Parallel programming

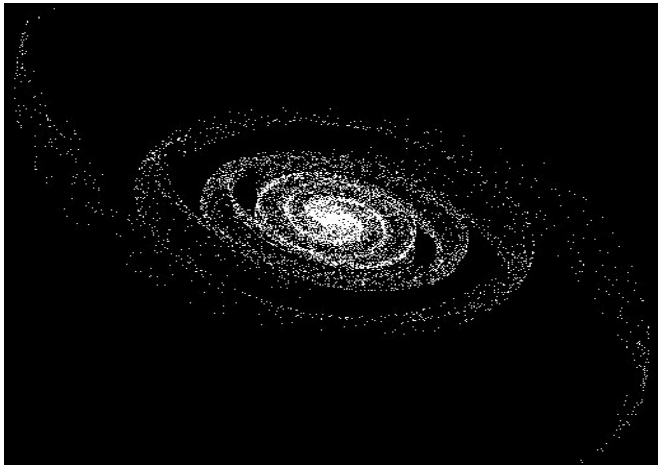
Semester project

2D Gravitational N-body simulation

FAKULTA
ELEKTROTECHNICKÁ
ČVUT V PRAZE

What is n-body simulation

- N-body simulation
  - simulation of a **dynamical** system of **particles** (in our case planets)
  - usually under the influence of **physical forces**, such as gravity
  - **iterative** algorithm
    - computing **differences** between two discrete consecutive time instants



Semester project assignment

- Implement a **sequential** and **parallel version** of the algorithm for 2D Gravitational N-Body simulation
  - Feel free to experiment! E.g., add stationary objects (stars)
- Export resulting simulation to gif
  - Simple library for gif creation can be found here: *https://github.com/ginsweater/gif-h*
- Prepare the **presentation** to show achieved results

Input instance - example

You can use the provided instance generator

```
3
10.235 18.654 33.55564
6.15 65.12 34.5064
198.654 0.215 15567.324
```

- Description of an instance:
  - First line: integer **number of stars** in the instance
  - All other lines represent stars
    - **x-coordination** of the star (double)
    - **y-coordination** of the star (double)
    - **mass** of the star (double)

Evaluation of semester project

- Functional code (2 points)
  - Functional sequential and parallel code (choose either C++11, OpenMP, OpenMPI).
  - Show an animation of the resulting simulation.
  - Code is tested by the lab teacher before the handover.
- Profile the bottlenecks for sequential code (2 points)
  - Profiler outputs (Intel Parallel Studio), detection of bottlenecks, analysis.
- Parallel code executed on Metacentrum + Xeon Phi (2 points)
  - Scalability and performance graphs and other performance metrics.
  - Metacentrum: PBS scripts, hardware info.
- Reproducibility of your experiments (2 points)
  - Benchmark scripts, description of used hardware and software.
  - A text file briefly describing how to carry out experiments.
- Presentation of your work (2 points)
  - Does your presentation satisfy formal requirements?
  - Is your presentation gripping to others (oral speech + slides)?
- General impression (4 points)
  - Have you used an algorithm with a better algorithmic complexity?
  - Is the performance of the optimized code outstanding?
  - Other aspects that may improve the general impression.

Requirements for your presentation

- **Base structure**:
  - Introduction
  - Profiling/bottleneck analysis
  - Scalability graph
  - Performance graph
  - Validity of the results
  - Discussion and conclusion
- The presentation should have at max **10 slides**.

