Laboratory for Parallel Programming
Prof. Dr. Felix Wolf

TECHNISCHE UNIVERSITÄT DARMSTADT

Mock exam

Large-Scale Parallel Computing

February 8, 2016

Name ______________________________

Surname ______________________________

Student number ______________________________

Study ______________________________

Intended degree ______________________________

- Please write your **name**, **surname**, and **student number** on every sheet.
- Please give your answers in readable and understandable form, and explain them properly.
- Please answer the tasks on the **task sheets** (also use back sides). Ask for additional task sheets if needed.
- Please **strike through** answers you don't want to be considered.
- Please use a **document proof pen** in black or blue color (No pencil!).
- If you are observed trying to deceive, the exam is graded as **0 points**.
- Please hand in **all sheets together with the task sheets** in the end.

Task	Pt.	OK
1.1	18	
1.2	26	
1.3	16	
1	60	

Task	Pt.	OK
2.1	25	
2.2	20	
2.3	15	
2	60	

	Task 1	Task 2	Sum
Points	60	60	120
OK			

Surname	Name	Student number

Task 1 (60 points)

Task 1.1 (6 + 3 + 3 + 3 + 3 = 18)

a) Give three costs and three benefits of parallelization (3 + 3 = 6)

Cost and benefits of parallelization

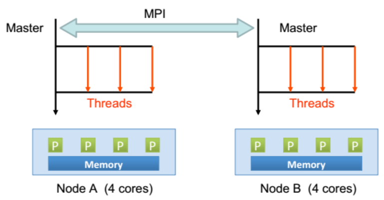
Benefit
- Speedup
- Sometimes improved interactivity and cleaner design (separation of concerns)
- More aggregate memory (distributed memory parallelization)

Cost
- Programming effort
- Program complexity
- Overhead (communication & synchronization)
- Bugs
- Potentially non-determinism
- Extra dependencies (library, compiler)

b) What is hybrid parallel programming? (3)

Hybrid programming

- Between processes parallelism via MPI
- Thread parallelism inside a process (e.g., via OpenMP)



c) What is peak performance? (3)

- Peak performance is the performance a computer is guaranteed not to exceed

d) What is spatial locality? (3)

Locality

- Programs tend to reuse data and instructions they have used recently
  - Allows to estimate what instructions and data a program will use in the near future
- A program typically spends 90% of the time in 10% of its code
- Locality also applies to data accesses, but usually not as much as to code accesses

Temporal locality	Spatial locality
Recently accessed items are likely to be accessed in the near future	Items with addresses close to each other tend to be referenced close together in time

Surname	Name	Student number

e) What is weak scaling? (3)

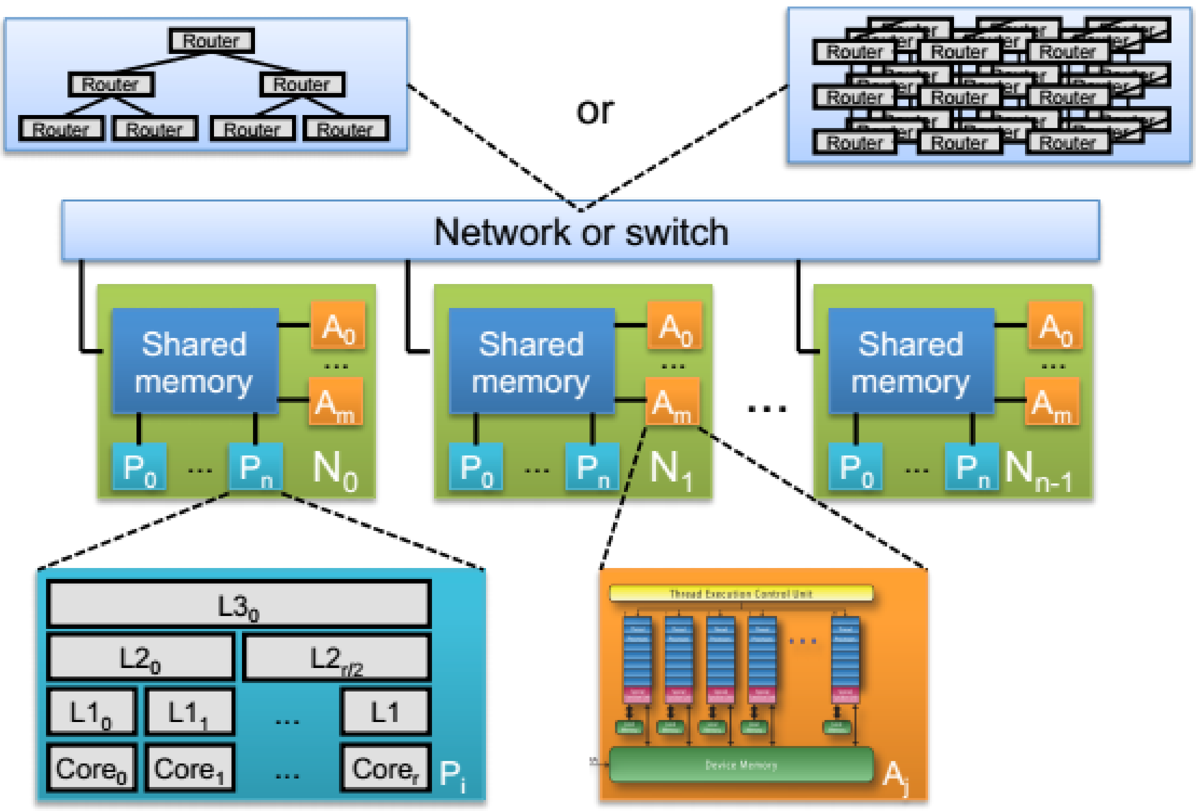
Scalability

Weak scaling
- Ability to solve a larger input problem by using more resources (here: processors)
- Problem size per processor remains constant
- Example: larger domain, more particles, higher resolution

