There exist many preconceived notions about how C, Java, and Go perform in various different applications. Our research explores and attempts to quantify exactly how the programming paradigms, data models, compiler optimizations, and other artifacts of various languages affect performance in different applications.

Our goals is to generate hard data to enable better software design and language decisions to be made when developing real-world software.

**ABSTRACT**

**GOALS**

- Generate Hard Evidence / Data
- Enable better software design through data
- Enable better language decisions through data

**LANGUAGES AND ALGORITHMS**

- Levenshtein Edit Distance
- Fibonacci Number Generation
- NBody Force Simulation

**MEMORY USAGE DATA**

Observation: Java always has a larger memory footprint due to the JVM, even when objects are avoided

Observation: gcc-02, gcc-03 unroll tail recursion into a loop causing significant performance increase in C

**NBODY DATA**

**RECURSIVE DATA**

**OP Amized NBody** RunAme (s)

**OP Amized** Levenshtein Edit Distance RunAme (s)

**Simple Levenshtein Edit Distance RunAme** (s)

**Simple NBody** RunAme (s)

Implementation > Language Choice

Compilers Matter!

JVM Overhead Is significant, Java regardless

Optimizations can be very fast.