UNIVERSITÄT ZU LÜBECK
INSTITUT FÜR TECHNISCHE INFORMATIK

Exercise Sheet 1

Think Parallel

Lecture *Parallel Computing Systems*, Winter semester 2024/2025

Dr. Javad Ghofrani
A discussion forum for the exercise can be found at: `moodle.uni-luebeck.de`.

Submission Guideline

Please submit assignments according to the following instructions:

1. Please zip your submission in a single file named: "PARMAI_sheet1_YOURLASTNAME.zip"
2. Provide plots where applicable (accepted file formats for plots: png and jpg)
3. Include all your code in C/C++, Python, Java, Matlab (or ask me before), with a readme of how to compile & start
4. Include a video where you go through and explain solution for each task (You can use OBS Studio to capture your screen)

Think Parallel (20 pt.)

Given the following array x composed of 8 distinct elements:

2 4 6 8 1 3 5 7

1. Develop a sequential algorithm and a parallel algorithm to calculate the Prefix Sum. (Try to find the fastest one!). **Reference :** Guy E. Blelloch, Prefix Sums and Their Applications,School of Computer Science Carnegie Mellon University Pittsburgh.
2. Calculate for each approach the number of time steps, the number of operations and the number of required CPUs.

Note : Arranging your work as on the lecture slides would be preferable

Parallel Algorithms (20 pt.)

In this exercise, we will be dealing with the scalar product of two vectors.
Given A and B, two vectors with 160 elements each.

1. Illustrate in a scheme or a simple algorithm the sequential scalar product of A and B.
2. Re-do the previous task using 8 processors (parallel processing).
3. How many time steps are required for sequential and parallel processing?
4. Calculate the speed-up S_8 and the efficiency E_8 for both approaches.
5. Generalize S_p and E_p as functions of the vectors length **n** and the number of used processors **p**.

PRAM - Parallel Random Access Machine

Matrix Multiplication (20 pt.)

Given A, B and C, three dense matrices of size N x N (A dense matrix is a matrix in which most of the entries are nonzero).
A matrix-matrix multiplication is illustrated as follows :

$$\begin{bmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,n} \\ a_{2,1} & a_{2,2} & \dots & a_{2,n} \\ \vdots & \vdots & \dots & \vdots \\ a_{n,1} & a_{n,2} & \dots & a_{n,n} \end{bmatrix} \times \begin{bmatrix} b_{1,1} & b_{1,2} & \dots & b_{1,n} \\ b_{2,1} & b_{2,2} & \dots & b_{2,n} \\ \vdots & \vdots & \dots & \vdots \\ b_{n,1} & b_{n,2} & \dots & b_{n,n} \end{bmatrix}$$

$$= \begin{bmatrix} c_{1,1} & c_{1,2} & \dots & c_{1,n} \\ c_{2,1} & c_{2,2} & \dots & c_{2,n} \\ \vdots & \vdots & \dots & \vdots \\ c_{n,1} & c_{n,2} & \dots & c_{n,n} \end{bmatrix}$$

$$c_{i,j} = \sum_{k=1}^{n} a_{i,k} * b_{k,j}$$

1. Modify the algorithm to adapt it to CRCW PRAM Model. Calculate the time complexity and the number of used processors.
2. Design an algorithm that works with **O** (n^2) processors. Calculate the time complexity, the speed-up and the efficiency. Compare and discuss the results.

Distributed Maximum Search (20 pt.)

Consider an array of **n** distinct elements.
The task is to search the maximum value in the array. We know that a sequential algorithm will always have a worst-case running time of $\mathbf{O}(n)$.

1. Given p processors, derive an efficient parallel algorithm for this task using the following models :
   - **EREW-PRAM**
   - **CREW-PRAM**
   - **CRCW-PRAM**
2. Determine the time complexity for each of the previous algorithms.

Programming Assignment (20 pt.)

Implement the above 4 problems in your prefered programming language, then record yourself running and explaining them.

Requirements for the implementation of each problem:

1. Implement sequential and parallel solution to process the same input.
2. The speedup is presented visually.
3. The records should clearly show your source code, results and explanation.