMSTADT

Aamer Shah, shah@cs.tu-darmstadt.de

Large-Scale Parallel Computing (WS 15/16)

Exercise 2

A solution will be presented on November 10th, 2015. Attendance is optional. The exercise solutions will not be corrected and graded.

Task 1

On HPC systems, many independent compute nodes are connected together through a network to form a single computing system. When a parallel program executes on such system, it executes an independent process on each compute node. As each node then has its own memory, applications use message passing to exchange data and information between the compute nodes.
In this task, we will implement such a message-passing-based application, but will execute it on our own systems. The program will be implemented in C and will use network sockets to pass messages.
The program will consist of a master task and three workers tasks. The objective of the program will be to read two matrices from a file and calculate its sum. The sum will be written back to an output file. The master task will read the two matrices and distribute them among all the tasks (master + workers). Each task will then calculate its partial sum of the matrices. The matrices will then be send back to the master. The master will collect all the partial sums to generate a complete sum matrix. This matrix will be written to an output file.
The attached archive file provides a skeleton for both the master and workers tasks. Use the `Makefile` to compile the programs and the `run_ex.sh` batch script to run the program. Extend the source files to implement matrix addition in the following steps:
The master performs the following steps:

a) Extend the master by first generating and sending ids to each worker task. The master itself has id 0.

b) Come up with a scheme to distribute the matrices among the tasks. This can be either distributing along rows, along columns or both.

c) Inform the workers how many rows and columns they are expected to receive. (Note: the master also gets a share)

d) Send each worker its share of the matrices

e) Calculate the local partial sum

f) Receive the partial sum from each worker

g) Generate the complete sum and write it to the output file

The workers perform the following steps:

a) Receive id from the master

b) Receive the number of rows and columns of the matrices

c) Receive the matrices from the master

d) Calculate the sum of the matrices

e) Send the matrices back to the master

Task 2

In image processing, a median filter is used to reduce noise from an image. The median filter works by using a 2D window of a certain size. Then for each element of the image, the value of the element is replaced by calculating the median of the window at that point. This way, sudden change in values is smoothed out and noise is reduced.
In this task, no implementation work is required. The task is to come up with a design of a message-passing-based application consisting of master and workers that apply median filter on an image. Perform the task in the following steps:

a) Identify how this task is different from matrix addition

b) Come up with a design of a median filter application by extending the design of the matrix multiplication program

c) What will happen if we want to apply the filter twice on the same image (of course without running the program twice)?

Task 3

Based on the above experiences, if you want to make a utility library for message passing applications, what kind of functions will you provide to the user?