Building Bridges with Julia
Using OHDSI R Packages in Julia

PRESENTER: Jacob S. Zelko

INTRODUCTION:
The amount of healthcare data is skyrocketing. Waiting on answers from this data just one day, can mean thousands of lives lost – as seen in COVID-19. Beyond the cost of human lives, the financial costs of needed hardware for delivering these crucial answers are rising. Between the issues of financial costs and the urgency for rapid insights, there is clear need in the OHDSI community to encourage tools that can bridge this problem.

METHODS:
Demonstrated here is a possible solution, using a dynamic and high performance programming language, Julia, to interoperate with R to utilize OHDSI packages and perform basic procedures easily to produce performance gains. For this approach, I used the R package JuliaConnectR to call Julia tools from within R and the Julia package RCall to call R tools from within Julia. Two basic exercises for benchmarking were done:

- Task 1: Read SynPUF CSV
- Task 2: Query Eunomia

RESULTS:

![Data chart showing performance comparison between Julia and R]

For Task 1, a 5x speedup is seen from the Julia in R example over R’s CSV reader. For Task 2, only using R and R embedded in Julia was considered. Leveraging Julia, the R in Julia example is 2x faster as the R implementation.

Task 1 Code: Reading SynPUF CSV

```r
# Using Julia's CSV reader using CSV
# Reading in SynPUF data data = CSV.File("synpuf.csv")

# Reading raw SynPUF data data <- read.csv("synpuf.csv")

# Load RCall package using RCall
# Read SynPUF data using R
data = read.csv("synpuf.csv")

# Load the JuliaConnectR package
library(JuliaConnectR)

# Import Julia's CSV reader
csv <- juliaImport("CSV")

# Read in example SynPUF data data <- csv(file="synpuf.csv")
```

Task 2 Code: Query Eunomia

```r
# Open connection to Eunomia
library('DatabaseConnector')
connectionDetails <- Eunomia::getConnectionDetails()
connection <- connect(connectionDetails)

# Create SQL Query
sql <- "SELECT * FROM cdm.person"

# Return people from SQL query
result <- renderTranslateQuerySql(connection, sql, cdm = "main")

# Make R data frame
data.frame(t(sapply(result, c)))
```

Julia in R (Fig 1D), can provide a nearly 5x speedup over the R CSV reader (Fig 1B).

SELECTED DISCUSSION TOPICS:
How were benchmarks made?

In Table 1, the minimum time from 10 evaluations of the code created for each task was recorded. For the “Read SynPUF CSV” task, 150 MB of SynPUF data were read. The benchmarking tools, BenchmarkTools.jl for Julia and bench for R were used to generate times.

Why Julia instead of language X?
- Interoperability with other languages
- High performance computing
- Understandable syntax.
- Emerging resources for OHDSI tasks (e.g. database interfaces, OMOP CDM, etc.)

What are future research directions?
Further directions for this research will be to
- Leverage existing OHDSI tools with Julia
- Identify improvements with Julia via language interoperability (i.e. R & Julia)
- Develop tools for actual study to test how feasible it is to leverage Julia in an OHDSI network study design.

ACKNOWLEDGEMENTS
Thank you so much to the following people for their support in this endeavor!
- Charity Hilton and Jon Duke (Georgia Tech Research Institute)
- Dilum Aluthge and Clark C. Evans (JuliaHealth)
- Kristin Kostka (OHDSI Community)

REFERENCES: