A Study on CRAN R and MRAN R Interpreters

Devharsh Trivedi
M. Tech. Student
Department of Computer Science & Engineering (Information and Network Security)
Institute of Technology, Nirma University, Ahmedabad, India

Abstract— R is popular software environment for statistical computing and graphics. It is widely used by data analysts and data miners and it is also used for testing machine learning algorithms. But how efficient is this interpreter? Can the performance be improved? Can we reproduce results? This paper aims to find out these answers.

Key words: R, Microsoft R, Multithreading

I. Introduction

R is a free programming language and software environment for statistical computing and graphics. The R language is widely used among statisticians and data miners for developing statistical software and data analysis. [1]

CRAN R refers to original R interpreter developed by R Core Team and MRAN R refers to extended R developed on original R by Microsoft.

II. History of R

R was created by Ross Ihaka and Robert Gentleman in 1993 at the University of Auckland, New Zealand, and is currently developed by the R Development Core Team, of which Chambers is a member. R is named partly after the first names of the first two R authors and partly as a play on the name of S. [2]

R is a language and environment for statistical computing and graphics. It is a GNU project which is similar to the S language and environment which was developed at Bell Laboratories (formerly AT&T, now Lucent Technologies) by John Chambers and colleagues. R can be considered as a different implementation of S. There are some important differences, but much code written for S runs unaltered under R.

R provides a wide variety of statistical (linear and nonlinear modelling, classical statistical tests, time-series analysis, classification, clustering …) and graphical techniques, and is highly extensible. The S language is often the vehicle of choice for research in statistical methodology, and R provides an Open Source route to participation in that activity.

One of R’s strengths is the ease with which well-designed publication-quality plots can be produced, including mathematical symbols and formulae where needed. Great care has been taken over the defaults for the minor design choices in graphics, but the user retains full control.

R is available as Free Software under the terms of the Free Software Foundation’s GNU General Public License in source code form. It compiles and runs on a wide variety of UNIX platforms and similar systems (including FreeBSD and Linux), Windows and MacOS. [3]

III. Variants of R

I could manage to find two major industry grade variants of R interpreter that are some sort of extension to existing R distribution from CRAN. (i) ValidR and (ii) Microsoft R

A. ValidR

ValidR is a product developed by Mango Solutions that is delivering a validated version of this R language into regulated industries. It is designed to transform a language that, as a standard installation, provides “absolutely no warranty”, into a system whose scripts comply with regulatory guidelines on the qualification and validation of systems such as the FDA’s 21 CFR Part 11. [4]

B. Microsoft R

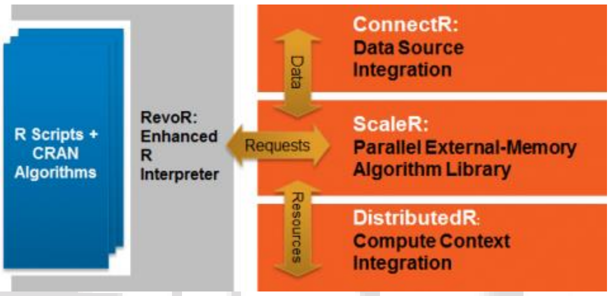
R Scripts + CRAN Algorithms
RevoR: Enhanced R Interpreter
Requests
Data
Resources
ConnectR: Data Source Integration
ScaleR: Parallel External-Memory Algorithm Library
DistributedR: Compute Context Integration

Fig. 1: Microsoft R Architecture

Microsoft R Open, formerly known as Revolution R Open (RRO), is the enhanced distribution of R from Microsoft Corporation. It is a complete open source platform for statistical analysis and data science. It is available for all: windows, linux and mac OS. [5]

C. Advantages of Microsoft R over R

R is memory bound and single threaded hence it is slow. Moreover R does not provide proper package management that means if you want to have reproducible results you may not get it as the package version used might have been changed.

There is a third party package called packrat that is used for dependency management where Microsoft R provides its own checkpoint utility that uses CRAN Time Machine so you can pick a specific date package. [6] [7]

Another major advantage of using Microsoft R is multithreading. It used Intel MKL library to use all available cores to make R processing faster. You can also change the number of cores to be used by setMKLthreads(<value>) argument. We will see how this improves performance in next sections. [8]

Microsoft offers R in two versions: Open and Server. Microsoft R Server compared to Microsoft R Open (MRO) provides additional packages and services for data gathering and processing and it performance of both is similar.

ValidR doesn’t seem to improve R like Microsoft R does but it only offers additional packages so in my performance comparison I have included Microsoft R Open and CRAN R as both of them are open source.

