Exercise 1 - Feedback

High Performance Computing for Science and Engineering

October 10, 2014

CSElab
Computational Science & Engineering Laboratory
http://www.cse-lab.ethz.ch

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Reporting Performance

- Given a diffusion solver using finite differences:
  - A takes 5s per timestep
  - B takes 1.8s per timestep
- Which one would you choose?

Reporting Performance

- Given a diffusion solver using finite differences:
  - A takes 5s per timestep with a single thread @ 233MHz
  - B takes 1.8s per timestep with multithreading @ 3GHz
- Which one would you choose?

Reporting Performance

- Context is everything!
  - What is the hardware setting?
    - CPU model, number of cores, memory bandwidth, GPU model, network,...
  - What about the software?
    - compiler, version, optimization flags
    - precision used: double/float/half?

Reporting Results

- Write the parameters used in the simulations:
    - it helps the reader reproduce the results
    - it helps you relearn the code when you get back at it
    - it helps debugging: not all values are equally good!

There is no bug free code

- Don’t be afraid of showing wrong results!
- Discuss what is not working and show it to us!
  - It might help in looking for bugs
  - It might help you finding out what to try next or see if you forgot something

Exercise 3

High Performance Computing for Science and Engineering

October 10, 2014

CSE**lab**
Computational Science & Engineering Laboratory
http://www.cse-lab.ethz.ch

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Von Neumann Stability Analysis

Used to understand stability of finite difference schemes

Assumes a solution (based on Fourier decomposition):

1D $\quad \underbrace{u_j^n = \rho^n}_{\text{time}} \underbrace{e^{ikx_j}}_{\text{space}}$ 2D $\quad u_{r,s}^n = \rho^n e^{ik_x x_r} e^{ik_y y_s}$

Bound if $\rho < 1$

Substitute assumption into FD scheme and find the condition for which the solution is bound at any time n

Performance evaluation of parallel algorithms

Execution time on p processors: ***T(p)***

- *T(1)* is the best time on a single CPU core
- *p T(p) >= T(1),* as otherwise running the parallel program on one core will give shorter time than *T(1)* => contradiction

Speed up: ***S(p) = T(1) / T(p)***

- clearly *S(p) <= p*
- *S(p) < 1* is possible, means computation on p cores is slower than on a single core. While not advisable we might be forced to still use such a parallel code for memory reasons.

Strong scaling

Strong scaling: defines how the solution time varies with increasing number of processors p for a **fixed total problem size** (i.e. fix workload is split among cores):

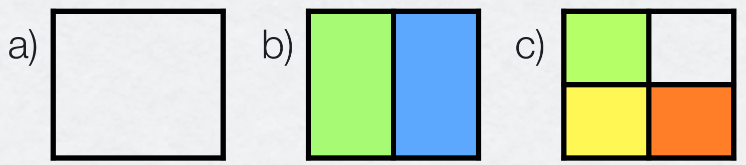
- **Speedup** for strong scaling:

$$S(p) = \frac{T(1)}{T(p)}$$

- **Efficiency** for strong scaling:

$$E_s(p) = \frac{S(p)}{p}$$

T(1) = time of one thread to process the data
T(p) = time of p threads to process the same data

