Making Your Python Code Fast

Andrew Bennetts
andrew@bemusement.org
Bazaar team, Canonical

Quick quiz:

What's the fastest possible Python program?

It's not

```
python -c ""
```

Answer:

```
python -S -c "import os; os._exit(0)"
```

Tools I use to investigate performance

timeit

```
python -m timeit -s "setup"
"statement"
```

```
python -m timeit
-s "import string"
-s "seq = string.letters"
"'z' in seq"
```

Result:

```
10000000 loops, best of 3: 0.157 usec
per loop
```

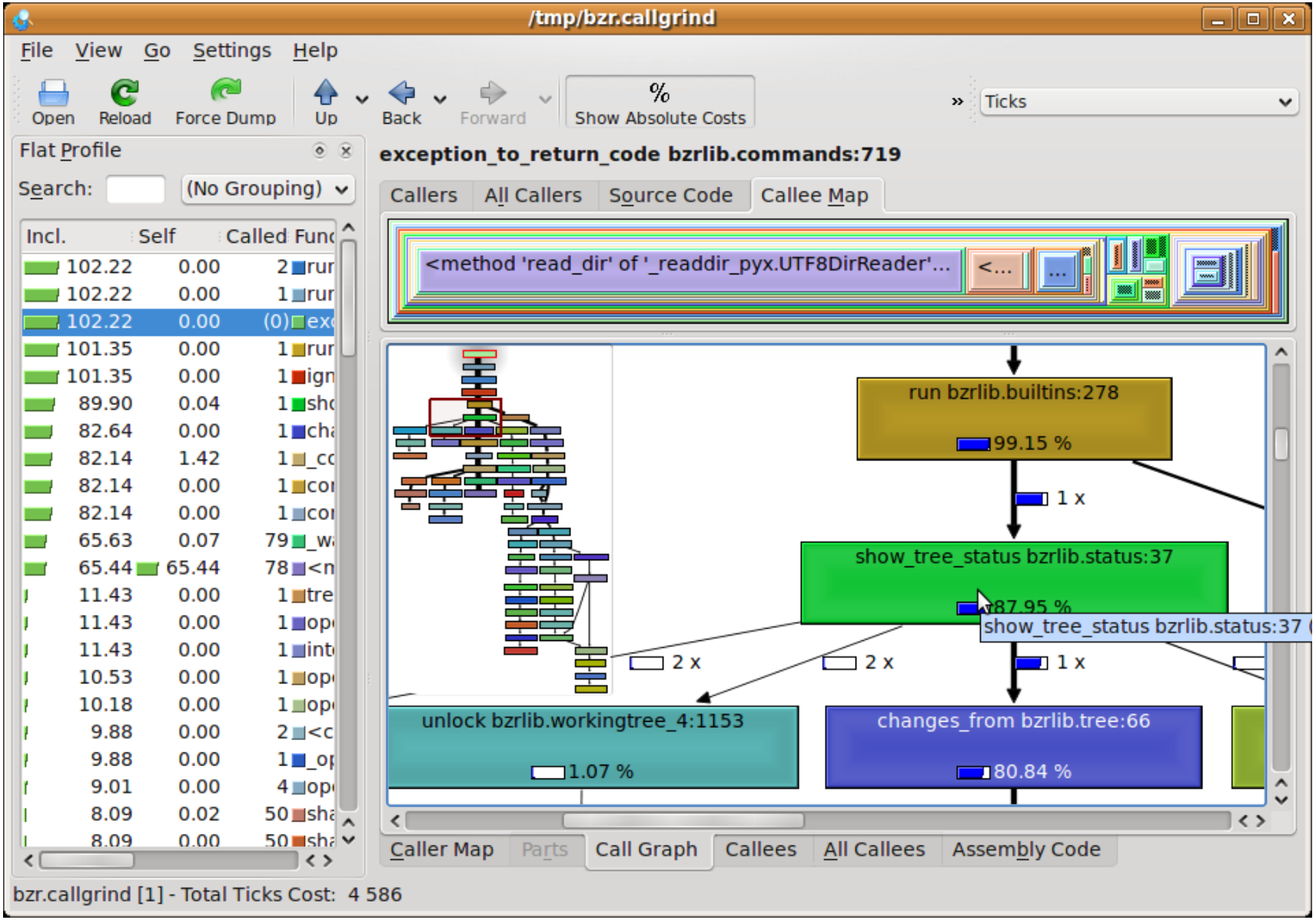
```
python -m timeit
-s "import string"
-s "seq = set(string.letters)"
"'z' in seq"
```

