Programming with Big Data in R

Drew Schmidt and George Ostrouchov

July 8, 2013

Affiliations and Support

The pbdR Core Team
http://r-pbd.org

Wei-Chen Chen[1], George Ostrouchov[1,2], Pragneshkumar Patel[2], Drew Schmidt[1]

Ostrouchov, Patel, and Schmidt were supported in part by the project "NICS Remote Data Analysis and Visualization Center" funded by the Office of Cyberinfrastructure of the U.S. National Science Foundation under Award No. ARRA-NSF-OCI-0906324 for NICS-RDAV center.

Chen and Ostrouchov were supported in part by the project "Visual Data Exploration and Analysis of Ultra-large Climate Data" funded by U.S. DOE Office of Science under Contract No. DE-AC05-00OR22725.

[1] Computer Science and Mathematics Division, Oak Ridge National Laboratory, Oak Ridge, TN
[2] Remote Data Analysis and Visualization Center, University of Tennessee, Knoxville, TN

About This Presentation

Downloads

This presentation and supplemental materials are available at:

http://r-pbd.org/user2013

About This Presentation

Speaking Serial R with a Parallel Accent

The content of this presentation is based in part on the **pbdDEMO** vignette *Speaking Serial R with a Parallel Accent*

https://github.com/wrathematics/pbdDEMO/blob/master/inst/doc/pbdDEMO-guide.pdf?raw=true

It contains more examples, and sometimes added detail.

About This Presentation

Installation Instructions

Installation instructions for setting up a pbdR environment are available:

http://r-pbd.org/install.html

This includes instructions for installing R, MPI, and pbdR.

About This Presentation

Conventions

We use:

- "." as a decimal mark
- "," as order of magnitude separator

Example	Yes	No
One million	1,000,000	1.000.000
One half	0.5	0,5
One thousand and one half	1,000.5	1.000,5

Contents

Contents

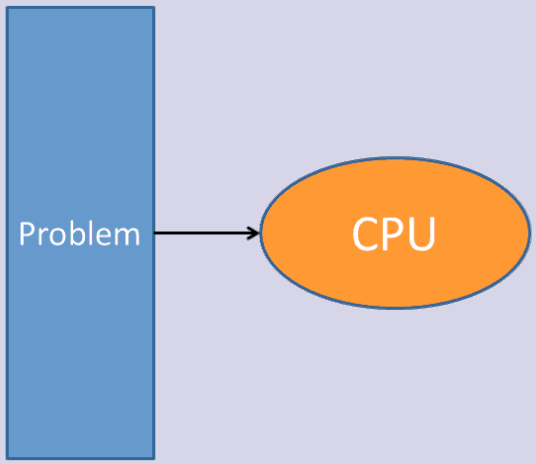
What is Parallelism?

Broadly, *doing more than one thing at a time*.

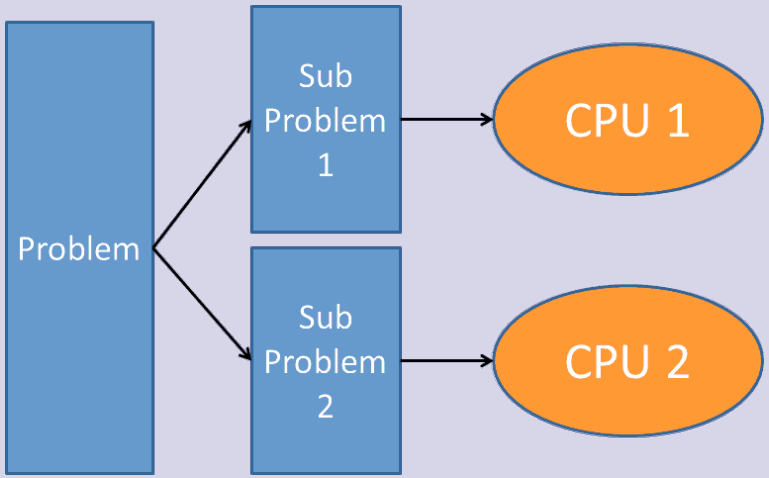
The simultaneous use of multiple compute resources to solve a computational problem:

