1. Introduction

In this exercise, we will try to calculate the bandwidth between two computers. If you are not familiar with the Sandstorm cluster infrastructure and/or Linux server manipulation, please read the Introduction document. You should also have completed the first two exercises.

2. The code

Following code contains some MPI_* commands, which we will use intensively in next exercises, this time you can take them at face value.

```c
/* main.c */
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "mpi.h"
#include "omp.h"

#define MASTER 0
#define SLAVE 1

int main(int argc, char** argv) {
    int numtasks, taskid;
    char hostname[MPI_MAX_PROCESSOR_NAME];
    MPI_Status status;
    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &taskid);
    MPI_Comm_size(MPI_COMM_WORLD, &numtasks);

    if (taskid == MASTER)
        printf("MASTER: Number of MPI tasks is: %d\n", numtasks);

    long NMAX = 1000000;
    long REPEAT = 10000;
    long i, N;
    for(N=10; N<=NMAX; N*=10) {
        char* A;
        A = (char*)malloc(N*sizeof(char));

        for(i=0; i<N; i++) A[i] = 'd';

        double t_start = omp_get_wtime();

        for(i=0; i<REPEAT; i++) {
            if (taskid == MASTER)
                MPI_Send(&A[0], N, MPI_CHAR, SLAVE, 1, MPI_COMM_WORLD);
            else //(taskid == SLAVE)
                MPI_Recv(&A[0], N, MPI_CHAR, MASTER, 1, MPI_COMM_WORLD, &status);
        }
    }
}```
double t_end = omp_get_wtime();
free(A);

double bandwidth = (double)REPEAT*N*sizeof(char)/(t_end - t_start);

if (taskid == MASTER)
    printf("%10ld;%10e\n", N, bandwidth);
}

MPI_Finalize();
return 0;

3. Running the Code

To be able to use the MPI programming scheme, we need to use a different compiler, e.g. `mpicc` is based on `gcc` and can be called in the same way with the additional parameters.

    mpicc main.c -o bandwidth.exe -fopenmp

To execute the program in parallel, we must use another calling scheme. The `mpiexec` takes as one of the parameters the number of processes that will run in parallel, which is in this case 2.

    mpiexec -n 2 ./bandwidth.exe

4. Results

In the following table you can write down the bandwidths as dependent from the size of data.

<table>
<thead>
<tr>
<th>N</th>
<th>Bandwidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td></td>
</tr>
<tr>
<td>100</td>
<td></td>
</tr>
<tr>
<td>1000</td>
<td></td>
</tr>
<tr>
<td>10000</td>
<td></td>
</tr>
<tr>
<td>100000</td>
<td></td>
</tr>
<tr>
<td>1000000</td>
<td></td>
</tr>
<tr>
<td>10000000</td>
<td></td>
</tr>
</tbody>
</table>

5. Contact

Should any difficulties arise, please do not hesitate and contact me per email. But remember – Google is your friend (http://www.giyf.com)! 

Tutor: Stefan Markic  stefan.markic@tum.de