Julia: A Fresh Approach to Parallel Programming

Alan Edelman
Massachusetts Institute of Technology

Abstract:

The Julia programming language is gaining enormous popularity. Julia was designed to be easy and fast. Most importantly, Julia shatters deeply established notions widely held in the applied community:

1) High-level, dynamic code has to be slow by some sort of law of nature;
2) It is sensible to prototype in one language and then recode in another language for speed or deployment; and
3) There are parts of a system for the programmer, and other parts best left untouched as they are built by the experts.

Julia shows the fascinating dance between specialization and abstraction. Specialization allows for custom treatment. We can pick just the right algorithm for the right circumstance and this can happen at runtime based on argument types (code selection via multiple dispatch). Abstraction recognizes what remains the same after differences are stripped away and ignored as irrelevant. The recognition of abstraction allows for code reuse (generic programming). A simple idea that yields incredible power.

Julia is many things to many people. In this talk we describe how Julia was built on the heels of our parallel computing experience with Star-P which began as an MIT research project and was a software product of Interactive Supercomputing. Our experience taught us that bolting parallelism onto an existing language that was not designed for performance or parallelism is difficult at best, and impossible at worst. One of our (not so secret) motivations to build Julia was to have the language we wanted for parallel numerical computing.

Brief Biography:

Professor Edelman has been working in the area of high performance computing systems, networks, software, and algorithms for 30 years. He also works on the theory and applications of Random Matrix Theory. He introduced the graduate course at MIT in high performance computing in 1994 and has trained a generation of students (from computer science, engineering, and the natural sciences) who have now gone on to work at national labs, universities, and industry. He has won many prizes for his work including the prestigious Gordan Bell Prize. He was the founder of Interactive Supercomputing, a company acquired by Microsoft in its fifth year employing nearly 50 people. He holds and has applied for several patents in the area of high performance computing networks, algorithms, and software. He is widely recognized for his broad expertise in hardware, software, networks, algorithms, and applications. With Julia he sees a new fresh approach to high performance technical computing that is likely to replace our current less productive, lower performing, and highly cumbersome practices.