Assignment no. 1

Implementing dense matrix multiplication in parallel distributed memory environment

DCAMM PhD Course
Scientific Computing

DTU

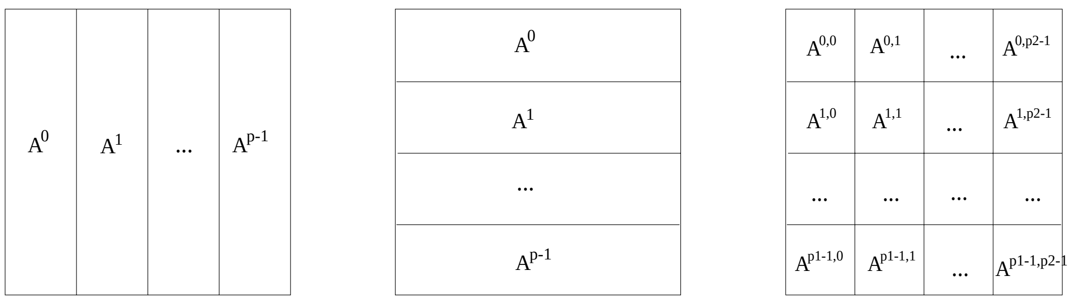
January 2008

1 Problem setting

Let two dense matrices be given, $A = \{a_{i,j}\}$ and $B = \{a_{i,j}\}$, $A, B \in \mathbf{R}^{\mathbf{n}\times\mathbf{n}}$, i.e., $i, j = 1, 2, \cdots, n$.
Let $C = AB$. As is well known, $c_{i,j} = \sum_{k=1}^{n} a_{i,k} b_{k,j}, \quad i, j = 1, 2, \cdots, n$.
The task is to design matrix×matrix multiplication procedure for performing the multiplication in a distributed memory parallel computer environment and implement the algorithm using Fortran/C/C++ and MPI.
The implementation must be independent of the size of the matrices n and the number of processors p to be used, thus, n and p must be input parameters.

2 Data generation and partitioning strategies

Create your matrices with entries initialized as random double precision numbers.
Choose the partitioning strategy which you find most suitable for the target computer architecture, for the operation (matrix-multiply) to be performed, and for the programming language you are using. There is no restriction both matrices to be partitioned using the same strategy.
Some possible partitioning strategies are shown in Figure 1.



(a) Colum-wise partitioning (b) Row-wise partitioning (c) Checkerboard partitioning

Figure 1:

2.1 Colum-wise (1D-type) partitioning

The matrices A and B are both partitioned in blocks, containing $r = n/p$ columns (Figure 1(a)). (We assume that p is a divisor of n.) C is computed in the same format. Then, the blocks $A^{(s)}, B^{(s)}, C^{(s)}$ are local for processor P_s, $s = 0, 1, \cdots, p-1$. Processor P_s must compute

$$c_{i,j} = \underbrace{\sum_{k=1}^{(s-1)r} a_{i,k}b_{k,j}}_{\text{comm.required}} + \underbrace{\sum_{k=(s-1)r+1}^{sr} a_{i,k}b_{k,j}}_{\text{local}} + \underbrace{\sum_{k=sr+1}^{(p-1)r} a_{i,k}b_{k,j}}_{\text{comm.required}}, \qquad i, j = 1, 2, \cdots, n.$$

or in block form, $C^{(s)} = AB^{(s)} = \sum_{j=0}^{p-1} A^{(j)}B^{(j,s)}, B^{(j,s)} \in \mathbf{R}^{r,r}$.

2.2 Row-wise (1D-type) partitioning

It is very similar to the column-wise partitioing (Figure 1(b)).

2.3 Checkerboard (2D-type) partitioning

It is depicted in Figure 1(c). The processors are assumed to form a 2D Cartesian grid where $p = p1 \times p2$. It is also assumed that n is divisible by both $p1$ and $p2$, namely, $r1 = n/p1$ and $r2 = n/p2$. (Described in details in the course book.)

3 Guidelines for the numerical experiments

Go up in matrix size as much as possible. You are free to use as many processors as you decide. Pay attention whether you are using the computer up to its full capacity.
When you do the timings, submit your jobs to the batch system.

4 Optional task

Find out and make a short survey on the so-called *Strassen's algorithm* for multiplying two matrices. Explain what are the advantages with respect to computational complexity and give your opinion regarding implementation and performance of this algorithm in parallel computer environment. Include a bibliography with some resent papers related to the topic. Give your opinion on the suitability of the algorithm, for which parallel platforms, and defend your opinion based on published results.

5 Writing a report on the results

The report has to have the following issues covered:

1. Problem description
2. Theoretical part
   (a) Description of the chosen partitioning strategy - why has this strategy been chosen?
   (b) Derivation of the computational and communication complexity of the algorithm for the chosen data mapping and logical computer geometry. Expressions for parallel runtime T_p, speedup S, efficiency E and isoefficiency function for your algorithm.
3. Numerical experiments

(a) Give a description/discussion of the parallel implementation - algorithms and data structures. What kind of communications have been used and why? Give figures for the memory requirements of your program implementation.

(b) Include table(s) of results for various problem (matrix) sizes, varying number of processors, corresponding speedup and efficiency figures.

(c) Present plots of the speedup (show also the ideal speedup on the plot). The plots cam be done using Matlab, Maple ot any other graphical tool.

(d) Analyse the speedup and efficiency results in comparison with the theoretical expectations. Does the scalability of your formulation depend on the the architectural characteristics of the machine? Make relevant comparisons with the results you obtained from the experiments.

(e) Add observations, comments how the particular computer architecture influences the parallel performance. Does the pair *algorithm - computer system* scales?

4. Conclusions, ideas for possible optimizations

The report must be written in English. A listing of the program code has to be attached to the report. Standard requirements are put on the design of the code, namely, structure and comments. You are encouraged to work in pairs. However, all topics in the assignment should be covered by each student, and you have to hand in individual reports!

6 Deadlines and credit points

The full report (paper copy!) must be submitted no later than January 30, 2008. Assignments submitted after this date will not be approved.

Success!

Maya Neytcheva

Any comments on the assignment will be highly appreciated and will be considered for further improvements. Thank you!