Julia Programming Language

By Hunter Capestany, Saylee Marulkar, Michael Wilson, Zach Monheim, and Jared Hubert

Introduction

- Main creators in 2009:
  - Jeff Bezanson, Stefan Karpinski, Viral B. Shah, and Alan Edelman
- Began to pick up steam in 2012
- NASA and the FAA are hugely responsible for development of Julia
- Julia has features that are similar to many languages you may know:
  - Python
  - Java
  - Matlab
- Provides asynchronous I/O, debugging, logging, a package manager, etc.

Dynamic Typing

- Similar to Python with dynamic inferred types.
- Can increase efficiency by still taking advantage of static type system by allowing indication of certain types.
    - `x::Int8 = 100` (Type declaration)
    - Can confirm program runs as expected.
    - Provides more information to compiler for performance.

Other Language Library Access

- The Julia language can use shared libraries from C and Fortran.
- Can use C and Fortran code in Julia
  - machine instructions generated by Julia's JIT are the same as a native C call would be
- Access to other libraries despite being a newer language

Negative Array Indexing

- Equivalent to Python, Julia supports negative indexing of arrays
- Not available in most other languages i.e. ArrayIndexOutOfBoundsException in Java
- Index into an array from the end instead of the start
- More efficient since calculating the offset using the length of the array is not needed
- More readable: Accessing Second to Last Element for Example
    - array[length(array) - 1]
    - array[-2]
    - Equivalent Statements but the second is more concise and quicker to compute

Garbage Collection

- Mark-Sweep Garbage Collection
  - Mark objects as reachable from set of roots and sweeps unreachable objects
- Does not do any kind of reference counting
- Does not move or copy objects like some techniques that other languages implement
- Makes use of Generational Garbage Collection
  - Focuses on younger and newly created objects
  - Moves surviving objects to older generations
- No garbage collector options
  - Helps ensure collector cannot be changed for the worst

JAOT (Just Ahead of Time) Compilation

- Like Fortran, C, C++
    - Compiles ahead of time only machine code for specific types
- However, dynamically typed
    - Scans instructions and compiles for types with type inference
    - Caches the found types
    - If a different type is found, recompiles and caches separately

Parallel Computing

- Asynchronous “tasks”/coroutines
  - “Tasks”
  - Allows for interrupts and switching between tasks
  - Uses idea of producer/consumer rather than caller/callee via Channels
- Multi-threading
  - Starts with single thread of execution by default
  - Generate multiple threads through use of “--threads” flag
  - Programmer is responsible for avoiding data-races via the use of a locking format
- Distributed computing
  - Provided by the “Distributed” module
  - Allows for the use of multiple CPUs with separate memory domains
  - Uses RemoteChannel for communication using put! and take! commands

Scientific Programming using Julia

- Data Visualization
    - Jupyter Notebook
        - https://github.com/JuliaCloud/JuliaBox
        - iJulia Kernel
- Data mining
- Large - scale linear algebra
- Parallel Computing
- Distributed Computing
- AI development and Deep Learning
    - TensorFlow.jil - https://github.com/malmaud/TensorFlow.jl
    - Mocha - https://github.com/pluskid/Mocha.jl

Other Applications

- Financial Analysis and Quants Packages
  - Miletus - a DSL for financial contracts
  - JuliaDB - a high performance in-memory and distributed database
  - JuliaInXL - call Julia from Excel sheets
  - Bloomberg - providing access to Bloomberg financial data
- Climate Modelling
  - Solves the two-language problem
  - Official language of CliMA - Climate Modelling Alliance
  - Differential Equations - computational simulations
- Bioinformatics
  - Genomic Data Sets
  - BioJulia - DNA Sequence Analysis

Conclusion

- Julia is a very versatile programming language
- Notable uses:
    - Time-series analytics
    - Mathematical models (Notably economics)
    - Satellite simulation
    - CS 4824 (Machine Learning) here at Tech
- Interested in more: https://julialang.org/ or check out JuliaCon

Question

Given all of this information, what other sorts of applications would Julia be optimal for?

Notable uses:

- Time-series analytics
- Mathematical models (Notably economics)
- Satellite simulation
- CS 4824 (Machine Learning) here at Tech

Sources

- Hall, Matt. “Julia in a Nutshell.” *Agile*, Agile, 4 Sept. 2014, agilescientific.com/blog/2014/9/4/julia-in-a-nutshell.html.
- Heller, Martin. “What Is the Julia Language? A Fresh Approach to Numerical Computing.” *InfoWorld*, InfoWorld, 27 June 2018, www.infoworld.com/article/3284380/what-is-julia-a-fresh-approach-to-numerical-computing.html.
- “Julia (Programming Language).” *Wikipedia*, Wikimedia Foundation, 16 Sept. 2020, en.wikipedia.org/wiki/Julia_(programming_language).
- Karpinski, Stefan. “The Julia Language.” *The Julia Programming Language*, julialang.org/.
- Krill, Paul. “New Julia Language Seeks to Be the C for Scientists.” *InfoWorld*, InfoWorld, 18 Apr. 2012, www.infoworld.com/article/2616709/new-julia-language-seeks-to-be-the-c-for-scientists.html.
- Perkel, Jeffrey M. “Julia: Come for the Syntax, Stay for the Speed.” *Nature News*, Nature Publishing Group, 30 July 2019, www.nature.com/articles/d41586-019-02310-3.
- Rao, Vicky Singh, et al. “Julia Programming Language - A True Python Alternative.” *Technotification*, 22 Aug. 2018, www.technotification.com/2018/08/julia-programming-language.html.