RC Quick Byte: Maximizing Efficiency Using Parallelization

Research Computing
UNIVERSITY OF COLORADO BOULDER

Maximizing Efficiency Using Parallelization

Date: April 18, 2024

Instructor: Andrew Monaghan

Contributors: Layla Freeborn, Trevor Hall, Brandon Reyes, Shelley Knuth

- Website: www.rc.colorado.edu/rc
- Documentation: https://curc.readthedocs.io
- Helpdesk: rc-help@colorado.edu
- Survey: http://tinyurl.com/curc-survey18

Slides
https://github.com/ResearchComputing/max_efficiency_parallel_quick_byte

Learning Objectives and Outline

- What is parallelization?
- Types of parallelization
- Is parallelization for me?

What is parallelization?

Serial

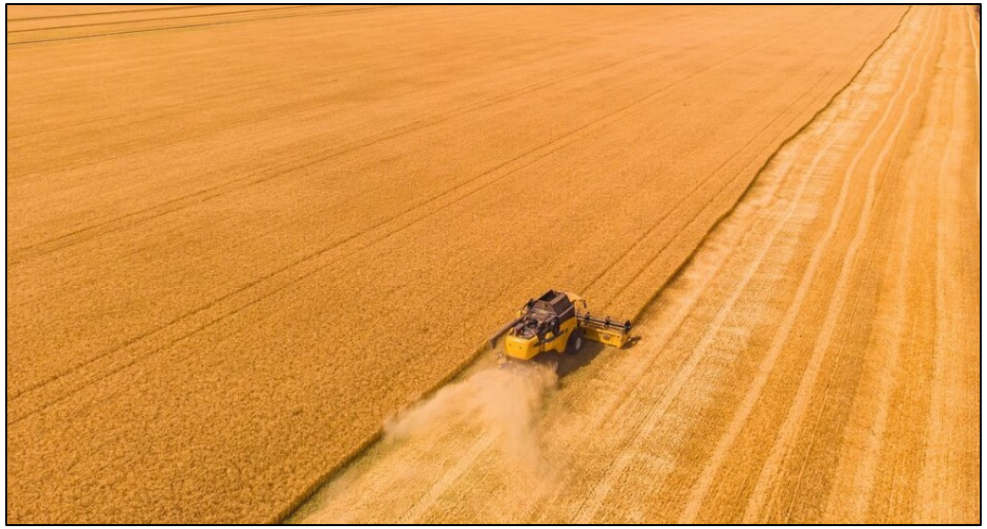


Image source: https://www.freepic.com

Parallel

Image source: https://bxjmag.com

How is parallelization achieved?

- A typical computer ("node") has 1 or more central processing units (CPUs)
- Each CPU has one or more *cores*
- Each core can execute one instruction (task) at a time
- Examples:
  - Your laptop: 4-8 cores
  - A supercomputer node: 32-128 cores
  - A GPU: thousands of (tiny) cores
- A supercomputer like CURC's "Alpine" has 100s to 1000s of nodes that can be used for parallel processing

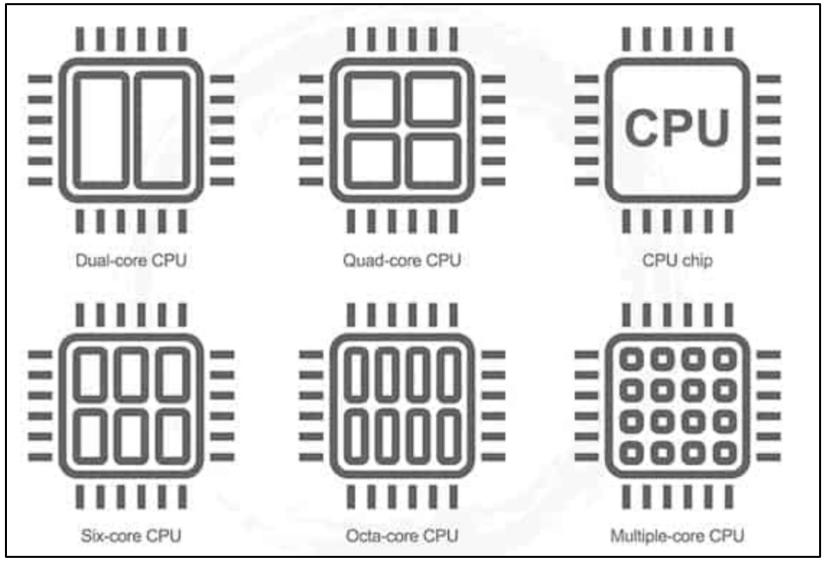
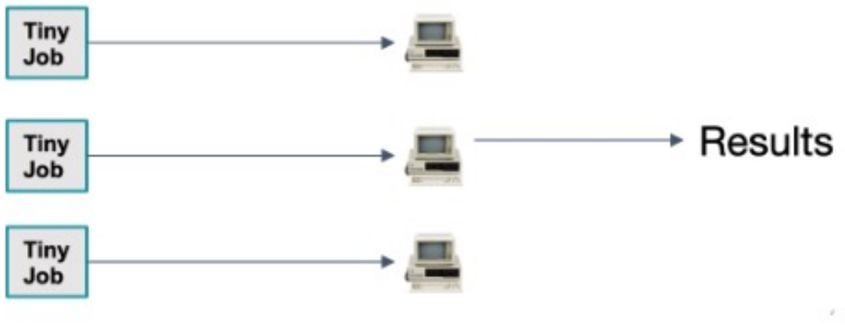