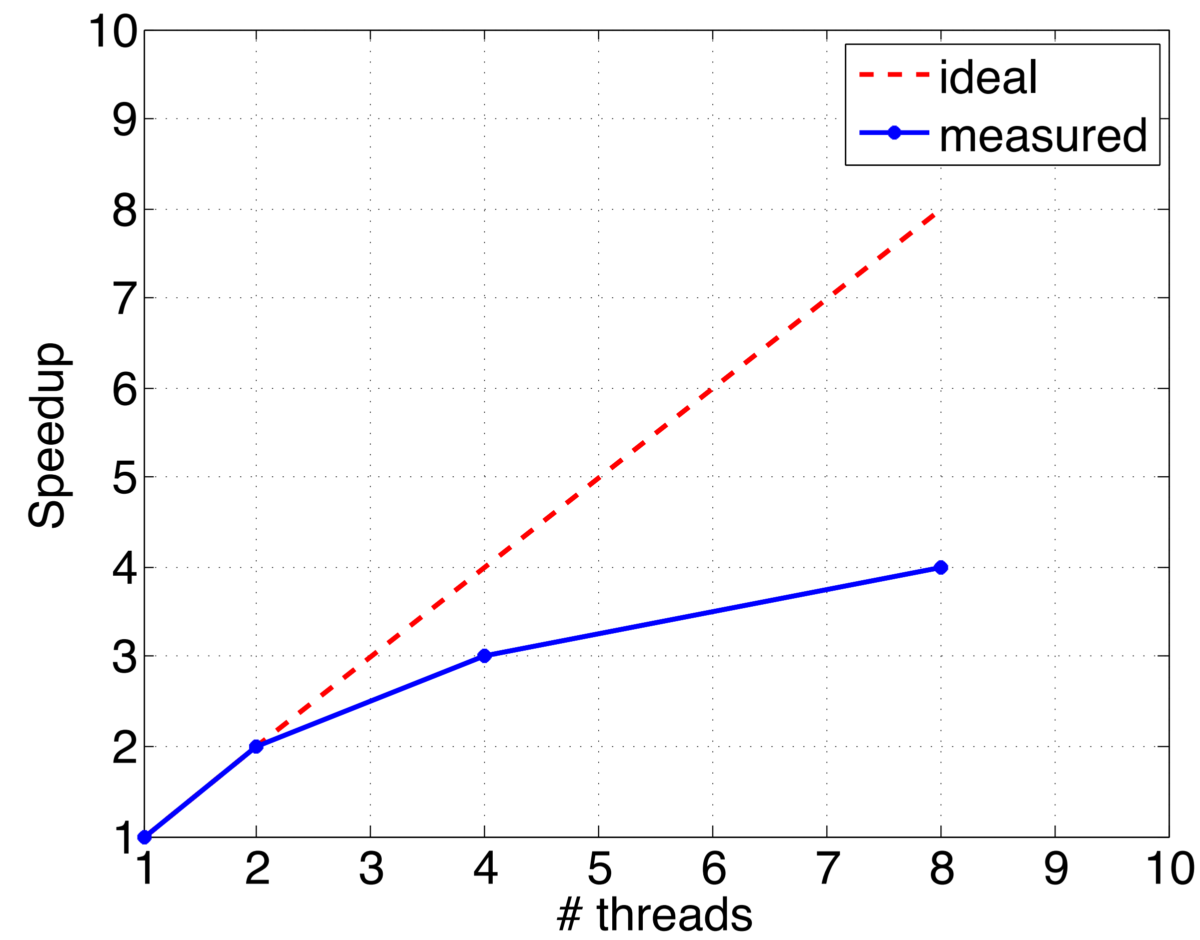


Split of the fixed problem size over a) 1 core, b) 2 cores, c) 4 cores

Example:

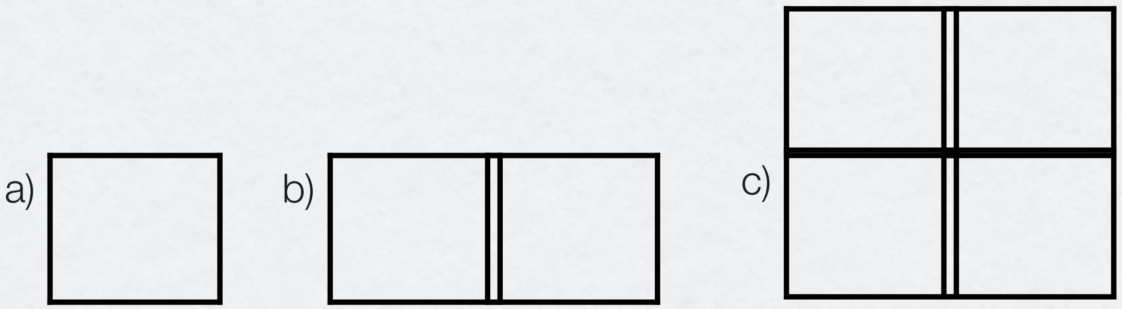
System with fix problem size N (e.g. # particles)

Problem size:	Execution time:	Speedup:
N = 100	T(1) = 12.0 s	S(1) = 1.0
N = 100	T(2) = 6.0 s	S(2) = 2.0
N = 100	T(4) = 4.0 s	S(4) = 3.0
N = 100	T(8) = 3.0 s	S(8) = 4.0



Weak scaling

Weak scaling: defines how the solution time varies with the number of processors p for a fixed **problem size per processor**



