Introduction to Parallel Programming and OpenMP

Ilya A. Kaliman

August 1, 2017

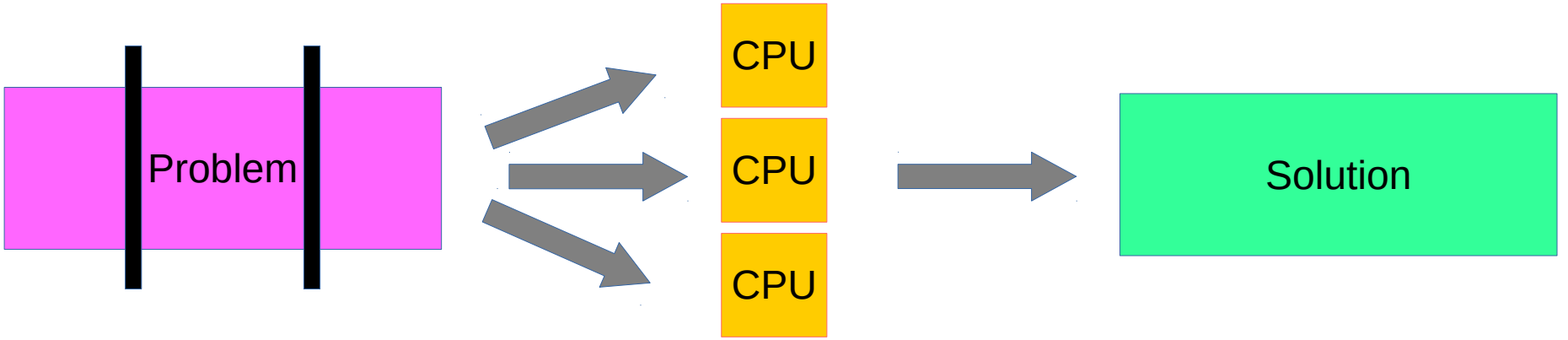
Outline

- Lecture on the basics of parallel programming and an introduction to OpenMP
- Hands-on OpenMP tutorial
- Work on your own project

What is parallel computing?