Strong scaling
- Ability to solve the same input problem faster as more resources are used
- Usually more challenging
- Limited by Amdahl's law and communication demand

Task 1.2 (8 + 10 + 8 = 26)

a) Draw the architecture of a typical supercomputer (8)

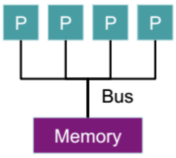


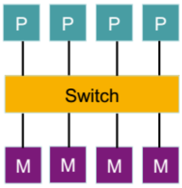
Surname	Name	Student number

b) What are the two types of shared memory systems? What are their pros and cons. Draw a figure of each. (10)

UMA (Uniform memory access)

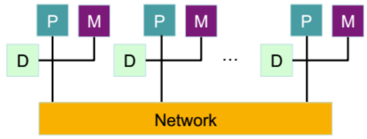
- Each CPU has same access time to each memory address
- Simple design but limited scalability (multicore or less)





NUMA (Non-uniform memory access)

- Memory has affinity to a processor
- Access to local memory faster than to remote memory
- Harder to program but more scalable



c) Describe Flynn's classification of computer architectures. (8)

Flynn's classification [1966]

- Single instruction stream, single data stream
  - Classical uniprocessor
- Single instruction stream, multiple data streams
  - Same instruction is executed by multiple processors using different data streams
  - Data parallelism
  - Examples: SIMD extensions for multimedia, vector processors
- Multiple instruction streams, single data stream
  - No commercial multiprocessor of this type ever built
- Multiple instruction streams, multiple data streams
  - Each processor fetches its own instructions and operates on its own data
  - Thread-level parallelism

Surname	Name	Student number

Task 1.3 (6 + 6 + 4 = 16)

a) Name and describe three metrics for network evaluation (6)

b) What is Amdahl's law? Based on Amdahl's law, how is the speedup calculated for single and multiple processor configurations. (6)

$$\text{Speedup}_{\text{overall}} = \frac{1}{(1-\text{Fraction}_{\text{enhanced}}) + \frac{\text{Fraction}_{\text{enhanced}}}{\text{Speedup}_{\text{enhanced}}}} < \frac{1}{(1-\text{Fraction}_{\text{enhanced}})}$$

Example

- Function foo() of a program takes 20 % of the overall time
- How is the speedup if the time needed for foo() can be halved?

$$\text{Speedup}_{\text{overall}} = \frac{1}{(1-0.2)+\frac{0.2}{2}} = \frac{10}{9}$$

- How is the speedup if the time needed for foo() can almost be eliminated?

$$\text{Speedup}_{\text{overall}} = \frac{1}{(1-0.2)+\frac{0.2}{\infty}} = \frac{10}{8}$$

c) What should be the size of the sequential region to achieve a speedup of 50 on 80 processors? (4)

Surname	Name	Student number

Task 2 (60 points)

Task 2.1 (5 + 5 + 15 = 25)

a) What is the difference between the size and extent of an MPI data type? (5)

b) Create an MPI data type that covers the diagonal of a square matrix of type double. Assume the length of each side of the matrix to be `DIM_LEN = 4*N`. However, instead of `4*N`, the datatype should only cover `N` consecutive elements on the diagonal. Note: Provide your solution in the C language. Only code for the creation of the MPI data type is necessary. (5)

Surname	Name	Student number

c) Now assume that a `4*N` $\times$ `4*N` matrix is stored on process 0. Use the data type described above to distribute the diagonal of this matrix equally among processes of the MPI program. The total number of processes is 4. (15)

Task 2.2 (5 + 5 + 10 = 20)

a) What are the necessary conditions to write a custom reduction function? Is the order of invocation affected by any property? (5)

Surname	Name	Student number

b) Write a reduction operator that calculates the LCM (least common multiple). Assume there is a function
`int mat_lcm(int val1, int val2)`
already defined that calculates the LCM of `val1` and `val2`. (5)

c) Assume that in an MPI program, each process has `5` values stored in an integer array `val_array`. Using the above function, write a program such that each process finds the LCM of all the values in the program. At the end, there should be only 1 value which is the LCM of all the values and this value should be known to each process. (10)

Surname	Name	Student number

Task 2.3 (5 + 10 = 15)

a) Write the necessary MPI code to create a ring topology of processes. Assume there are in total `N` processes. (5)

b) Write the necessary MPI code, such that in the ring topology created above, each process prints its rank in `MPI_COMM_WORLD`. The algorithm should be such that, rank 0 (in `MPI_COMM_WORLD`) passes around a token message in the ring topology. When a process receives this token, it prints its `MPI_COMM_WORLD` rank and forwards the token to its neighbor in the ring. Process 0 starts with printing its rank as it already has the token. (10)