Technische Universität München
Institut für Informatik
PD Dr. Ralf-Peter Mundani
Philipp Samfass, M.Sc.

SoSe 2018

High Performance Computing - Programming Paradigms and Scalability

Exercise Sheet 4: Parallel Structures, Load Balancing

1 Matrix Transposition

The sequential transposition of a matrix A of size $N \times N$ works as follows:

```
1     int    i, j;
2     double temp;
3
4     for(i = 0; i < N; i++) {
5         for(j = 0; j < i; j++) {
6             temp    = A[i][j];
7             A[i][j] = A[j][i];
8             A[j][i] = temp;
9         }
10    }
```

a) In order to compute A^T in parallel, one can choose between function, data, and competitive parallelism. Depending on the underlying hardware (shared or distributed memory) these approaches have different strong and weak aspects. Give an implementation idea for each approach and discuss your results.

b) Compute for a data parallel approach with column-wise data decomposition the speed-up depending on N and p (the amount of processes) only! Exchanging two local elements $A(i, j)$ and $A(j, i)$ takes t_{ex} time, transmitting one element from process p_i to process p_j takes $t_{com} = 1.5 \cdot t_{ex}$ time. For simplification you can transmit data from p_i to p_i itself). What can you observe for the parallel efficiency?

2 Diffusion Model

Given is a 2-dimensional *Illiac* mesh with 4×4 nodes (see figure 1). The single nodes have an initial load situation, i.e. amount of tasks still to be processed, as follows:

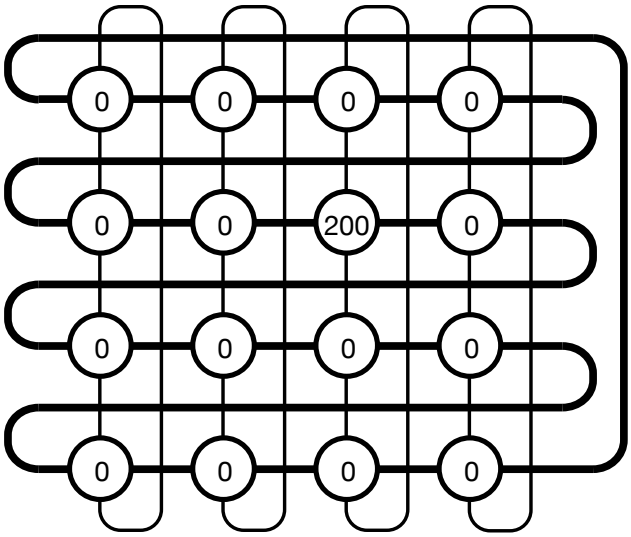

Figure 1: 4×4 Illiac mesh.

Compute the first two load balancing steps based on a diffusion model for $\alpha = 0.1$ using total-step iteration (i.e. for $w_i^{(t+1)}$ only values $w_i^{(t)}$ are used) as update scheme!