Julia: A fresh approach to numerical computing

Jeff Bezanson, Alan Edelman, Stefan Karpinksi, Viral B. Shah
presented by Bogdan Enache

Department of Computer Science
University of Illinois at Urbana-Champaign

October 26, 2018

Overview

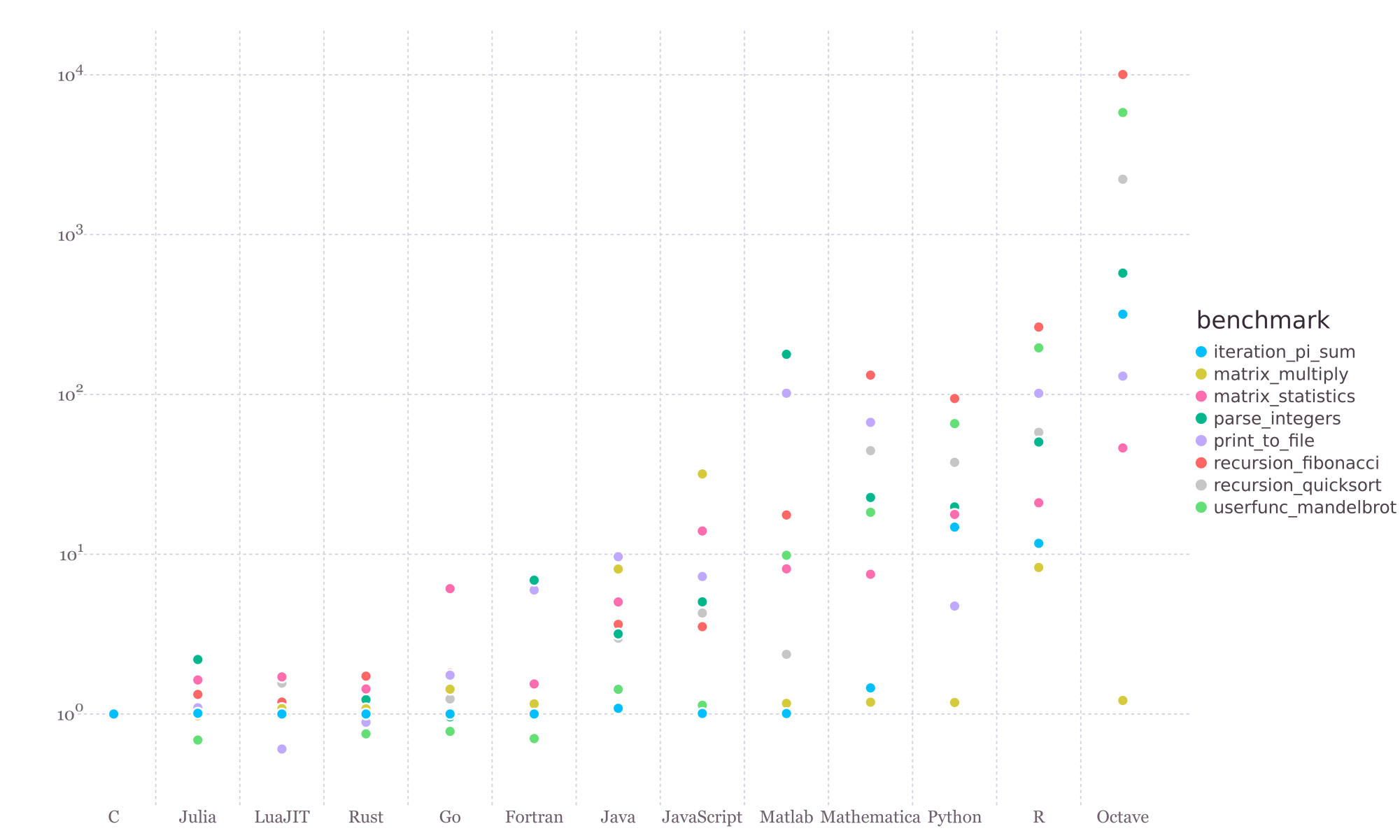
“Machine performance without sacrificing human convenience”
Julia questions these “laws of nature”:

- High level dynamic programs have to be slow
- One must prototype in one language and deploy in another for performance
- There are parts of a system made for the user, and others left to experts

“Julia is designed to be easy and fast.”

“Features that work well together”

1. An expressive type system, allowing optional type annotations
2. Multiple dispatch using these types to select implementations
3. Metaprogramming for code generation
4. A dataflow type inference algorithm allowing types of most expressions to be inferred
5. Aggressive code specialization against run-time types
6. Just-In-Time (JIT) compilation using the LLVM compiler framework
7. Julia’s carefully written libraries that leverage the language design


10^4
10^3
10^2
10^1
10^0
C
Julia
LuaJIT
Rust
Go
Fortran
Java
JavaScript
Matlab
Mathematica
Python
R
Octave
benchmark
iteration_pi_sum
matrix_multiply
matrix_statistics
parse_integers
print_to_file
recursion_fibonacci
recursion_quicksort
userfunc_mandelbrot

CUDAnative

```
function vadd(a, b, c)
    i = (blockIdx().x-1) * blockDim().x + threadIdx().x
    c[i] = a[i] + b[i]
    return
end

len = 100
a = rand(Float32, len)
b = rand(Float32, len)

d_a = CUDAdrv.Array(a)
d_b = CUDAdrv.Array(b)
d_c = similar(d_a)

@cuda (1,len) vadd(d_a, d_b, d_c)
c = Base.Array(d_c)
```

Conclusions

Pros:

- Macros / Metaprogramming
- JIT to native code makes things fast
- Dynamic features make development easy
- Emojis

Cons:

- 1-based indexing
- Garbage Collector
- Jit Overhead

References

The Paper.
https://docs.julialang.org/en/latest/manual/performance-tips/
https://nextjournal.com/sdanisch/julia-gpu-programming
https://docs.julialang.org/en/v1/manual/parallel-computing/index.html
https://julialang.org/blog/