# Empirical Evaluation of Python-based Tools for Distributed Computing on the Raspberry Pi


**Jacob J. Morra and Qusay H. Mahmoud**
Department of Electrical, Computer and Software Engineering
University of Ontario Institute of Technology


## Motivation

Benchmarking of Python-based Modules, Libraries, and APIs for Distributed Computing could provide value to a broad community of developers, for the following primary reasons:

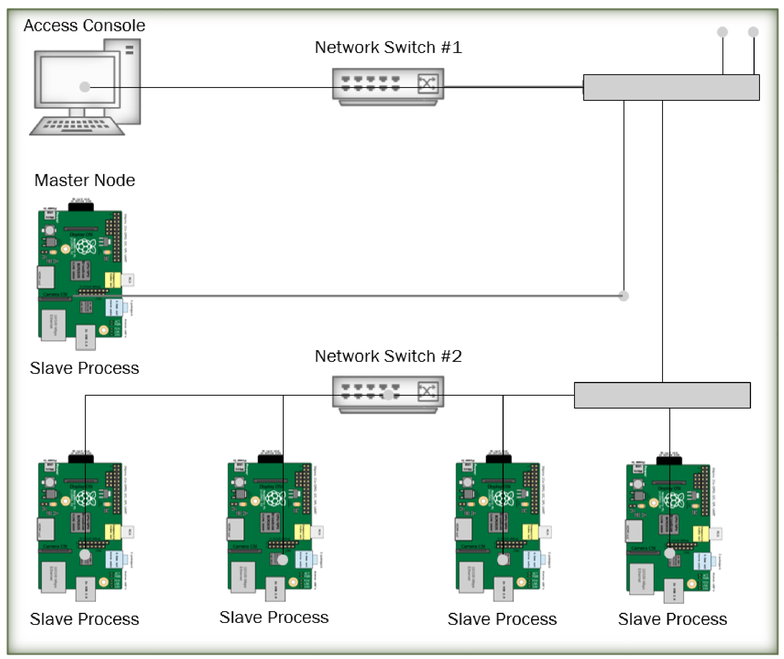
- **Cost-effective distributed computing in hobbyist communities could benefit from an optimal tool**
- **Efficiencies gained in run times could be scalable to highly intensive applications in scientific computing**

## Research Objective

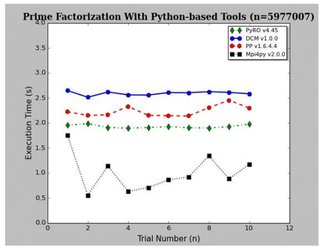
Using a cluster of Raspberry Pi's as an inexpensive test bench:

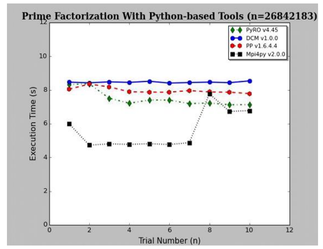
- **Observe, record, and compare run times for four Python-based Tools – Python Remote Objects v4.45 (PyRO), Distributed Computing Module v1.0.0 (DCM), Parallel Python v1.6.4.4 (PP), and Mpi4py v2.0.0**
- **Observe, record, and compare run times for varying distributed cluster sizes (C=1 to C=5)**
- **Base results around two sets of relatively load-balanced test algorithms**
  1. **Prime Factorization**
  2. **Pi Determination**