Problem size with a) 1 core, b) 2 cores, c) 4 cores

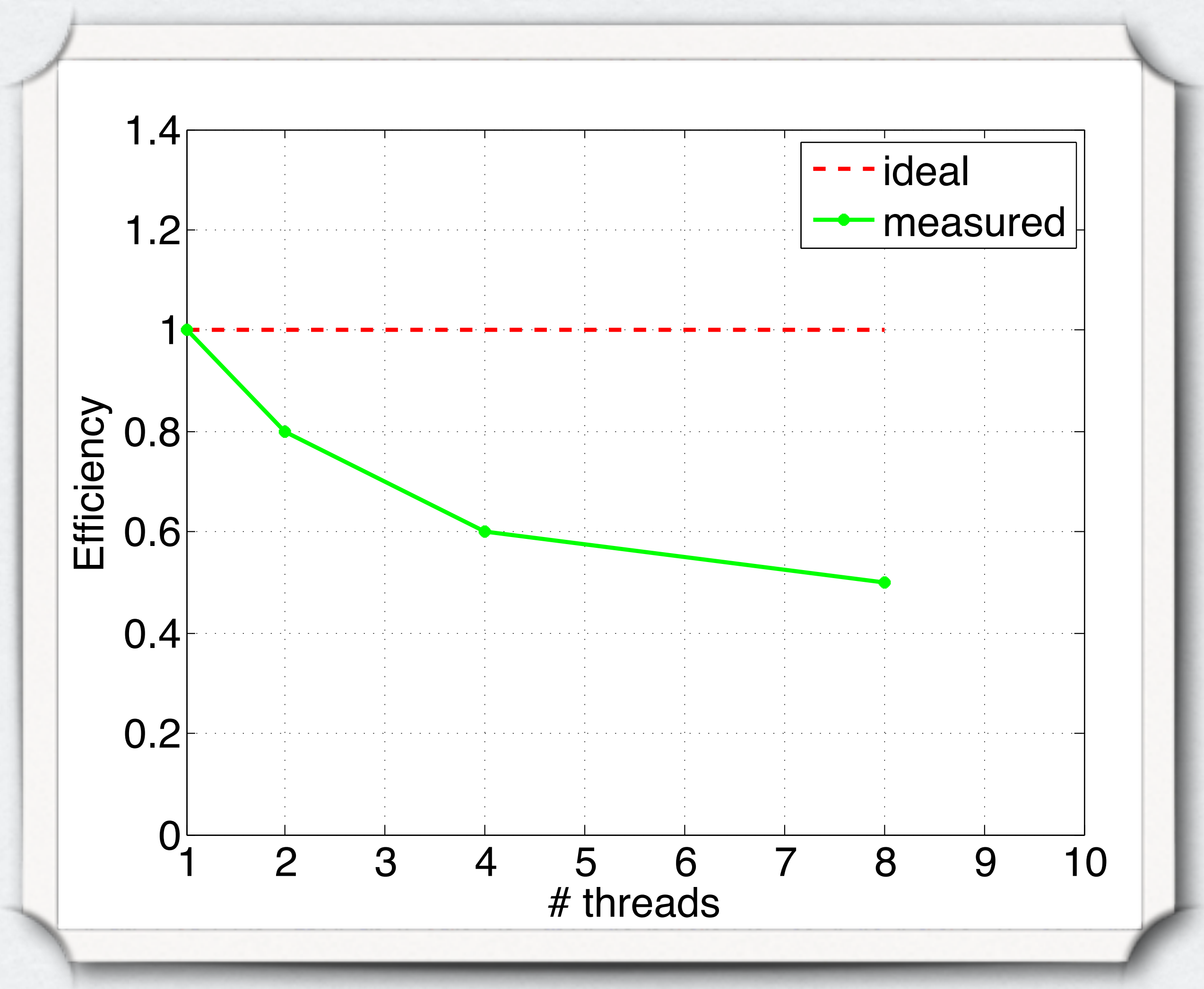
- Speedup for weak scaling doesn't make sense
- **Efficiency** for weak scaling:

$$E_w(p) = \frac{T(1)}{T(p)}$$

T(1) = time of one thread to process the data
T(p) = time of p threads to process p times the data

Example:

Problem size:	Execution time:	Efficiency:
N = 100	T(1) = 12 s	E(1) = 1.0
N = 200	T(2) = 15 s	E(2) = 0.8
N = 400	T(4) = 20 s	E(4) = 0.6
N = 800	T(8) = 24 s	E(8) = 0.5



Report

- Hand-in
    - Code
    - PDF with results and comments
    - Optional: movie of the simulation (VMD, OpenGL,...)
- Rule: feedback is proportional to the effort in reporting
    - Even if something does not work, tell us about it!