1 requirements

For the processing of the tasks of this course you should (at least) know one of the pro-gramming languages C / C ++ or Fortran77 / Fortran90 / Fortran95. Optionally, it is worth the literacy skills now.
The visit of the lecture Parallel Algorithms is not explicitly provided, however, could facilitate the processing of the tasks clear knowledge of the lecture. It is perhaps useful to copy the transcript of students. A script for the lecture can be found as a script AUD II "on the home page of the internship

http://www-ai.math.uni-wuppertal.de/SciComp/teaching/ss14/praktikum

The tasks should be processed in groups of two.
For the achievement of the bill are the processing

The tasks should be processed in groups of two.
For the achievement of the bill are the processing
• the import end "blocks (tasks 1-3) as well

• two of the three thematic blocks
- Direct solution of linear systems (tasks 4-5)
- Iterative solution of linear systems (tasks 6-7)
- Calculation of eigenvalues and vectors (two of the tasks 8-10)
sufficiently. It is
• For questions 1, 2 and 4 only parallelization with MPI,
• in the remaining tasks (3, 5-10) both parallelization with MPI and
by feeds parallelization with OpenMP.
The deadlines for the processed objects (demonstration and listings) will be announced in the preliminary discussion.

2 literature

The parallel programs should be created firstly using the MPI communication library (Message Passing Interface). As an introduction to MPI about could under

http://moss.csc.ncsu.edu/ mueller / cluster / mpi.guide.pdf

also available short Introduction by Peter Pacheco, possibly supplemented by material from

http://www-unix.mcs.anl.gov/mpi/tutorial/

In addition, provides

http://www-unix.mcs.anl.gov/mpi/

an introduction to the MPI World with many other documents (in particular the definitions of standards MPI-1.1 and MPI-2.0) and free available MPI implementations
(for Linux). There are also references can be found on the textbooks of Gropp, Lusk and Skjellum or Pacheco.
It is recommended that you go through at least Pacheco short introduction before the
MPI programming is started.

For programming with OpenMP should under http://www.openmp.org

available documentation (especially the OpenMP Standard
be used for each programming language in use).

3 General information

• The objects must of course not be solved in the order given. However, it is recommended to process at least the import tasks 1-3 first.

• The following descriptions (for example, the data distributions, etc.) refer to the MPI parallelization. In OpenMP shared storage is based, so that the data must not be distributed. This leads to a significantly against over MPI
simplified parallelization.

• For the OpenMP parallelization you should proceed from a serial program, are pasted in the then appropriate OpenMP directives.

• Drove you numerical calculations Fundamentally, by double precision and store the data (real numbers, vectors and matrices) in an appropriate format (double in C / C ++ or double precision in Fortran).

4 Block 0 - Introductory tasks

Task 1 (1 point)
Write a program which for each process determines it's number and prints out
in formated output
Test your program for p = 1, 2, 4,. 7

At p = 4 processes the output should look something like this:
Program running on 4 processes, I am no process. 0
Program running on 4 processes, I am no process. 3
Program running on 4 processes, I am no process. 1
Program running on 4 processes, I am no process. 2

Exercise 2 (2 points)
Write a program that
a) in process 0 number is set by user's input
b) then transmits it through the process ring to all other processes while in
each process doubled and outputs.
Test your program for p = 1, 2, 4,. 7
Note: The processes are by default uniformly configured as a ring.

For p = 4 processes the output should look something like this:

```
Enter a number in [0, 1000]: 12.5

Process 0 got 12.5000 and computed 25.0000
Process 1 got 25.0000and computed 50.0000
Process 3 got 100.0000 and computed 200.0000
Process 2 got 50.0000 and computed 100.0000
```

Exercise 3:

Implement the Simpson's Rule with n subintervals for calculating

$$s \approx \int_a^b f(x)\,dx$$

and measure the time required.
Test your program for [a, b] = [0, 2],

$$f(x) = x^{16}$$

and n = 512, 1024, 65536 and p = 1.4,. 7

Description: The Simpson's Rule of integration range [a, b] equidistant by n + 1 points is

$$x_i = a + ih,\ i = 0, \ldots, n,$$

with h = (b - a) / n in n intervals [x i, x i + 1] with associated center points

$$x_{i+1/2} = \frac{x_i + x_{i+1}}{2}, i = 0, \ldots., n-1,$$

disassembled and the integral by the sum

$$s = \frac{h}{6} * \sum_{i=0}^{n-1} \left(f(x_i) + 4 * f\left(x_{i+\frac{1}{2}}\right) + f(x_{i+1})\right)$$

approximated

Note: To obtain a meaningful runtime, you should synchronize before timekeeping all processes with a barrier and determine the maximum over the local times at the end.