Parallelism

Serial Programming

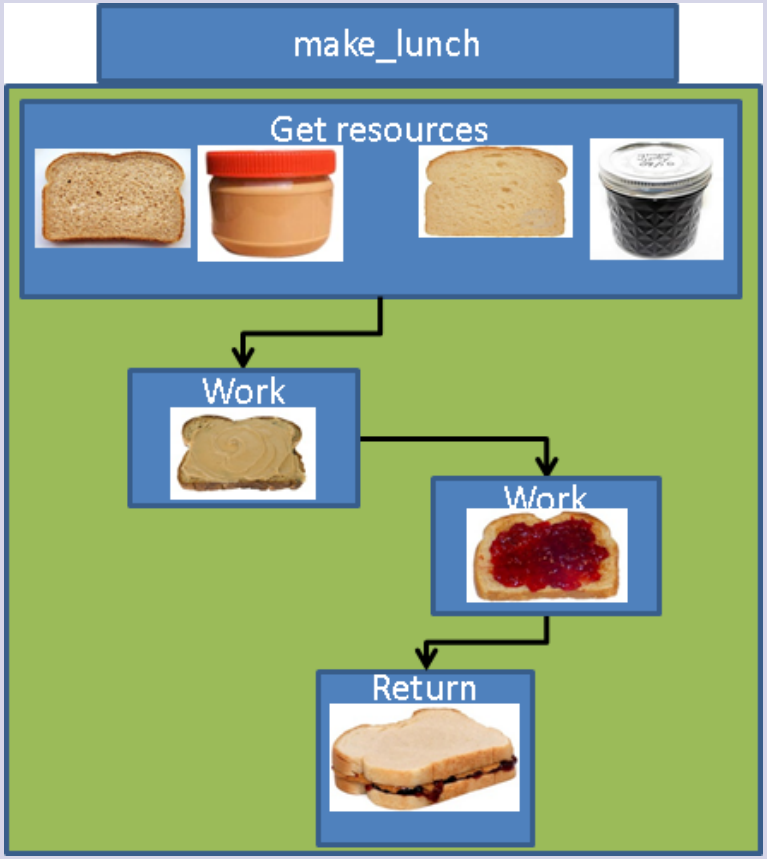


Parallel Programming

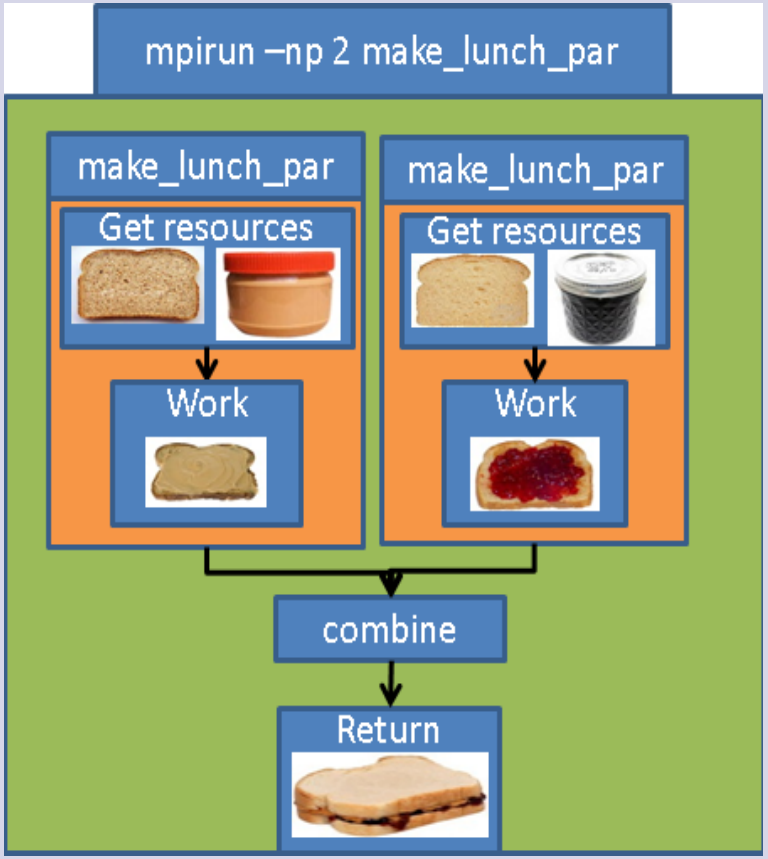


Parallelism

Serial Programming



Parallel Programming



Kinds of Parallelism

- *Data Parallelism*: Data is distributed
- *Task Parallelism*: Tasks are distributed

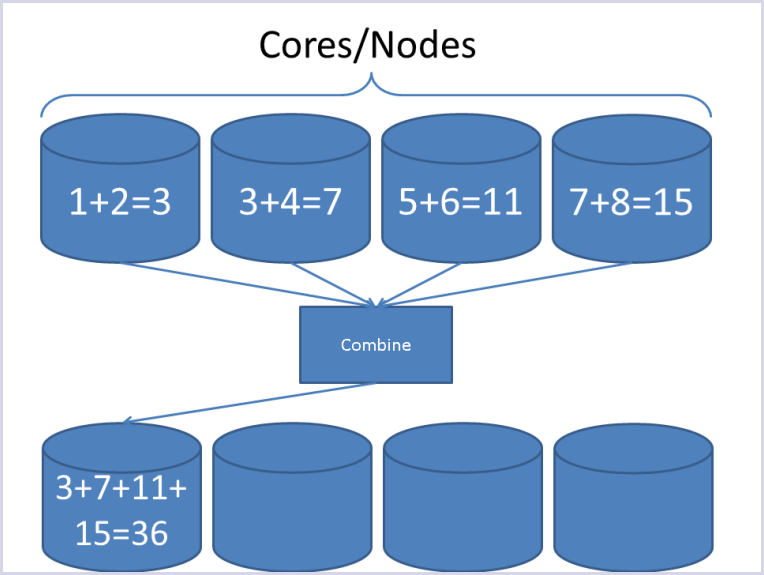
pbdR Paradigms: Data Parallelism

With data parallelism:

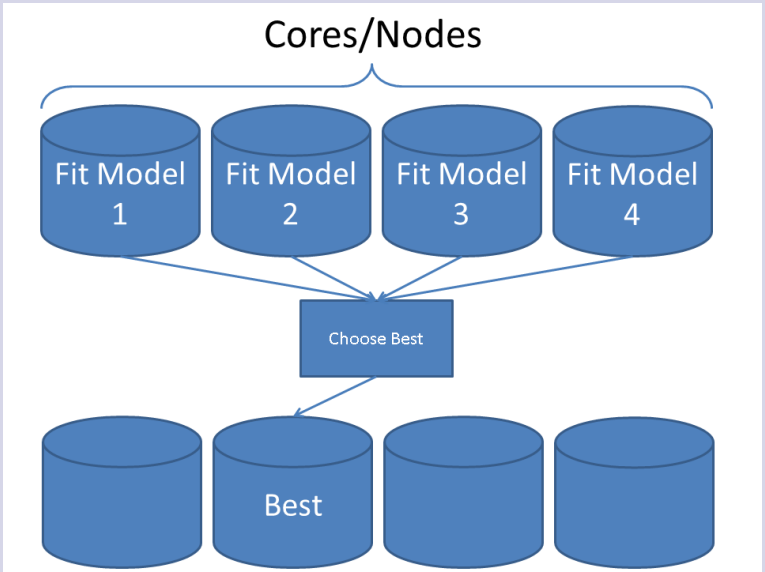
- No one processor/node owns all the data.
- Processors own local pieces of a (conceptually) global object

Data vs Task Parallelism

Data Parallelism



Task Parallelism



Difficulty

1. *Implicit parallelism*: Parallel details hidden from user
2. *Explicit parallelism*: Some assembly required...
3. *Embarrassingly Parallel*: Also called *loosely coupled*. Obvious how to make parallel; lots of independence in computations.
4. *Tightly Coupled*: Opposite of embarrassingly parallel; lots of dependence in computations.