Serial computing

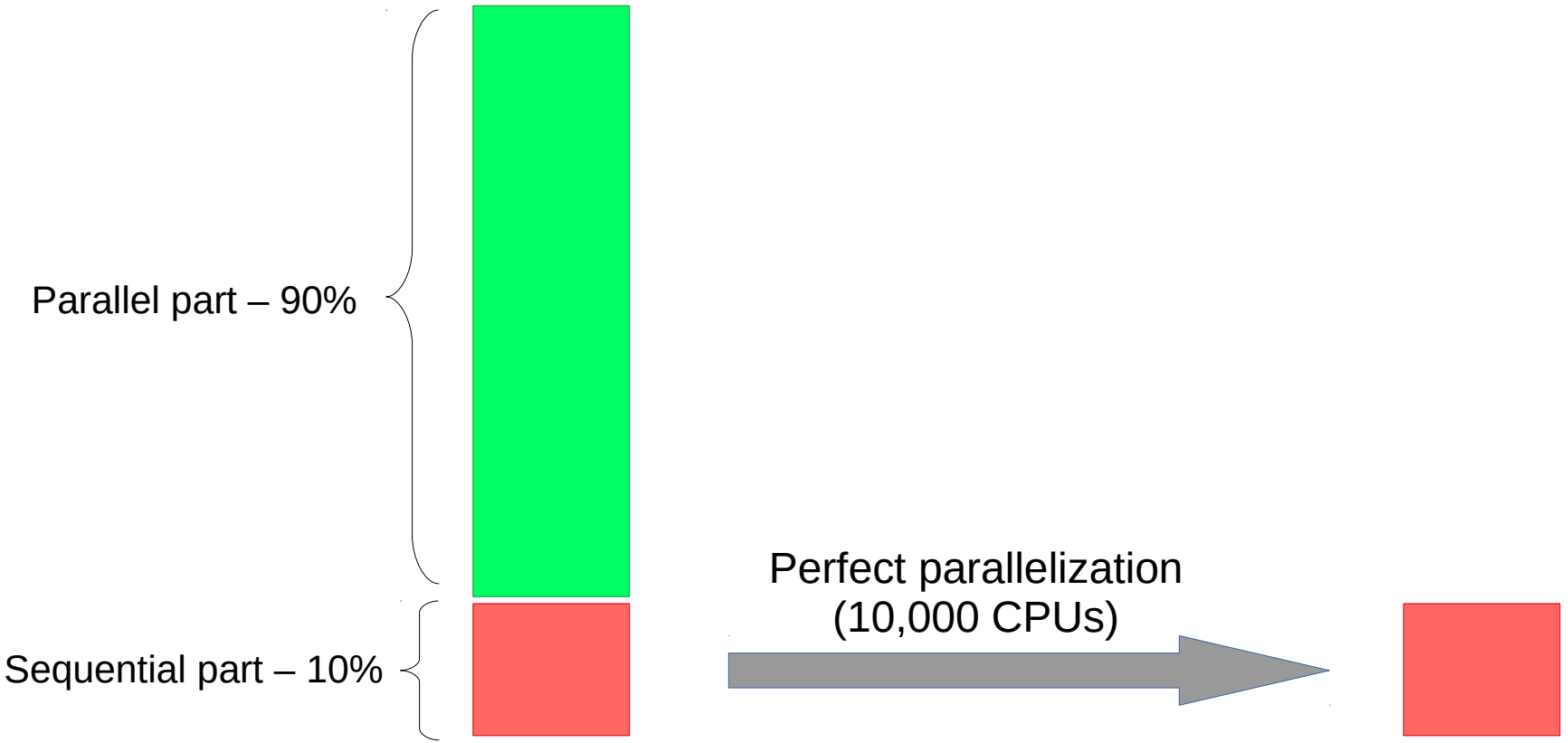
Parallel computing



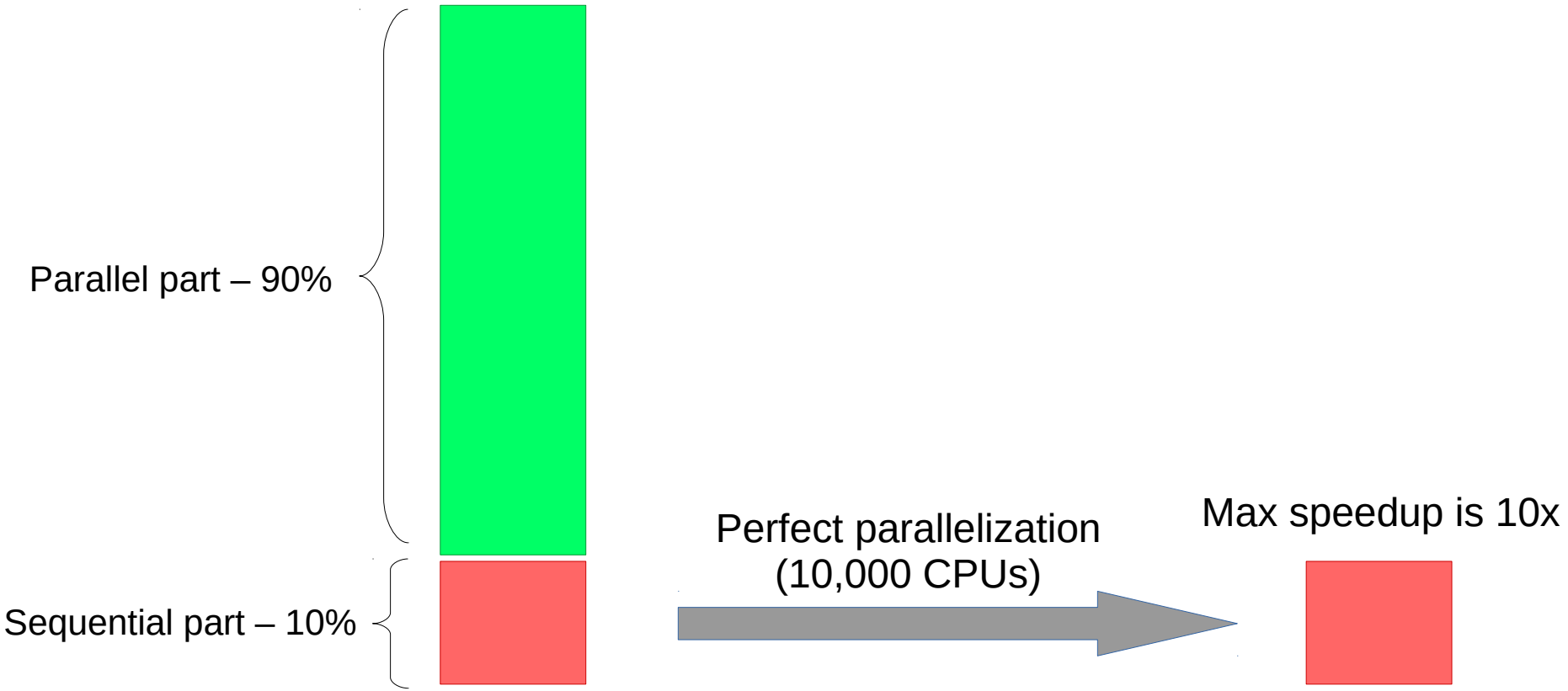
Why parallelize?