Presentation - Introduction

- Describe the algorithm you used for simulation
- Mention the used technology (C++11 threads, OpenMP, OpenMPI)

Presentation - Profiling

- Describe
  - The bottleneck of the sequential algorithm
  - Include a profiler analysis from Intel Parallel Studio (graphs, excerpts from text files) indicating the bottleneck
  - The parts of the algorithm which is possible to parallelize

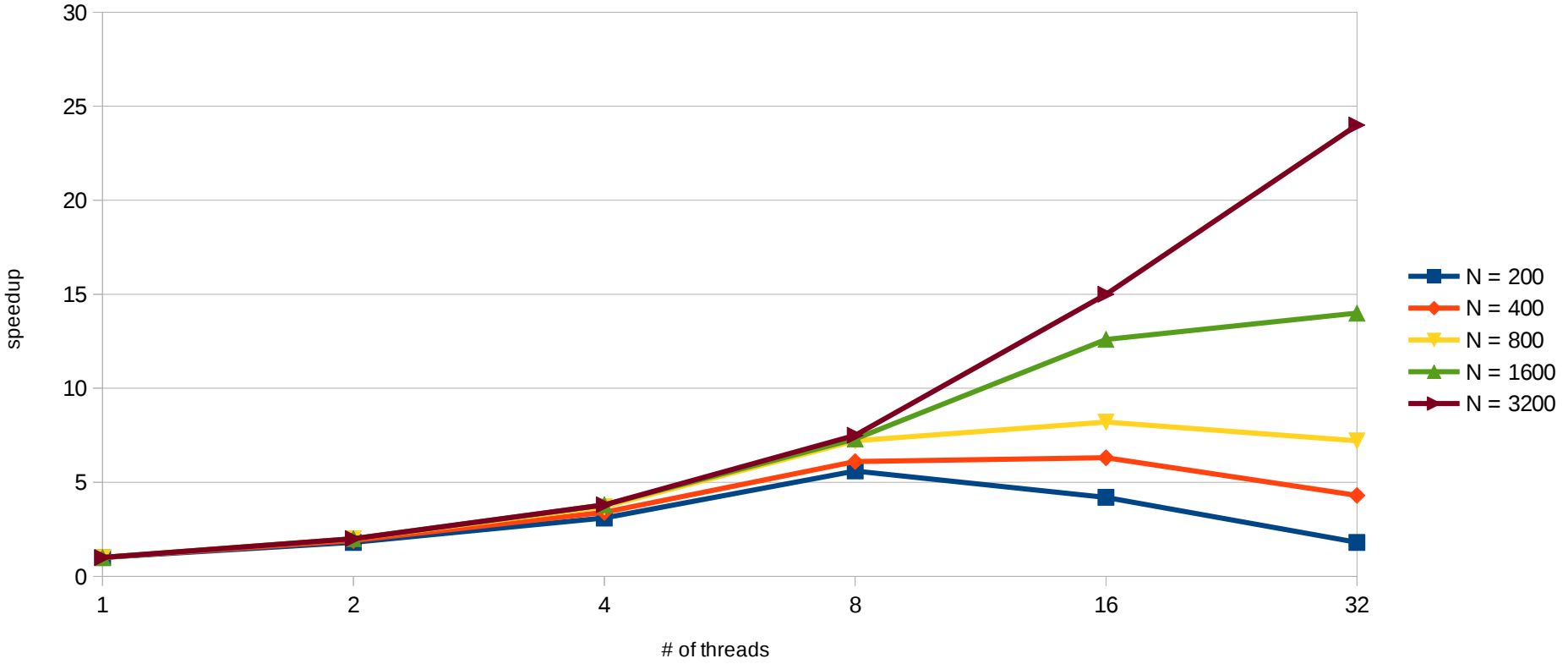
Presentation – graphs

- Graphs:
  - 1) Speedup of Parallel CPU version (e.g., Xeon Phi) vs Sequential version (**scalability graph**)
  - 2) Graph showing the algorithm runtime based on the size of an input instance (**performance graph**).
- Each graph should have a title, legend, and an appropriate format of axes (+units)
- Description of the hardware and software

Presentation – Scalability graphs

- Shows the speedup with respect to the number of used threads
  - Scalability graph for up to 256 threads

