Renjin: The new R interpreter built on the JVM

Alexander Bertram
BeDataDriven
Renjin is a new interpreter for the R language.
Why?

- Performance
- Memory
- Speed
- Parallelism
- Easier Integration
- Java Virtual Machine
- GC
- 500k libs
- JIT tools
- 500k
- GC
- 500k
- JIT tools
- 500k
- GC
- 500k
- JIT tools
- 500k
Sure, but why Renjin?

Packages

- bigvis
- biglm
- scaleR

+ High performance for specific applications
- Require rewriting existing code
- Limited applicability

Forks

- scaleR
- pqR

+ Marginal improvements for all code
- Unable to address underlying limitations of the GNU R interpreter
What do I get, like, today?
Flexible

Command-Line Interpreter

> renjin -f myscript.R

Embeddable Java Library
Multiple In-process sessions, Shared Data

Web Request → Renjin Session 1 → Vector
Web Request → Renjin Session 2 → Vector
Web Request → Renjin Session 3 → Vector

Vector is an immutable data structure.
Memory Efficiency

```r
x <- runif(1e8)  # GNU R
                  # +721 MB
y <- x + 1        # +761 MB
comment(y) <- "important!"  # +763 MB
```

- `getAttributes()`
- `Vector Interface`
- `length()`
- `getElement(int index)`
packages.renjin.org

- Pre-built Package Repository
- Proper Dependency Management
- Automated Testing of Renjin
- Translation of C/Fortran to JVM Bytecode
Seamless Access to Java/Scala Classes

```scala
import (com.acme.Customer)

bob <- Customer$new(name='Bob', age=36)
carol <- Customer$new(name='Carole', age=41)

bob$name <- "Bob II"
cat(c("Name: ", bob$name, "; Age: ", bob$age))
```
Simple to embed in larger systems

```java
// create a script engine manager
ScriptEngineManager factory = new ScriptEngineManager();

// create an R engine
ScriptEngine engine = factory.getEngineByName("Renjin");

// load package from classpath
engine.eval("library(survey)");

// evaluate R code from String
engine.eval("print('Hello, World')");

// evaluate R script on disk
engine.eval(new FileReader("myscript.R"));

// evaluate R script from classpath
engine.eval(new InputStreamReader(
    getClass().getResourceAsStream("myScript.R")));
```
@DataParallel
@Deferrable
public static String chartr(
    String oldChars,
    String newChars,
    @Recycle String x)
{
    StringBuilder translation = new StringBuilder(x.length());
    for(int i=0; i!= x.length(); ++i) {
        int codePoint = x.codePointAt(i);
        int charIndex = oldChars.indexOf(codePoint);
        if(charIndex == -1) {
            translation.appendCodePoint(codePoint);
        } else {
            translation.appendCodePoint(newChars.codePointAt(charIndex));
        }
    }
    return translation.toString();
}
Under the hood
Specialized Execution Modes

“Slow” AST Interpreter
- Supports full dynamism of R
- Compute on the language

Vector Pipeliner
- Acts like a query planner
- Batches, auto-parallelizes vector workflows

Scalar Compiler
- Partially evaluates & compiles loop bodies, apply functions to JVM byte code
Queuing up work for the Vector Pipeliner

```r
x <- runif(1e6)
y <- sqrt(x + 1)
z <- mean(y) - mean(x)
attr(z, 'comments') <- 'still not computed'
print(length(z)) # prints "1"
                   # but doesn't
                   # evaluate the mean
print(z)        # triggers computation
```
x <- `runif`(1e6)
y <- `sqrt`(x + 1)
z <- `mean`(y) - `mean`(x)
Real-world case study:
Distance Correlation in the Energy Package
**Distance correlation**: robust measure of association. Zero if and only if variables are independent.
```r
dcor <- function (x, y, index = 1) {
  x <- as.matrix(dist(x))
  y <- as.matrix(dist(y))
  n <- nrow(x)
  m <- nrow(y)
  dims <- c(n, ncol(x), ncol(y))
  Akl <- function(x) {
    d <- as.matrix(x)^index
    m <- rowMeans(d)
    M <- mean(d)
    a <- sweep(d, 1, m)
    b <- sweep(a, 2, m)
    return(b + M)
  }
  A <- Akl(x)
  B <- Akl(y)
  dCov <- sqrt(mean(A * B))
  dVarX <- sqrt(mean(A * A))
  dVarY <- sqrt(mean(B * B))
  V <- sqrt(dVarX * dVarY)
  if (V > 0)
    dCor <- dCov/V
  else dCor <- 0
  return(list(dCov = dCov, dCor = dCor, dVarX = dVarX, dVarY = dVarY))
}
```

- `dist(x)` Evaluates as a view.
- Defer `rowMeans(x)` until later.
- Need to evaluate.
Run time of distance correlation of 10 pairs of variables

GNU R  C  Renjin
Where do we go from here?
Inspired by...
Join us!

Contribute!

Contract us for Commercial Support

Download & Test

Sponsor Development!

> Renjin.org