- Solve problems in less time
  - Days → hours
- Solve larger problems
  - 100 atoms → 10,000 atoms
- Concurrently do several things
- Make more efficient use of hardware
  - All modern systems have multiple CPUs/cores
  - Performance of a single core does not grow

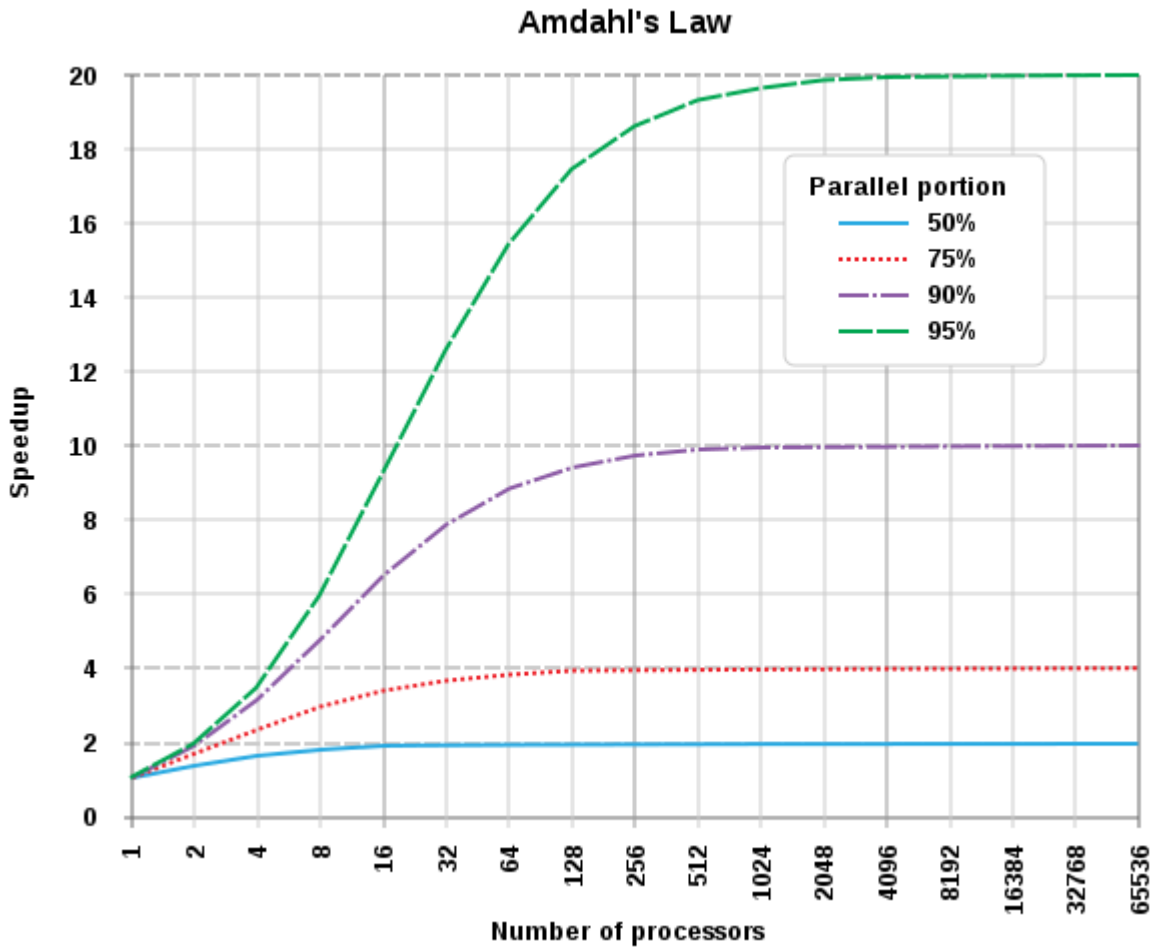
Parallelization limits: Amdahl's Law



Parallelization limits: Amdahl's Law



Parallelization limits: Amdahl's Law



$$t(P) = t_{seq} + \frac{t_{par}}{P}$$

$$Speedup(P) = \frac{t(1)}{t(P)}$$

t_{seq} – sequential portion
t_{par} – parallel portion
P – number of CPUs

Figure: wikipedia

Parallel programming challenges

- Dead/live locks
- Race conditions
- Resource oversubscription
- Algorithm/data structure design
- Complicated debugging
- ...