Scalability

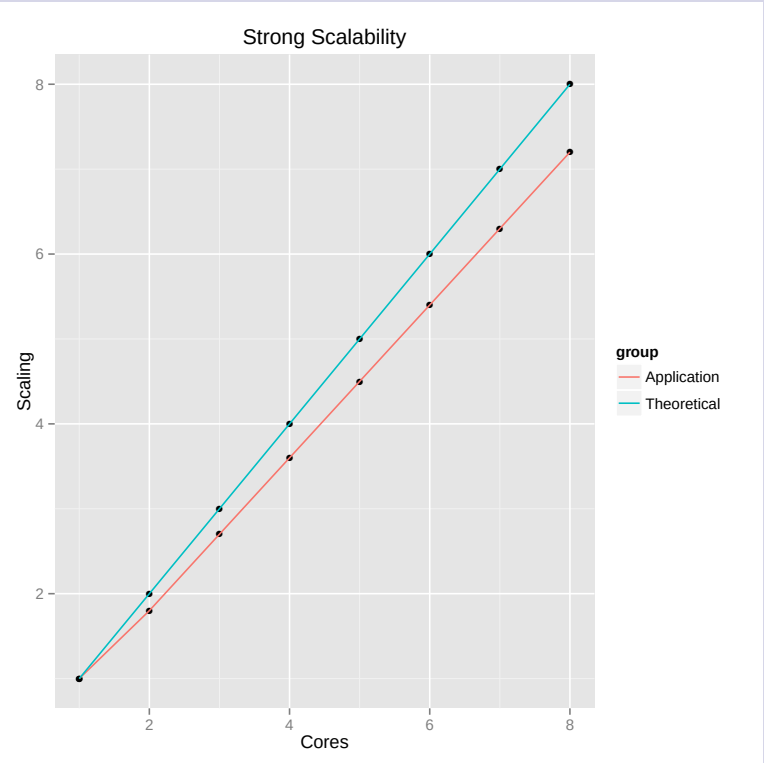
Scalability: unitless measure of performance;

