Administrivia

- HW1 due this Sunday, Oct 4.
- HW1 clarification:
  - Almost all computation should be via RDDs, with limited calls to collect(), take(), or other Actions.
  - … except P6, which explicitly requires repeated Actions to generate text.
GIL Discussion
GIL BATTLE ROYALE!
Why a GIL?
Why a GIL?

- Thread safety:
  - Makes every built-in type thread-safe.
  - Reference counting Garbage Collector.
  - C extensions can ignore threads.
Why is the GIL a problem?
Problems

- Speed penalty, even for single threaded.
- “True” multithreading isn’t possible.
- Complicated scheduler / compute / IO interactions.
- The GIL is not a scheduler (maybe in 3.6 or later).
How much does the GIL matter?
“It is only in multithreaded programs that spend a lot of time [...] interpreting CPython bytecode, that the GIL becomes a bottleneck.”
Interpreted Code

• If you’re writing pure Python, you probably don’t care that much about performance.

• If you care about performance, you’re probably writing a Cython or C extension.
But…

• Watch David Beazley’s talks, particularly for cases where the GIL causes crazy results.

• But many of these have been fixed in 3.2+
Opinionated mode ON.
Knuth, 1974

“[P]rogrammers have spent far too much time worrying about efficiency in the wrong places and at the wrong times;

premature optimization is the root of all evil (or at least most of it) in programming.”
Where I get my opinions

- Academia / Research work since 2007 (warning!).
- C, C++, Matlab, Python, PERL, Dylan, LISP.
Software Engineering
The model.
Software Engineering

The reality.
Python is a Glue Language

- I can hook an image recognizer running on a GPU
- to a stream of images culled from Instagram,
- store the results in an MySQL database,
- and serve cat pictures from a web server…
Python is a Glue Language

• I can hook an image recognizer running on a GPU to a stream of images culled from Instagram,
• store the results in an MySQL database,
• and serve cat pictures from a web server…
• … in a couple of hours.
Python Performance Stack

- Python
- Numpy
- Cython & C
- PyCUDA / PyOpenCL
Opinionated mode OFF (supposedly).
map / map_async

- map()
  - wait for result

- map_async()
  - returns immediately.
  - return a MapResult type, which get() method.
  - (also ready() and wait() methods)
map_async example
Queues & Pipes

- Interprocess, first-in, first-out.
- Pipe
  - One input, one output.
- Queue
  - Multiple writers, multiple readers
Queue example
Communication costs

- Serialization (or marshaling)
  - Must convert Python objects to byte strings (and back)
- Transport
- Locking
  - Make sure Queue is accessed safely.
Shared memory

• It actually is possible to share memory between processes.

• multiprocessing.Value, multiprocessing.Array

• (...and now you need locks...)
Coming up

- Threading module in Python
- Cython, threads, and OpenMP extensions.