Parallel computing in R

embarrassingly parallel execution

Michael Helbraun

Microsoft

Agenda

- Why parallelize?
- Terms
- Options
- Environments

Agenda

- Why parallelize?
- Terms
- Options
- Environments

Why parallelize?

Open source R is generally in memory and single threaded, but parallelizing allows us:

- To better leverage multicore hardware
- To evaluate larger solution spaces
- To open up bigger problem scenarios
- To execute faster

Note: There is a cost to parallelization – for small problems it may not be faster.

Agenda

- Why parallelize?
- Terms
- Options
- Environments

Terms

- HPA vs HPC
- Explicit vs Implicit parallelization

Agenda

- Why parallelize?
- Terms
- Options
- Environments

Options

For embarrassingly parallel (HPC) problems we typically see foreach or rmr2 used

For HPA problems there are many additional packages

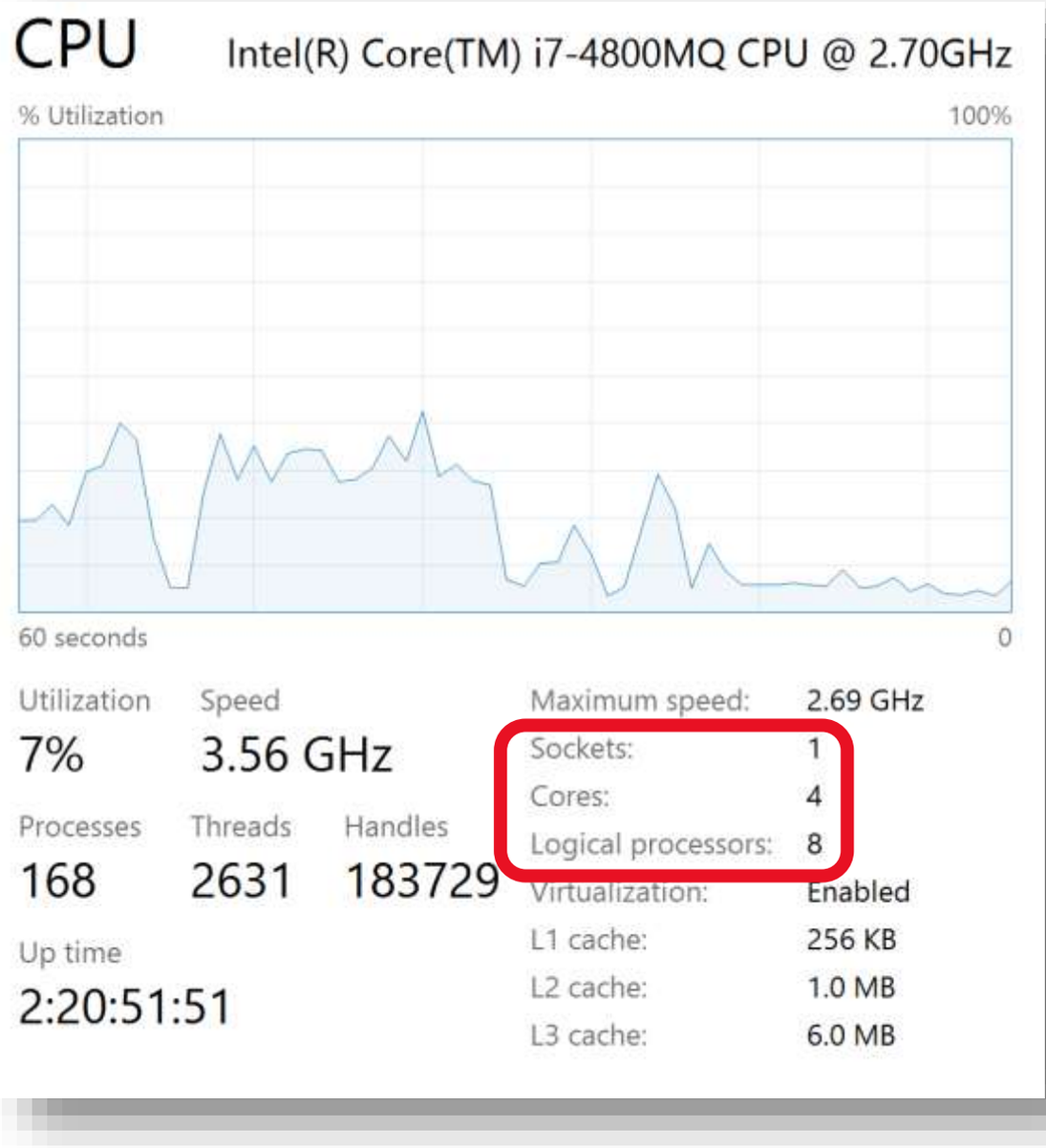
- Big R, Distributed R, OracleR, H2O, sparkR or sparklyr (+ MLlib), RHadoop, MRS

Agenda

- Why parallelize?
- Terms
- Options
- Environments

Environments

- Almost any modern laptop will have multiple cores
- Cloud.
  - Azure instances have up to 32 cores per machine
  - R images already exist or you can build your own VM



Demo – foreach and rxExec

	foreach	rxExec
License	Open source Apace license	Commercial license
Parallel backends	doFuture, doMC, doMPI, doParallel, doRedis, doRNG, doSNOW	All foreach *plus* SQL Server, Teradata, Hadoop MR, and Hadoop Spark
Larger than memory support	Same as OSR: manually managed or specialty packages (e.g., ff)	Support for blockwise operations on Rx* data objects
Random Number support	doRNG (L'Ecuyer-CMRG)	doRNG and natively in rxExec ("MCG31", "R250", "MRG32K3A", "MCG59", "MT19937", "MT2203", "SFMT19937", "SOBOL", "L'Ecuyer-CMRG", and "auto"
Performance		
Package support	There are R packages that depend on foreach and it's available parallel backends	These open source packages would like need rework to leverage the ScaleR backends

Demo:
foreach and rxExec