

POIR 613: Computational Social Science

Pablo Barberá

University of Southern California

`pablobarbera.com`

Course website:

pablobarbera.com/POIR613/

Parallel computing

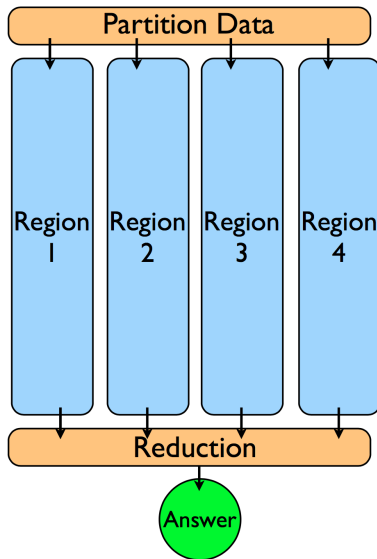
Some hardware terms:

- ▶ **Node**: a single motherboard, with possibly multiple processors
- ▶ **Processor**: silicon containing one or more cores
- ▶ **Core**: unit of computation
- ▶ Most modern CPUs (processors) have multiple cores

Logic of parallel computing

Split-apply-combine framework
(Hadley Wickham and others):

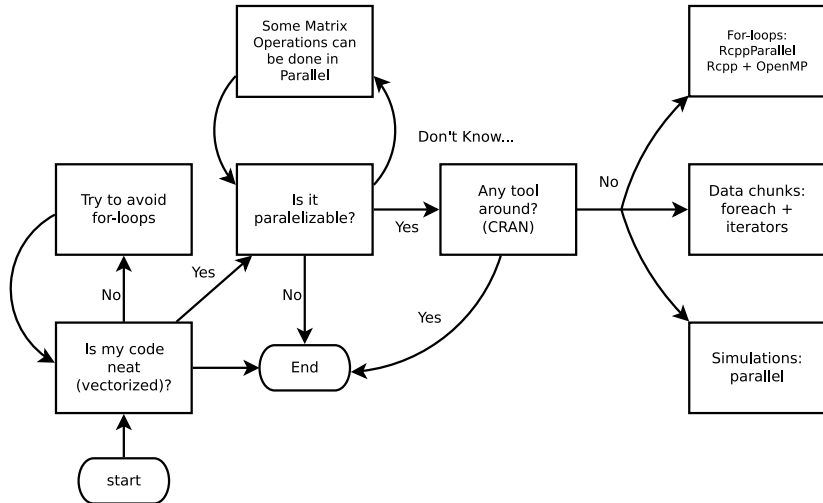
- ▶ **Split** your code and data across multiple nodes/processors/cores
- ▶ **Apply** computation in each region
- ▶ **Combine** the individual results into an aggregate answer



Logic of parallel computing

- ▶ BUT: **overhead** (e.g. splitting and combining data also take some time, no free lunch!)
- ▶ Works best with **embarrassingly parallel** problems:
 - ▶ Statistical simulation using multiple seeds
 - ▶ Word counts in documents
 - ▶ Cross-validation or ensemble learning
 - ▶ **Rule-of-thumb**: can you change the order of the iterations without altering the result?
- ▶ Sometimes problematic: applying on subsets of data, or when full dataset is needed in each node
- ▶ **Not parallelizable**: Markov-Chain Monte-Carlo methods, cumulative sums, etc.

Parallel computing



Source: Vega Yon and Garrett Weaver, 2017

Parallel computing in R

Two main approaches:

1. R packages

- ▶ `parallel`: built-in package with support for parallel computation, including random-number generation (good for statistical simulation)
- ▶ `foreach`: new type of loops that supports parallel execution (good for data analysis)
- ▶ `iterators`: tools for iterating over various R data structures (more advanced)

2. Running C++ code in R:

- ▶ `RcppArmadillo`: interact with C++ linear algebra library
- ▶ `OpenMP`: utility to improve multiprocessing using shared memory; works across all platforms

And **many others** (e.g. Spark, Hadoop, RcppParallel...) we will not cover in this course. See the [High-Performance and Parallel Computing Task View](#)

For more: see [slides+code](#) by Vega Yon and Garrett Weaver