Simplest form of parallelism: bash

- bash $: ./prog input1 > output1 &
- bash $: ./prog input2 > output2 &
- bash $: ./prog input3 > output3 &
- bash $: ./prog input4 > output4 &

Parallel Programming Technologies

- Programming languages
  - Cilk, Chapel, Coarray Fortran, Charm++, ...
- Distributed systems
  - Message Passing Interface (MPI)
- Heterogeneous
  - OpenCL, CUDA
- Shared-memory
  - OpenMP

OpenMP

- OpenMP is an Application Programming Interface (API)
- Allows to develop **portable** and **scalable** applications on shared-memory platforms
- Consists of
  - Compiler directives (pragmas)
  - Runtime library (functions omp_xxx_xxx, omp.h)
  - Environment variables (export OMP_NUM_THREADS=4)
- Supports
  - C and C++
  - Fortran

Why OpenMP?

- Standard
- Well-established (supported by most C, C++, Fortran compilers)
- Relatively easy to use
- Straightforward path from sequential program to a parallel one
- Supported virtually everywhere (supercomputers)

Standards

- http://www.openmp.org
- Standards
  - OpenMP 4.5 (Nov 2015) – latest
  - OpenMP 3.1 (July 2011)
  - ….
  - OpenMP 1.0 (April 1999)
- Usually backwards compatible (adds new features without changing the behavior of older programs)