Presentation – Scalability graph

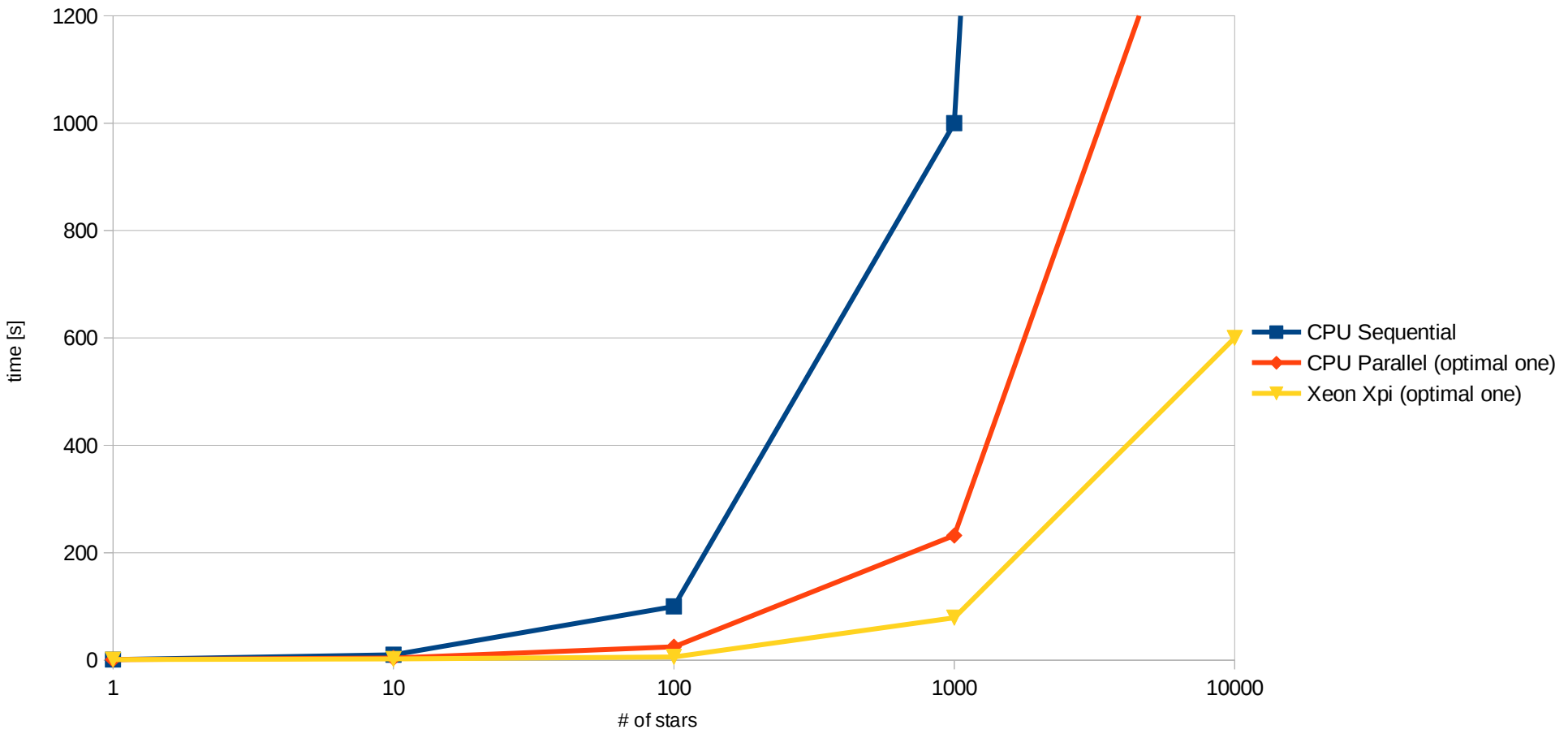


- N – number of stars
- For constant number of simulation iterations (e.g., 100 iterations)

Presentation – Performance graph

- Shows how much time the algorithm takes to finish the computation depending on the number of stars
- Set the reasonable upper threshold for measuring (e.g., 5 minutes)

Presentation – Performance graph



Presentation

Validity of the results

- Show the resulting gif image (5-10 seconds).

Discussion and conclusion

- Discuss the achieved results.
- Explain what was the most complicated part and why the results are as provided.
- What is the limiting factor of the parallelisation in your algorithm.