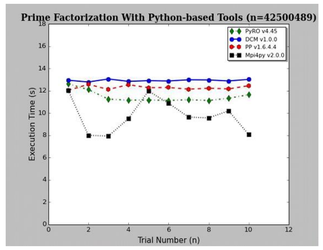
## Hardware Setup

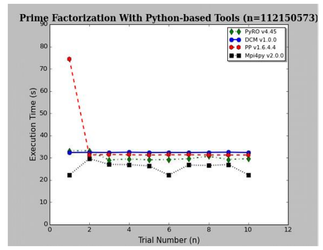


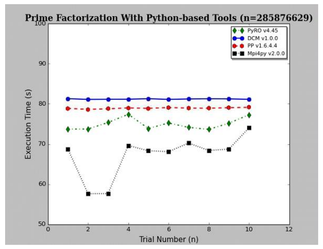

## Prime Factorization Results

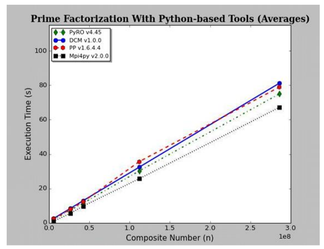


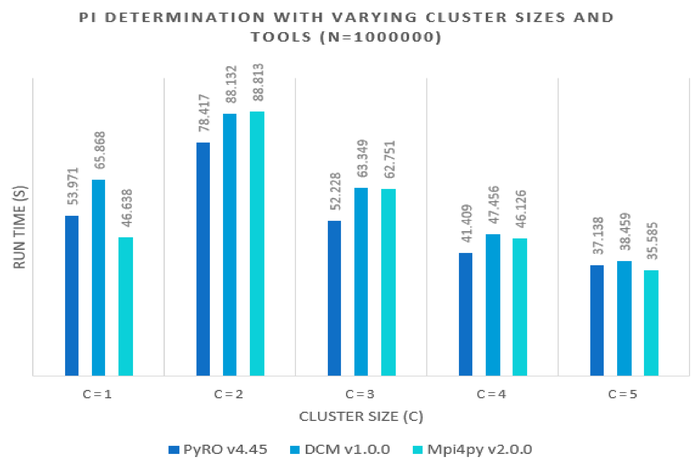



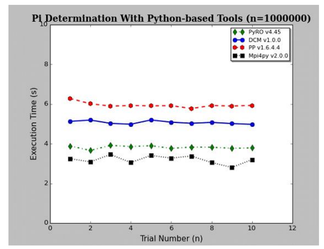



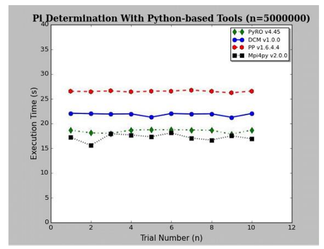

- Across five sets of ten run times per tool, DCM consistently exhibits the longest run times for Prime Factorization
- Mpi4py exhibits the shortest run times for Prime Factorization; however, it also yields the greatest variance in results

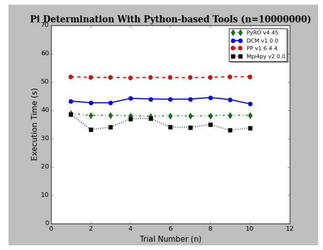
## Cluster Size Comparison

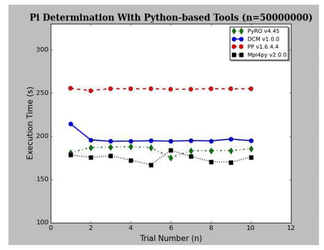


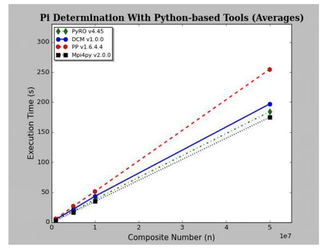

## Pi Determination Results




- Across four sets of ten run times per tool, PP consistently exhibits the longest run times for Pi Determination
- Mpi4py exhibits the shortest run times for Pi Determination; however, it also yields the greatest variance in results

## Conclusion

**Based on empirical results from the benchmarking tests:**

- Mpi4py is recommended as a baseline for the execution of distributed tasks;
- PyRO is recommended as an alternate choice for distributed computing, with benefits over Mpi4py with respect to ease of use, API availability (i.e. DCM), and lower run time variance;

## Future Work

**The following tasks would be prioritized in future research:**

- Performing the same set of benchmarks with more powerful machines, or alternatively with a significantly larger cluster size;
- More testing with alternate C++-based tools or Java-based tools for distributed computing