PyPy : A fast Python Virtual Machine

Romain Guillebert

Pycon IE

October 12th, 2014

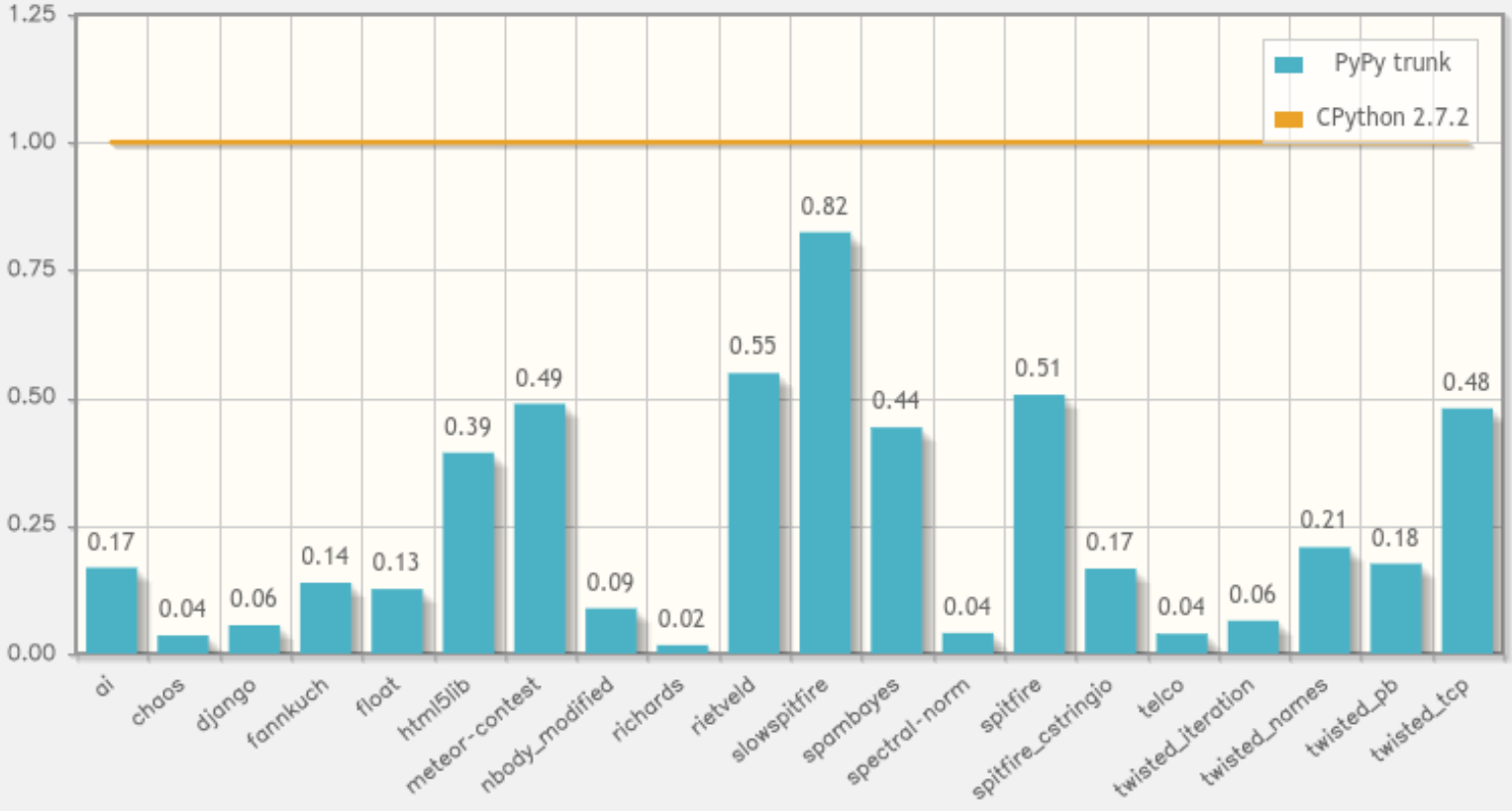
Me

- rguillebert on twitter and irc
- PyPy contributor since 2011
- NumPyPy contributor
- Software consultant (hire me !)

Introduction

- "PyPy is a fast, compliant alternative implementation of the Python language"
- Aims to reach the best performance possible without changing the syntax or semantics
- Supports x86, x86_64, ARM
- Production ready
- MIT Licensed

Speed



Plot 1: The above plot represents PyPy trunk (with JIT) benchmark times normalized to CPython. Smaller is better.

It depends greatly on the type of task being performed. The geometric average of all benchmarks is 0.15 or **6.6** times *faster* than CPython

Speed

- Automatically generated tracing just-in-time compiler
- Generates linear traces from loops
- Generates efficient machine code based on runtime observations
- Removes overhead when unnecessary
- But Python features which need require overhead remain available (frame introspection, pdb)

Performance ?

- Things get done faster
- Lower latency
- Less servers for the same performance

Demo

- Real-time edge detection

Compatibility

- Fully compatible with CPython 2.7 & 3.2 (minus implementation details)
- Partial and slow support of the C-API
- Alternatives might exist

Ecosystem

- We should (slowly, incrementally) move away from the C extension API
  - Makes assumptions on refcounting, object layout, the GIL
  - The future of Python is bound to the future of CPython (a more than 20 years old interpreter)
  - It's hard for a new Python VM without C extension support to get traction (not only PyPy)
- This doesn't mean we should lose Python's ability to interface with C easily
- CFFI is the PyPy team's attempt at solving this

CFFI (1/2)

- Where do we go from here ?
- CFFI is a fairly new way of interacting with C in an implementation independant way
- Very fast on PyPy
- Decently fast on CPython
- The Jython project is working on support

CFFI (2/2)

- More convenient, safer, faster than ctypes
- Can call C functions easily, API and ABI mode
- Python functions can be exposed to C
- Already used by pyopenssl, psycopg2cffi, pygame_cffi, lxml_cffi
- Other tools could be built on top of it (Cython cffi backend ?)

Success stories

Magnetic is the leader in online search retargeting, with a large, high volume, performance-critical platform written in Python. [...]
The Magnetic bidders were ported from CPython to PyPy, yielding an overall 30% performance gain.

- Julian Berman
  magnetic.com

Success stories

Currently we have improvements in raw performance (read: response times) that span from 8% to a pretty interesting 40%, but we have a peak of an astonishing 100-120% and even more.
Take into the account that most of our apps are simple "blocking-on-db" ones, so a 2x increase is literally money.

- Roberto De Ioris
  Unbit

Success stories

In addition to this our main (almost secret) objective was reducing resource usage of the application servers, which directly translates to being able to host more customers on the same server.

- Roberto De Ioris
  Unbit

Future

- Python 3.3
- NumPyPy
- STM
- You can donate to help the progress of these features : pypy.org

Questions

- Questions ?