$$\frac{\tau_i}{\tau_0}$$

Types of Scalability: Strong and Weak

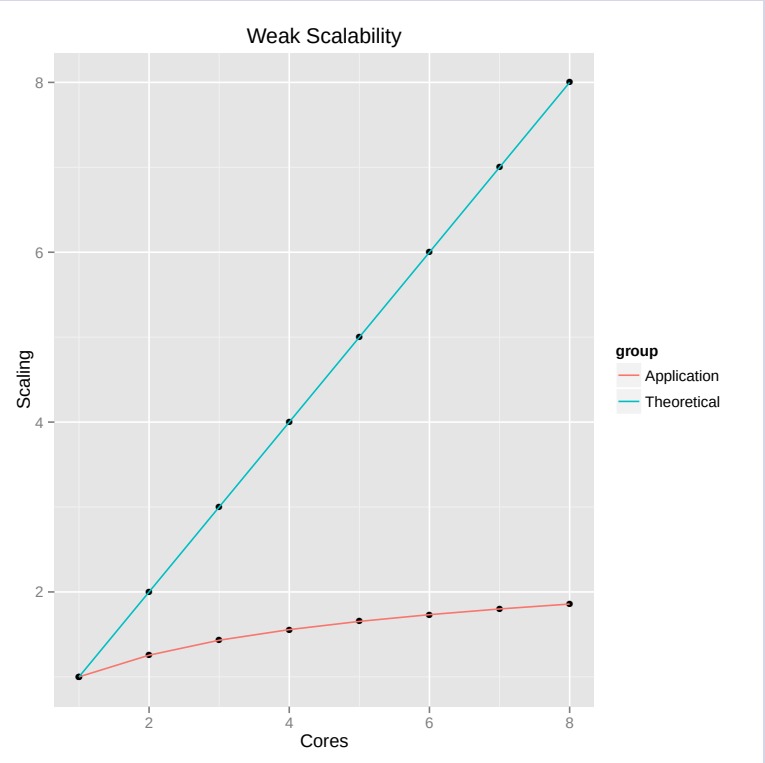
Strong

Fix *total* data size



Weak

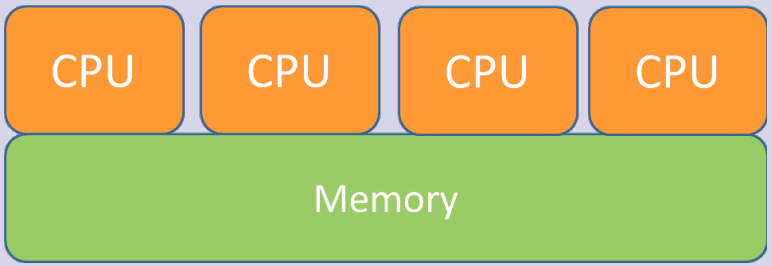
Fix *local* data size



Shared and Distributed Memory Machines

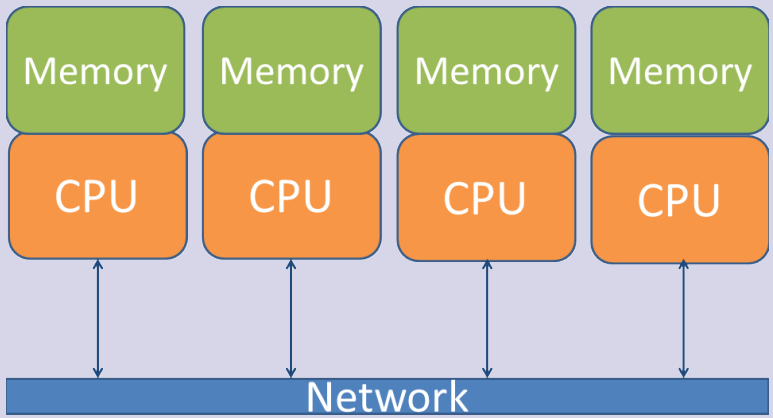
Shared Memory

Different processors can directly access and modify each others' memory. There is only one node.



Distributed

Different processors/nodes can not directly access/modify different processors'/nodes' memory.



Shared and Distributed Memory Machines

Shared Memory Machines

Thousands of cores

Nautilus, University of Tennessee

1024 cores

Distributed Memory Machines

Hundreds of thousands of cores

Kraken, University of Tennessee

112,896 cores

R and Parallelism

What about R?

Problems with Serial R

1. Slow.
2. If you don't know what you're doing, it's *really* slow.
3. Performance improvements usually for small machines.
4. Very ram intensive.
5. Chokes on big data.

Parallel R Packages

Shared Memory

1. **foreach**
2. **parallel**
3. **snow**
4. **multicore**

Distributed

1. **Rmpi**
2. R+Hadoop
3. **pbdR**

R and Parallelism

The solution to many of R's problems is parallelism. However . . .

What we have

1. Mostly serial.
2. Mostly not distributed
3. Data parallelism mostly explicit

What we want

1. Mostly parallel.
2. Mostly distributed.
3. Mostly implicit.

Why We Need Parallelism

1. Saves time (long term).
2. Data size is skyrocketing.
3. Necessary for many problems.
4. Like it or not, it's coming.
5. *It's really cool.*