OpenMP: fork-join parallel model

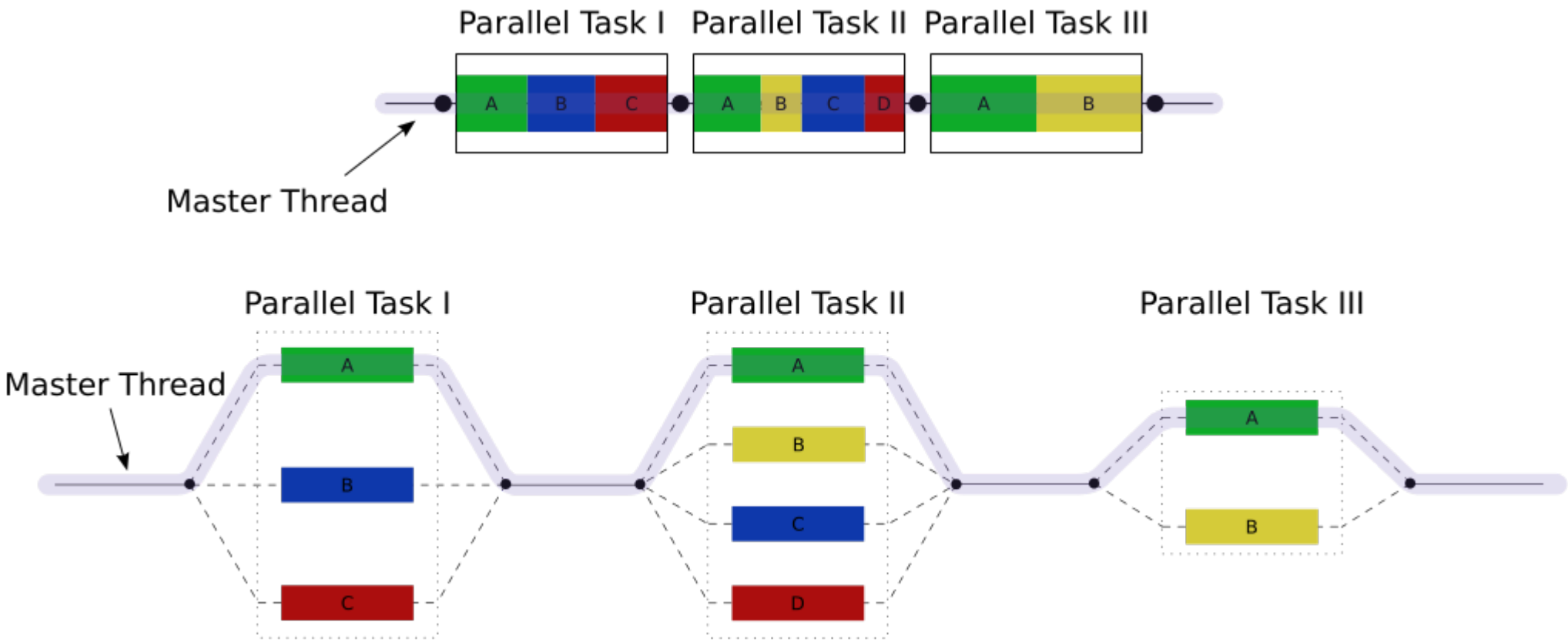


Figure: wikipedia

OpenMP Compiler Flags

- GNU (gcc, g++, gfortran)
  - **-fopenmp**
- Intel compiler (icc, icpc, ifort)
  - **-openmp** or **-qopenmp**
- Clang (clang, clang++)
  - **-fopenmp**
- Microsoft (cl.exe)
  - ***/openmp***

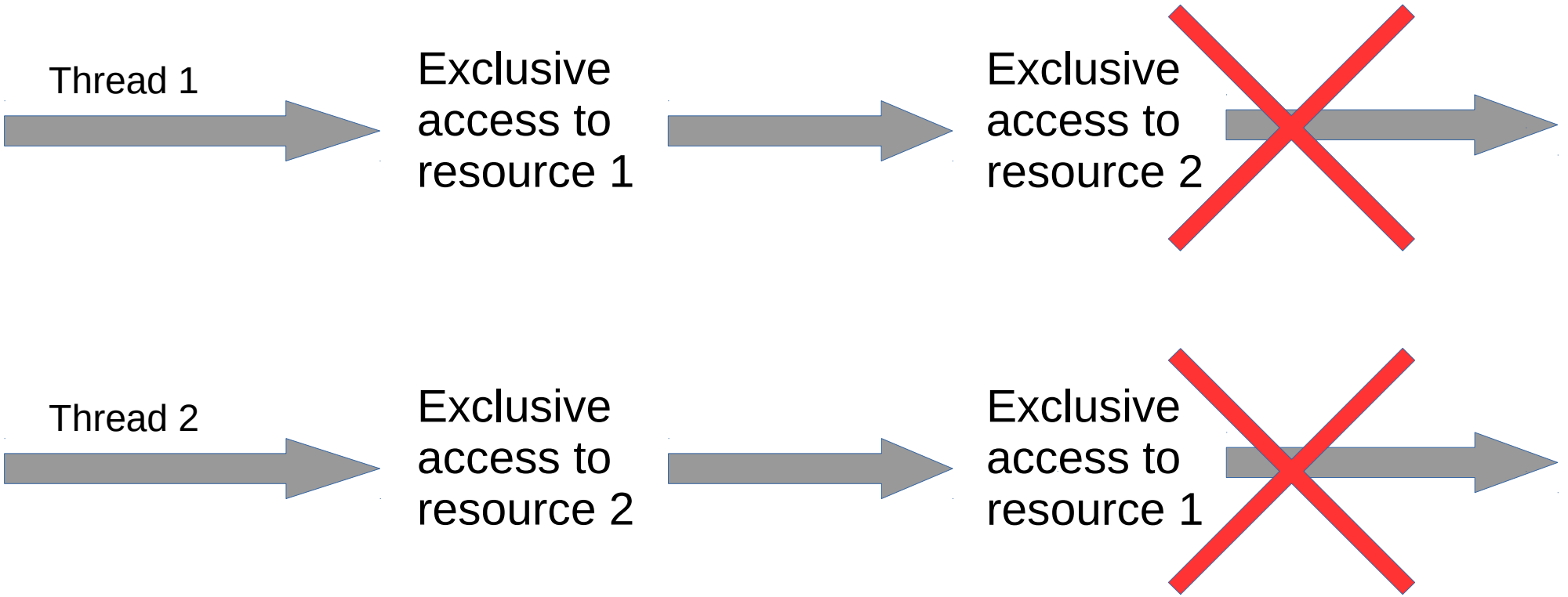
Data race

- We have in our code:
  - x = x + 1
- In reality:
  - Load x from memory to CPU register
  - Add 1
  - Store x to memory

Data race

- We have in our code:
  - x = x + 1
- In reality:
  - Load x from memory to CPU register
  - Add 1 ← **another thread may access memory here**
  - Store x to memory

Dead lock


Thread 1
Exclusive access to resource 1
Exclusive access to resource 2
Thread 2
Exclusive access to resource 2
Exclusive access to resource 1

GitHub repository with examples

git clone https://github.com/ilyak/openmp-tutorial