HotShot: The New Python Profiler

10th International Python Conference
Alexandria, Virginia

Fred L. Drake, Jr.
fdrake@zope.com

ZOPE

©2002 Zope Corporation. All Rights Reserved.

What is a profiler?

- A tool that...
  - Slows program execution
  - Masks timing-related problems
  - Requires low-level support in the interpreter
  - Measures where time is spent

©2002 Zope Corporation. All Rights Reserved.

ZOPE

But I thought…

…profiling was supposed to make my program faster?

- No, that's your job as a programmer
- No, the computer isn't that smart
- No, the profiler makes the computer do *more* work, not less

©2002 Zope Corporation. All Rights Reserved.

ZOPE

But…

I wanna go faster!

©2002 Zope Corporation. All Rights Reserved.

ZOPE

Speeding up

How do we get our apps to be faster?

1. Determine how fast we want to go
   - We can't go faster than systems we're relying on, like disk I/O or network communications
2. See how fast we *are* going
   - Not profiling, just look at the clock
   - Stop now if it's fast enough
3. Find where we're wasting time
   1. Profile the application
   2. Analyze the data

ZOPE

©2002 Zope Corporation. All Rights Reserved.

Speeding up (cont.)

4. Change the program to do less work
   - Find a better algorithm (often the best)
   - Change the code to run faster (usually tedious)
5. Go back to step 2

©2002 Zope Corporation. All Rights Reserved.

Profilers for Python

`profile` module

- Part of the standard library
- 100% Pure Python
- As of Python 2.2, it's even fairly robust
- Large in-memory data structures
- Documented
- Incredibly slow

©2002 Zope Corporation. All Rights Reserved.

ZOPE

Profilers for Python (cont.)

HotShot (`hotshot` package)

- Part of the standard library
- New in Python 2.2 (preliminary)
- Foundation written in C
- Analysis tools written in Python
- Can support per-line information
- Logs data to disk (lower memory use)
- Still pretty raw
- Undocumented

©2002 Zope Corporation. All Rights Reserved.

ZOPE

Why two different profilers?

Profiling is hard to get right in practice:

- “Watching” may be a simple concept, but there are many details to be careful about
- A profiler needs to “see” many kinds of events
  - Function calls/returns
  - Cycle detector operation
  - Context switches (both thread and process, depending on how we measure time)
  - Source line changes (if we want that level of detail)

©2002 Zope Corporation. All Rights Reserved.

ZOPE

And in practice?

`profile` sees these events:

- Function calls/returns

HotShot sees these events:

- Function calls/returns
- Source line changes (optional)

©2002 Zope Corporation. All Rights Reserved.

ZOPE

What makes HotShot better?

We've changed how Python calls the profiler

- Old way: call a Python function for each event
  - `sys.setprofile()` sets the function
- New way: call a C function for each event
  - New C API function sets the function

Can be used in a "coverage-only" mode

- Doesn't keep track of timing, only of what code is executed

©2002 Zope Corporation. All Rights Reserved.

What about the other events?

- Only changed *how* the profiler is called
- Need to change the interpreter to call the profiler for the other events

©2002 Zope Corporation. All Rights Reserved.

ZOPE

What needs to be done?

- Need to call profiler when entering and exiting the cycle detector
- Need to call profiler on thread context switches
- Should allow separate functions for each event type (allows the profiler to do even less work)
- Allow Python→C and C→Python transitions to be monitored

ZOPE

©2002 Zope Corporation. All Rights Reserved.

Using HotShot

Can we use HotShot today?

- Yes

Will it help?

- Maybe.
- Basic profiling won't take as long:
  - The `profile` module could easily add 100% or more to execution time
  - HotShot usually adds much less (~20%)
  - (Exact speedup is strongly application dependent)

ZOPE

©2002 Zope Corporation. All Rights Reserved.

A Simple Example

```
import hotshot
import hotshot.stats

def main():
    # do something useful here
    print "Hello, Python 10!"

profiler= hotshot.profile("logfile.dat")
profiler.run("main()")
profiler.close()

stats = hotshot.stats.load("logfile.dat")
```


©2002 Zope Corporation. All Rights Reserved.

So what is that `stats` thing?

A `pstats.Stats` instance, of course!

- The same thing we're used to when using the old profiler…
- with the same old methods for getting information out

But isn't there something more?

- Yes and no…

ZOPE

©2002 Zope Corporation. All Rights Reserved.

Additional capabilities

Per-line information

- Stored by the profiler, but still hard to use
- Hopefully Python 2.3 will include better tools
- IDE integration might be a good idea

Application-specific data

- The profiler's `addinfo(`*key*`, `*value*`)` method can be used to track additional information
- No good tool support
- API might not be quite right, so may change

©2002 Zope Corporation. All Rights Reserved.

ZOPE

Getting more information

Need to describe some profiler mechanics:

- `profile` module maintained timing information in a shadow stack as it ran
- HotShot keeps very little information while the program is running
  - Events are logged to a file in a compact form
  - Log file must be decoded to retrieve data
  - Line number and “addinfo” data are just additional event types

ZOPE

©2002 Zope Corporation. All Rights Reserved.

Getting more information (cont.)

Two levels of log decoding:

- Low-level: `_hotshot.logreader()`
  - Iterator object
  - Returns a 4-tuple; first item is the record type
  - Times are reported in deltas from the previous event that included time data, not easy to use

©2002 Zope Corporation. All Rights Reserved.

ZOPE

Getting more information (cont.)

- Higher-level: `hotshot.log.LogReader`
  - Iterator object
  - Returns a 3-tuple; the second item is a 3-tuple giving source location
  - Get “addinfo” data by overriding the `addinfo(`*key*`, `*value*`)` method
  - Masks events related to log format

©2002 Zope Corporation. All Rights Reserved.

ZOPE

Coverage mode

`_hotshot.coverage(`*logfn*`)`

- Creates a profiler that doesn't take any time measurements
- Use your choice of log reader to decode the log, but ignore the timing information (it will all be `-1` anyway)
- No helpful tools, yet

©2002 Zope Corporation. All Rights Reserved.

Coverage mode (cont.)

Why use HotShot for coverage analysis?

- Nicer tools are available (from Skip Montanaro, Gareth Rees)
- It's just incredibly fast!
- There's almost no Python-perceivable overhead when we don't collect timing data

©2002 Zope Corporation. All Rights Reserved.

ZOPE

Outstanding issues

- Need better ways to collect timing data
  - `gettimeofday()` on Unix
    - Good resolution for most platforms
    - Doesn't deal with interference from other threads or processes at all
  - `QueryPerformanceCounter()` on Windows
    - Good resolution
    - Tim hasn't complained

©2002 Zope Corporation. All Rights Reserved.

Outstanding issues (cont.)

- Need better user tools
  - Just a matter of time
- Documentation!

©2002 Zope Corporation. All Rights Reserved.

©2002 Zope Corporation. All Rights Reserved.

ZOPE