Making R faster

2014-10-07

Today:

- Sleeping beauty
- Lab3 introduction
  - You'll want parallelization and Rcpp
- Local parallelization with foreach
  - Using the SCF cluster
- C++ integration with Rcpp
- Lab2 reading

Sleeping Beauty

Lab 3

Clustering stability of k-means.

Algorithm 1 Calculation of clustering similarities in k-means

for $k = 2$ **to** k_{max} **do**
  for $i = 1$ **to** n **do**
    $\text{sub}_1 = \text{subsample}\,(X, m)$, a subsample of fraction m of dataset X
    $\text{sub}_2 = \text{subsample}\,(X, m)$, a subsample of fraction m of dataset X
    $L_1 = \text{cluster}\,(\text{sub}_1)$
    $L_2 = \text{cluster}\,(\text{sub}_2)$
    $\text{intersect} = \text{sub}_1 \cap \text{sub}_2$
    $S\,(i, k) = \text{similarity}\,(L_1\,(\text{intersect})\,, L_2\,(\text{intersect}))$
  end for
end for

Local parallelization

Parallelization has a few different flavors:

- Multicore processors
- GPUs
- Computer clusters

This is as far as we'll go in this class

Resources

Chris Paciorek (of STAT 243) is a local expert. The material today is mostly his.

Upcoming seminars:

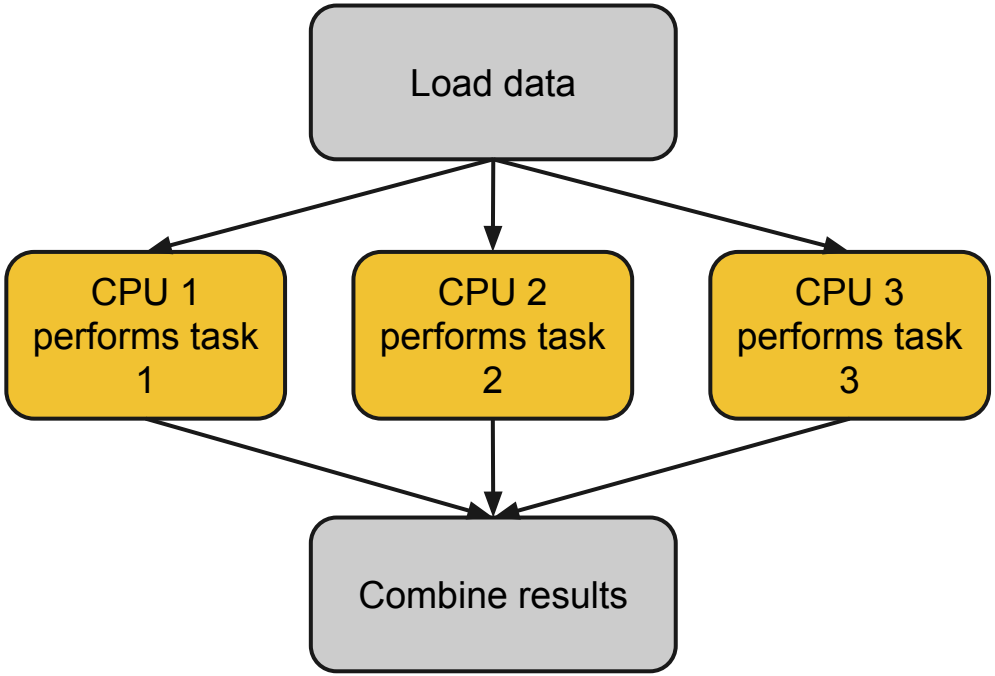
Session 1: Monday October 13, 4:10 - 5:15 pm

Use of the department cluster, basic implementation of embarrassingly parallel calculations in R, Python, Matlab and C

Session 2: Monday October 20, 4:10 - 5:15 pm

Parallel random number generation, more advanced parallelization techniques in R and C (beyond parallelized for loops), use of openMP in C and of MPI in R and C

Local Parallelization



The parallel tasks cannot talk to one another.

You can parallelize to either speed up computation or split up a large data set.

How would you parallelize:

- Monte Carlo integration?
- A bootstrap?
- OLS regression?
- K-means?

R Example

foreach_example.R

Using the SCF cluster

Read section 2 of Chris's document. Long story short:

- Set up a shell script that runs your job (e.g. shell_example.sh).
- Choose a computer and ssh to it.
- Copy your files to that computer
  - Clone a git repository there or
  - Use scp
- Submit your job using something like:
  - `qsub -pe smp 4 shell_example.sh`

This is how many cores you need

This will run your analysis and produce output

Anatomy of shell script

```
#!/bin/bash
R --no-save < shell_example.R
```

This basically just runs commands as if you had typed them. Make sure it's executable:

```
chmod 755 shell_example.sh
```

Rcpp

R is inefficient with memory and for loops. C++ is better because you have more control.

Rcpp allows you to easily integrate C++ code into R.

Demo:

`Rcpp_demo.R`

Exercises:

- Use `foreach` to parallelize k-means with random different starting points.
- Use `Rcpp` to make the binary matrix from lab2 for a particular question.
- Run something on the SCF.