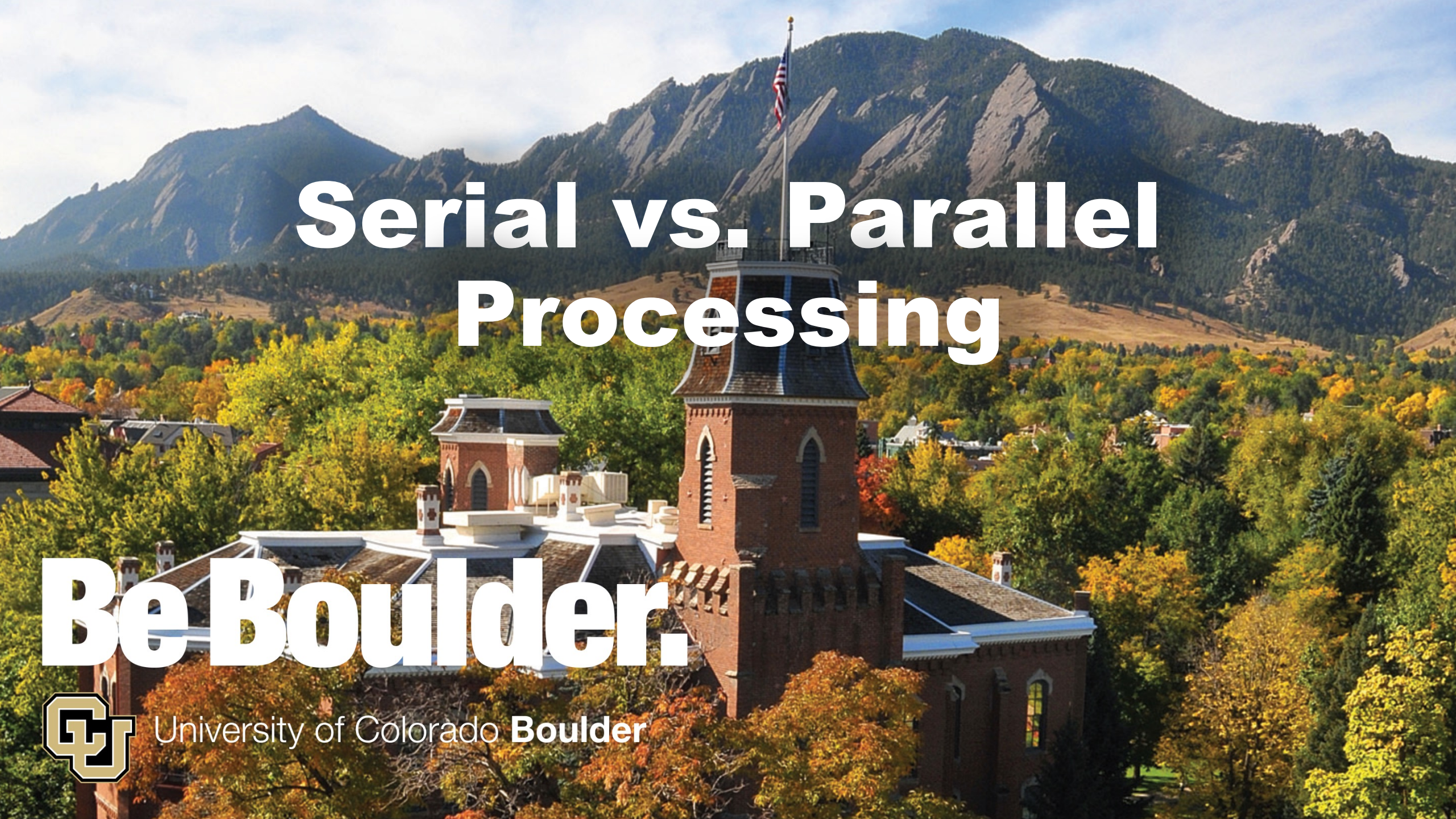

Serial vs. Parallel Processing
Be Boulder.
University of Colorado Boulder

Serial vs. Parallel Processing

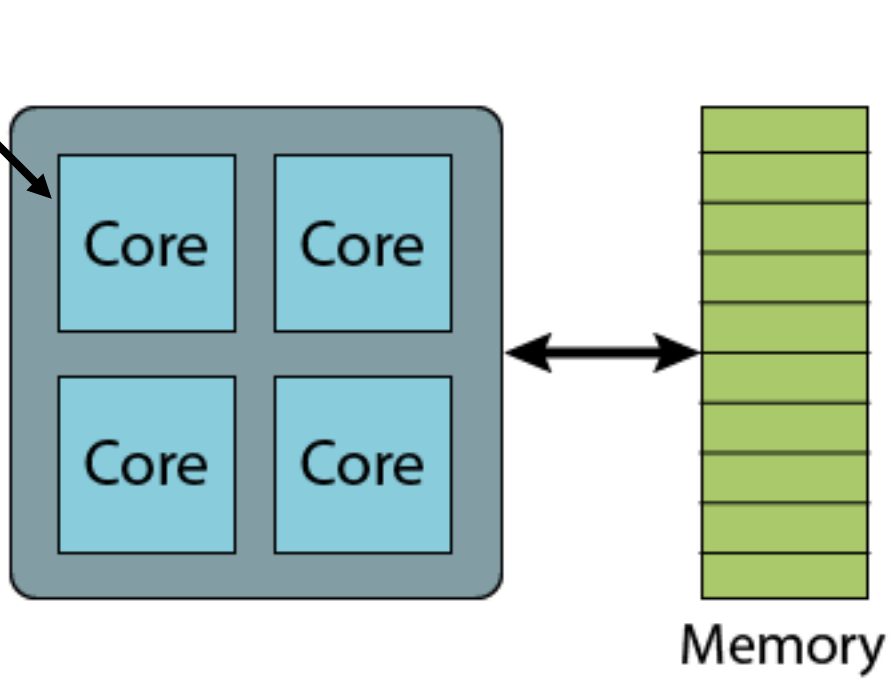
- Serial processing
  - A problem is broken into a set of discrete instructions
  - These instructions are carried out sequentially on a single processor
  - Only one instruction is executed at a tifme
- Parallel processing
  - Idea where many instructions are carried out simultaneously across a computing system
  - Can divide a large problem up into many smaller problems

Why Parallelize?

- Single core too slow for solving the problem in a “reasonable” time
  - “Reasonable” time: overnight, over lunch, duration of a PhD thesis
- Memory requirements
  - Larger problem
  - More physics
  - More particles
  - Larger images
  - Larger neural networks

Basic Computer Architecture

- Old computers – one unit to execute instructions
- New computers have 4 or more cpu cores