IV. Benchmarks

I have used two R scripts for benchmarking both computation and graphics. Script for benchmarking computation is adopted from R Benchmark 2.5 (06/2008) by Simon Urbanek. And for benchmarking graphics I have created my own script that (i) reads a file, (ii) processes it and (ii) plots it accordingly. [9]

Source code for graphics benchmark is as follow:

```
mybenchmark <- function()
{
 library(plotly)
 mylog <- read.csv("E:\\Vertical Scalability\\Current\\mydata.csv")
 at_list <- unique(mylog$ActivityType)
 for (i in 1:length(at_list))
  assign(paste0("mylog", i), mylog[mylog$ActivityType %in% at_list[i], ])
 for (i in 1:length(at_list))
  assign(paste0("mycount", i), as.data.frame(table(unlist(eval(
    parse(text = paste0("mylog", i, "[,2]"))
  )))))
 p1 <- plot_ly(
  x = mycount1[, 1],
  y = mycount1[, 2],
  type = "bar",
  name = at_list[1]
 )
 for (i in 2:length(at_list))
 {
  string <- paste0(
   "add_trace(p",
   i - 1,
   ",x = mycount1[,1],y = mycount",
   i,
   "[,2],type = 'bar',name = '",
   at_list[i],
   "')"
  )
  assign(paste0("p", i), eval(parse(text = string)))
 }
 layout(
  paste0("p", length(at_list)),
  barmode = "stack",
  xaxis = list(title = '<-- Time -->'),
  yaxis = list(title = '<-- Count -->')
 )
}
system.time(mybenchmark())
```

If you view the plot generated by this plotly plot function it would look like as follow:

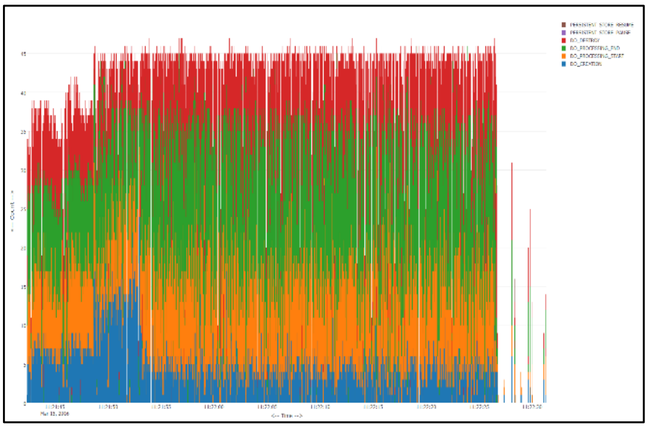

Fig. 2: Plot output

V. Result & conclusion

Distribution	Version	Bits	Plotting Time	Computing Time	Total Time
MRO/RRE8/4T	3.2.2	64	58.96	131.36	190.32
R	3.2.4	64	59.39	232.68	292.07
R	3.2.2	64	60.57	234.39	294.96
R	3.2.3	64	61.66	232.51	294.17
MRO/4T	3.2.3	64	62.61	125.84	188.45
R	3.2.4	32	66.81	242.00	308.81
R	3.2.2	32	67.31	242.77	310.08
R	3.2.3	32	67.60	244.54	312.14

Table 1: Time taken in seconds for various R distros

MRO/RRE8/4T stands for Microsoft R Open 3.2.2 for Revolution R Enterprise 8 using 4 threads and MRO/4T is Microsoft R Open 3.2.3 using 4 cores.

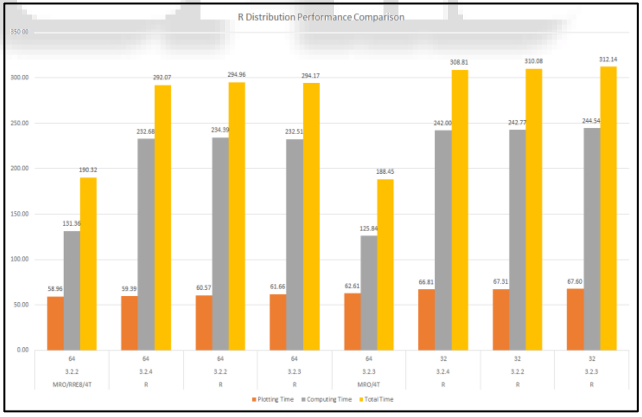

Fig. 3: Performance comparison of R distros

It was observed that all versions took almost same average time for plotting graphs while Microsoft R took half of the time than regular R for computing complex calculations. So it can be concluded that MRAN R is twice as fast as CRAN R.

References

[1] https://en.wikipedia.org/wiki/R_%28programming_language%29
[2] Kurt Hornik. The R FAQ: Why is R named R? ISBN 3-900051-08-9. Retrieved 2008-01-29.
[3] https://www.r-project.org/about.html
[4] http://www.mango-solutions.com/wp/products-services/products/validr/

[5] https://mran.microsoft.com/open/
[6] https://rstudio.github.io/packrat/
[7] https://mran.revolutionanalytics.com/timemachine/
[8] https://software.intel.com/en-us/intel-mkl
[9] http://r.research.att.com/benchmarks/R-benchmark-25.R