Image source: https://spacehop.com/wp-content/uploads/2021/01/cores.jpg

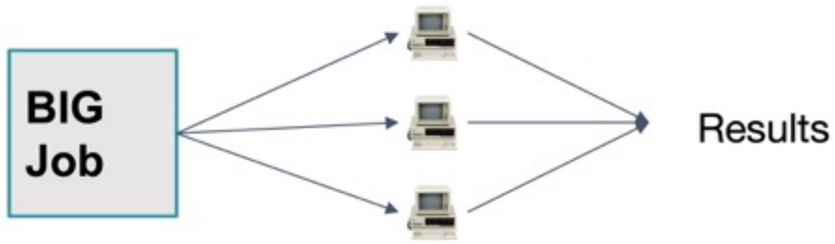
Types of parallelization

Code-**external** parallelization



Example: Image processing

Code-**internal** parallelization



Example: Climate Model

Code-external parallelization

- Also referred to as:
  - HTC: High throughput computing
  - “Embarrassingly” parallel computing
- Used for repetitive, independent tasks
  - Processing images from satellites, microscopes
  - Monte Carlo-type statistical modeling
- CURC has lots of tools to facilitate HTC!
  - https://github.com/ResearchComputing/easy_parallelization_htc_primer

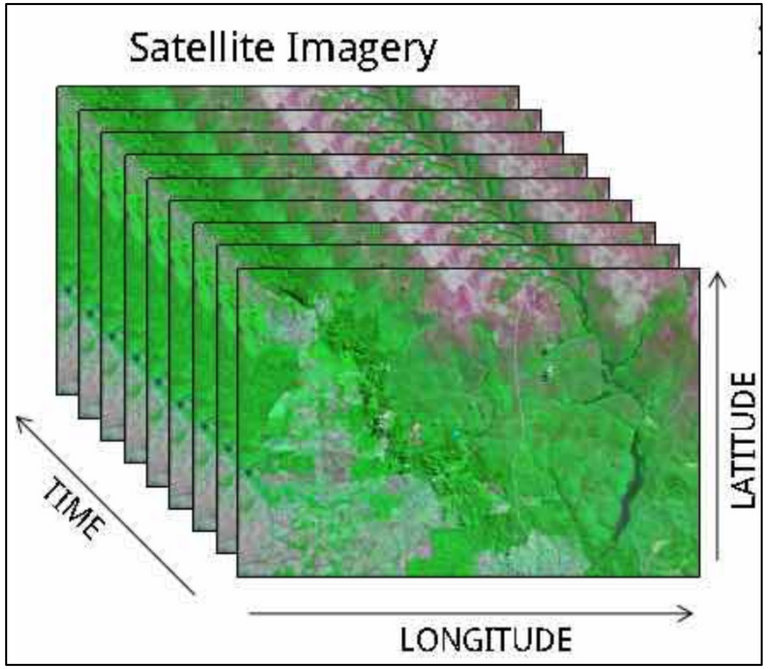


Image source: https://cran.r-project.org/web/packages/Rwtss/vignettes/Rwtss.html

Code-internal parallelization

- Types:
  1. Shared-memory ("multithreading")– single *node* (computer)
  2. Distributed-memory ("multiprocessing or "MPI") – multiple nodes
  3. Accelerated -- GPUs

- Used for dependent, independent tasks
  - Climate or earthquake simulations (PDEs)
  - Machine learning with GPUs (matrices)

- CURC supports all types of code-internal parallelization!

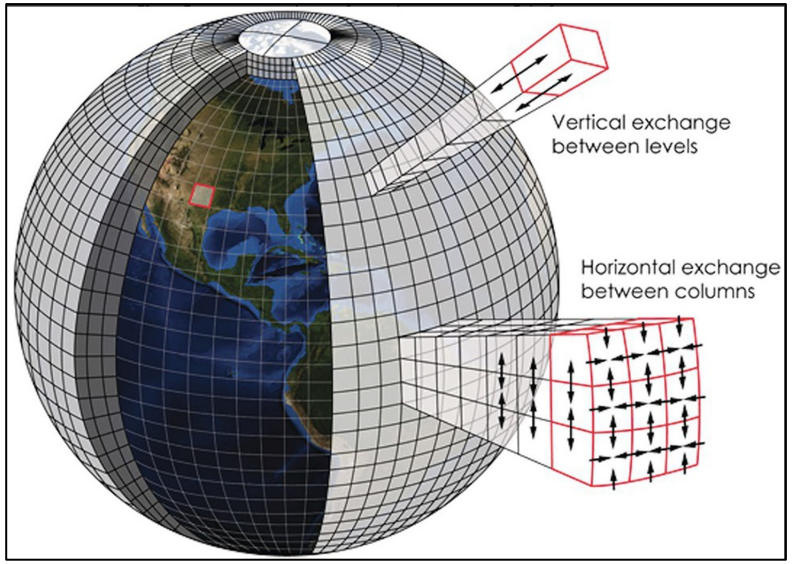

Vertical exchange between levels
Horizontal exchange between columns

Image source: www.earthmagazine.org/article/todays-weather-forecast-good-strong-chance-improvement

Information source: *https://researchcomputing.princeton.edu/support/knowledge-base/parallel-code*

Additional topics (if time allows)

- Is parallelization for me?
    - Computational time constraints?
    - Computational memory constraints?
    - Level of effort to parallelize?

- How do I get started with parallelization?
    - Look for existing code!
    - Consult with your Research Computing staff

Thank you!

Survey and feedback

http://tinyurl.com/curc-survey18