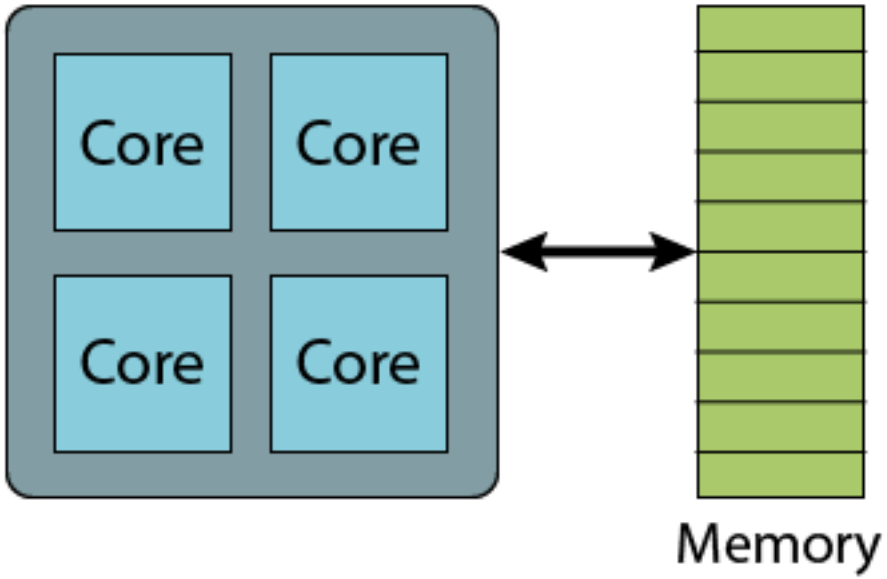


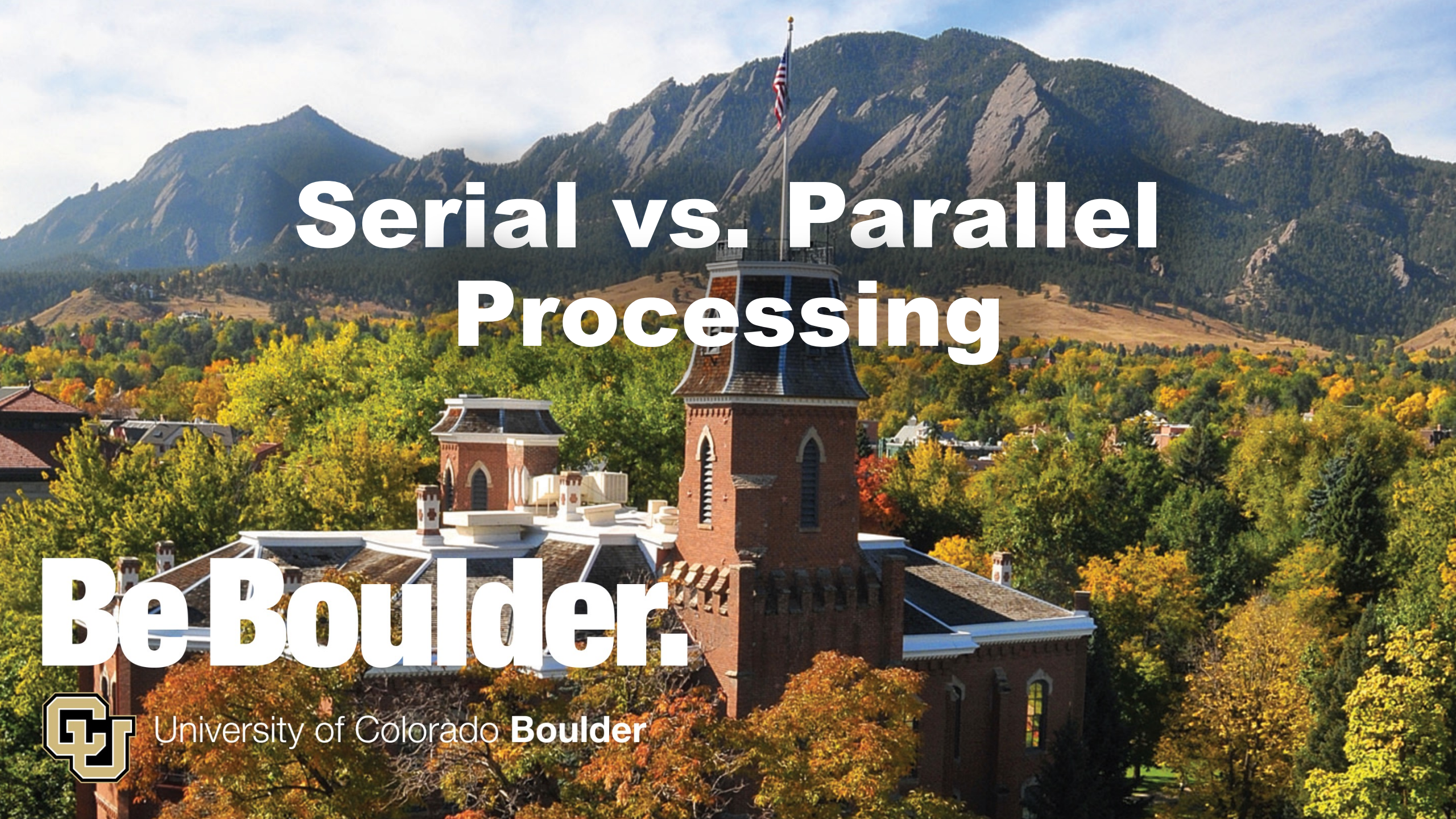
Serial Processing – Thought Experiment

- Let's say you own a lawn service company
- You have one hundred clients who each want their lawn mowed, with patterns
- Each of them want their lawn mowed by the end of the week
- A serial process would be for you to mow all one hundred laws yourself
- You cannot mow lawn 2 until you mow lawn 1, etc
- Let's say doing this takes you the full 7 days to complete, working 16 hour days

Serial Processing

- Instructions are executed on one core
- The other cores sit idle
- If a task is running, Task 2 waits for Task 1 to complete, etc.
- Wasting resources
- Want to instead parallelize and use all cores




Serial vs. Parallel Processing
Be Boulder.
University of Colorado Boulder

Parallel Processing – Thought Experiment

- Let's say that you decide that 100 lawns is too many for one person to mow in a week
  - Or you want to finish it faster
- Therefore you hire one additional person to help you
- How long (in theory) should it take you to finish the lawns?
  - Either 3.5 days working 16 hours each day, or 7 days working 8 hour days
- You could accomplish this either by both working on one lawn together or each of you working on a different lawn at the same time (more on this later)

Parallel Processing – Thought Experiment

- Similarly, you could hire three more people
  - Now five total
- How long should it take you to finish?