On p=4 processes the output should look something like this:

```
Enter the interval bounds (a, b) : 0 2
Enter the number of subintervals (n) : 65536
Simpsons Rule for [ 0.0000, 2.0000 ] with        65536 subintervals
        yielded the approximation 7710.117647058816
        and took 0.042057 seconds
```

5 Block 1 - Direct solution of linear systems

Task 4 (6 points)

https://en.wikipedia.org/wiki/Circulant_matrix

A matrix A ∈ R n × n is split cyclically on the processes P_1,. , , , P_p,
In the case P_i contains exactly the column j of A with j ≡ i mod p.

a) Write a routine DistributeColumns, which is a completely in process Pi
This matrix A ∈ R m × n column cyclically distributed to the processes

b) Write a routine SelectColumns, which cyclically split on the distributed processes
matrix A ∈ R m × n in a process is Pi.

c) Write a routine Transponse, which cyclically split on the distributed processes
column-cyclically Matrix A ∈ R m × n distributed to the processes Transpose
A^T is build without A or A^T in a single process (determined).

There are respectively sent no individual elements of the matrix, but few communication operations are used as possible. Test your routines, for differenet n, m and p = 1, 3, 4 processes the matrix

$$A=(a_{ij})\in R^{mxn}\, with\, a_{ij}=i+\frac{j}{1000}$$

each:

1. local In a process P_i is produced
2. from there DistributeColumns distributes columns cyclically,
3. Transposed with (distributed) transpose,
4. the matrix A^T with CollectColumns is again gathered in P_i and
5. compare there with A^T.

Task 5(10 Points)

Implement a parallel solver for linear systems of equations of the form

$$AX = B, A \in R^{n\,x\,n}, B \in R^{n\,x\,q}$$

so that multiple systems

$$Ax^{(i)} = b^{(i)}, A \in R^{nxn}, b^{(i)} \in R^{n} (i = 1, \ldots\ldots, q)$$

can be solved with the same matrix A.
To do this, follow these steps.

Realize it the Gaussian elimination with column pivoting.
Transform additionally the right side B equal with, that work with the matrix (A | B) instead of A. To do this, from the following serial (kji-) form of Gaussian elimination out:

```
for k = 1:n
    {Column pivoting}
    Determine largest element $a_{p_k,k}$ in A(k: n, k)
    Swap lines k and $p_k$ of A
    {Determine multipliers}
    for I = k + 1 : n
        $a_{ik} := a_{ik} / a_{kk}$
    for j = k + 1 : n + q
        { Add ( - $a_{kj}$) – times the multiplier column to column j}
        for  i = k + 1 : n
            $a_{ij} := a_{ij} - a_{ik} * a_{kj}$
```

Use a column cyclic storage of data.

b) In the above Gauss algorithm (AB) is overwritten by

$$(U \mid \tilde{B}) = (L^{-1} P^{T} A \mid L^{-1} P^{T} B)$$

with $P^T A = LU$. Here P is determined by the interchange permutation matrix, L a lower and U an upper triangular matrix. Realize for now still necessary triangle solver $U X = \tilde{B}$.
To do this, from the following serial (ij) form:

```
for i = n: -1:1
    for k = 1:q
        $s^{(k)} := 0$
    for j = i + 1: n
        for k = 1 : q
            $s^{(k)} := s^{(k)} + a_{ij} * x_j^{(k)}$
    for k = 1 : q
        $x_i^{(k)} := (a_{i,n+k} - s^{(k)})/a_{ii}$
```

c) For $A \in R^{m \times n}$ is the Frobenius norm $\|A\|_F$ defined as:

$$\|A\|_F := \left(\sum_{i=1}^{m} \sum_{j=1}^{n} a_{ij}^2\right)^{\left(\frac{1}{2}\right)}$$

Calculate each $\|LU - P^T A\|_F$ and $\|\widetilde{B} - UX\|_F$ For this use, if necessary, use the SelectColumns routine from task 4.

Test your method on the basis of matrices

$$A = \left(a_{(ij)}\right) \in R^{(nxn)} \quad \mathit{with} \quad a_{ij} = \frac{(i+j)}{n} * \sin\left(\frac{(ij\pi)}{(n+1)}\right)$$
$$B = \left(b_{(ij)}\right) \in R^{(nxq)} \quad \text{with} \quad b_{ij} = i + j$$

for n = 100, 500 and q = 1,10 for p = 1, 3, 4 Processors.