Universität Stuttgart
Institut für parallele und verteilte Systeme
Prof. Dr. rer. nat. Miriam Mehl
Dipl.-Ing. Florian Lindner

SS 2016

Parallel Numerics

Exercise 3: Sparse Matrices

1 Sparse Matrices and Graphs

Given the matrix

$$A = \begin{pmatrix} 4 & 0 & 7 & 0 & 0 & 0 \\ 0 & 5 & 42 & 0 & 12 & 3 \\ 0 & 4 & 7 & 2 & 0 & 5 \\ 0 & 0 & 44 & 32 & 0 & 0 \\ 0 & 0 & 0 & 0 & 99 & 8 \\ 100 & 2 & 0 & 0 & 0 & 102 \end{pmatrix}$$

i) What methods we know to store sparse matrices? What are advantages and disadvantages? Show how to store matrix A using the various formats.

ii) What graph datastructure can be used to represent the sparsity pattern of a (non-)symmetric matrix? How many nodes are needed?

iii) What is an adjacency matrix? How can it be obtained? How does it relate to the beforementioned graph?

iv) Give the graph of matrix A. Based on the properties of the matrix, what kind of graph you need?

v) How can not only the adjacency of a matrix be saved in a graph, but also the values?

2 Parallel QR

We consider tall and skinny QR as used, e.g. in the first step of block-wise parallel QR decomposition.

$$\overbrace{\begin{bmatrix} \\ \\ A \\ \\ \\ \end{bmatrix}}^{m} = \begin{bmatrix} \\ \\ Q \\ \\ \\ \end{bmatrix} \cdot \begin{bmatrix} R \end{bmatrix}, \quad A \in \mathbb{R}^{n \times m}$$

based on the Gram-Schmidt orthonormalization as described in the lecture notes in 4.2. For parallelization, we use a decomposition of A und Q into p row-blocks. Processor i in a distributed memory system stores A_i and Q_i. R is stored at the master process ($i = 1$).

$$\begin{bmatrix}\boxed{A_1}\\ \boxed{A_2}\\ \vdots\\ \boxed{A_p}\end{bmatrix} = \begin{bmatrix}\boxed{Q_1}\\ \boxed{Q_2}\\ \vdots\\ \boxed{Q_p}\end{bmatrix} \cdot [R]$$

i) Calculate the communication costs, i.e.

   a) The number of messsages and

   b) the amount of data (number of matrix entries)

   for each of the five algorithmic steps and for the whole QR-decomposition of A.

ii) Compare i with the communication avoiding QR-algorithm for $p = 2^q$, which calculates small QR-decomposition for the blocks of A independently and combines them to larger blocks in a fan-in-like process

$$\begin{bmatrix}\boxed{A_1}\\ \boxed{A_2}\\ \boxed{A_3}\\ \boxed{A_4}\end{bmatrix} \Rightarrow$$

$$\begin{array}{l} p=1: \ \boxed{A_1} = \boxed{Q_1}\boxed{R}\\ p=2: \ \boxed{A_2} = \boxed{Q_2}\boxed{R}\\ p=3: \ \boxed{A_3} = \boxed{Q_3}\boxed{R}\\ p=4: \ \boxed{A_4} = \boxed{Q_4}\boxed{R}\end{array} \Rightarrow \begin{array}{l} p=1: \ \begin{pmatrix}\boxed{Q_1} & \\ & \boxed{Q_2}\end{pmatrix}\begin{pmatrix}\boxed{R_1}\\ \boxed{R_2}\end{pmatrix} = \begin{pmatrix}\boxed{Q_1} & \\ & \boxed{Q_2}\end{pmatrix}\begin{bmatrix}\bar{Q}_1\end{bmatrix}\begin{bmatrix}\bar{R}_1\end{bmatrix}\\ p=3: \ \begin{pmatrix}\boxed{Q_3} & \\ & \boxed{Q_4}\end{pmatrix}\begin{pmatrix}\boxed{R_3}\\ \boxed{R_4}\end{pmatrix} = \begin{pmatrix}\boxed{Q_3} & \\ & \boxed{Q_4}\end{pmatrix}\begin{bmatrix}\bar{Q}_3\end{bmatrix}\begin{bmatrix}\bar{R}_3\end{bmatrix}\end{array} \Rightarrow$$

$$p=1: \ \begin{pmatrix}\boxed{Q_1} & & & \\ & \boxed{Q_2} & & \\ & & \boxed{Q_3} & \\ & & & \boxed{Q_4}\end{pmatrix}\begin{pmatrix}\boxed{\bar{Q}_1} & \\ & \boxed{\bar{Q}_3}\end{pmatrix}\begin{pmatrix}\boxed{\bar{R}_1}\\ \boxed{\bar{R}_3}\end{pmatrix} =$$

$$\underbrace{\begin{pmatrix}\boxed{Q_1} & & & \\ & \boxed{Q_2} & & \\ & & \boxed{Q_3} & \\ & & & \boxed{Q_4}\end{pmatrix}\begin{pmatrix}\boxed{\bar{Q}_1} & \\ & \boxed{\bar{Q}_3}\end{pmatrix}\begin{bmatrix}\bar{\bar{Q}}_1\end{bmatrix}}_{Q}\boxed{R}$$

iii) Implement both variants, compare scalability and runtimes for $m = 100, 3200$ and $p = 1, 2, 4, 8, 16, 32$.