  - In theory, five times faster
- However, it doesn't actually work out this way. Why?
  - Overhead
    - Communication
      - Who is mowing which lawn?
      - If you split a lawn, who mows which parts?
      - How do you make sure the patterns match up?

Parallel Processing – Thought Experiment (Cont.)

- However, it doesn't actually work out this way. Why?
  - Resource contention
    - Fights over who gets to use the best lawn mower
- So maybe instead of five times as fast its four times as fast
  - Still faster
- More people?
  - Too many people slows down the process too much to make it worthwhile
    - Diminishing return
    - 100 might be too many

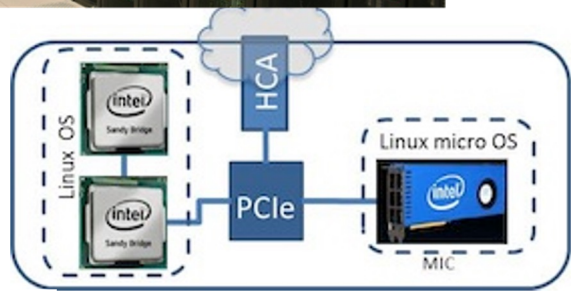
Parallel Overhead

- Should you convert your serial code to parallel?
- Usually do it to speed up
- But need to consider things like overhead
- Overhead because of
  - Startup time
  - Synchronizations
  - Communication
  - Overhead by libraries, compilers
  - Termination time

https://computing.llnl.gov/tutorials/parallel_comp/#ModelsShared

Programming to Use Parallelism

- Parallelism across processors/threads - OpenMP



- Parallelism across multiple nodes - MPI

www.scan.co.uk

Parallel Processing Musts and Tricks

- Need to be able to break the problem up into parts that can work independently of each other
  - Can’t have the results from one CPU depend on another at each time step

- Do loops are a great place to start looking for bottlenecks in your code