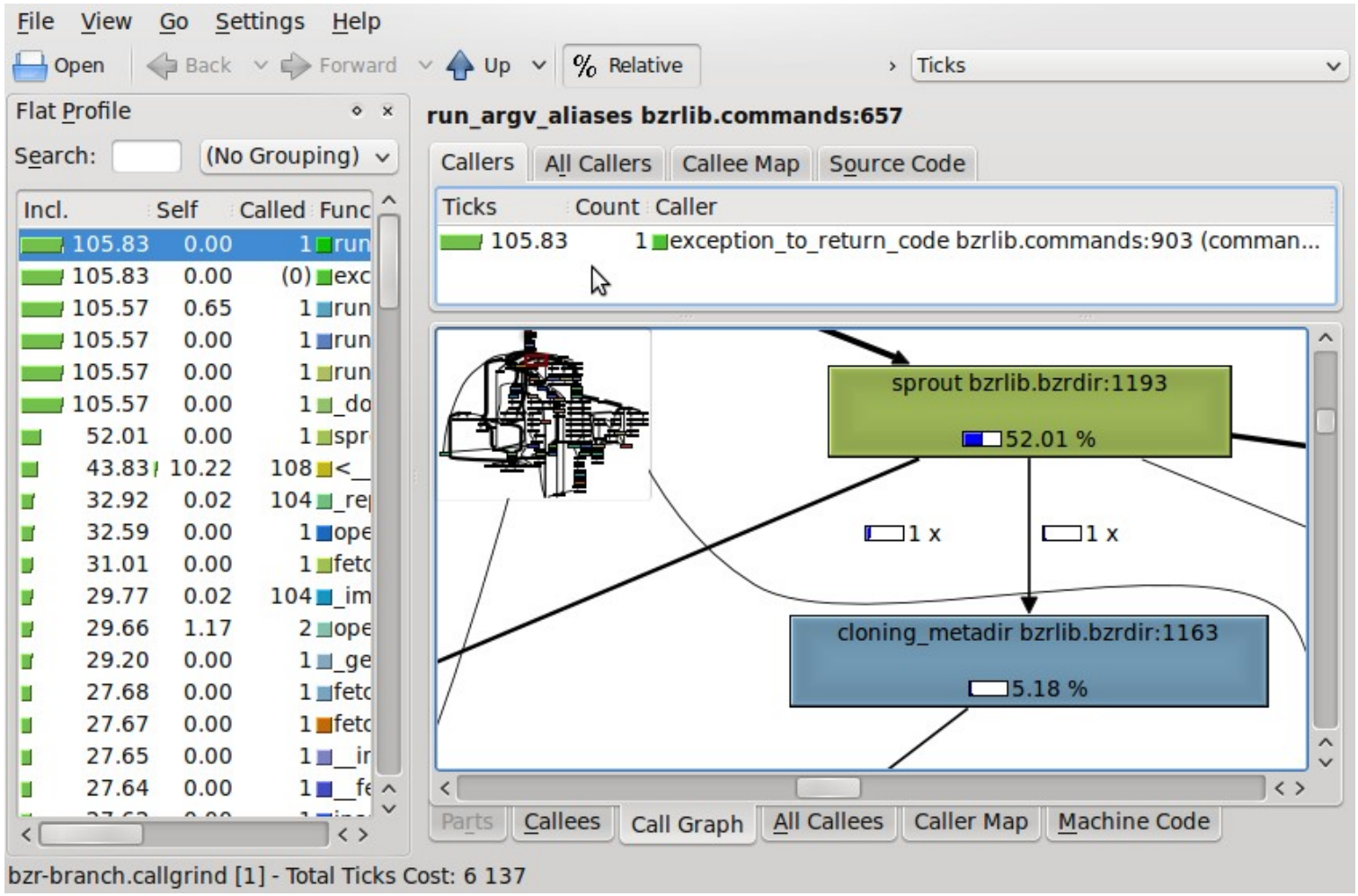
Result:

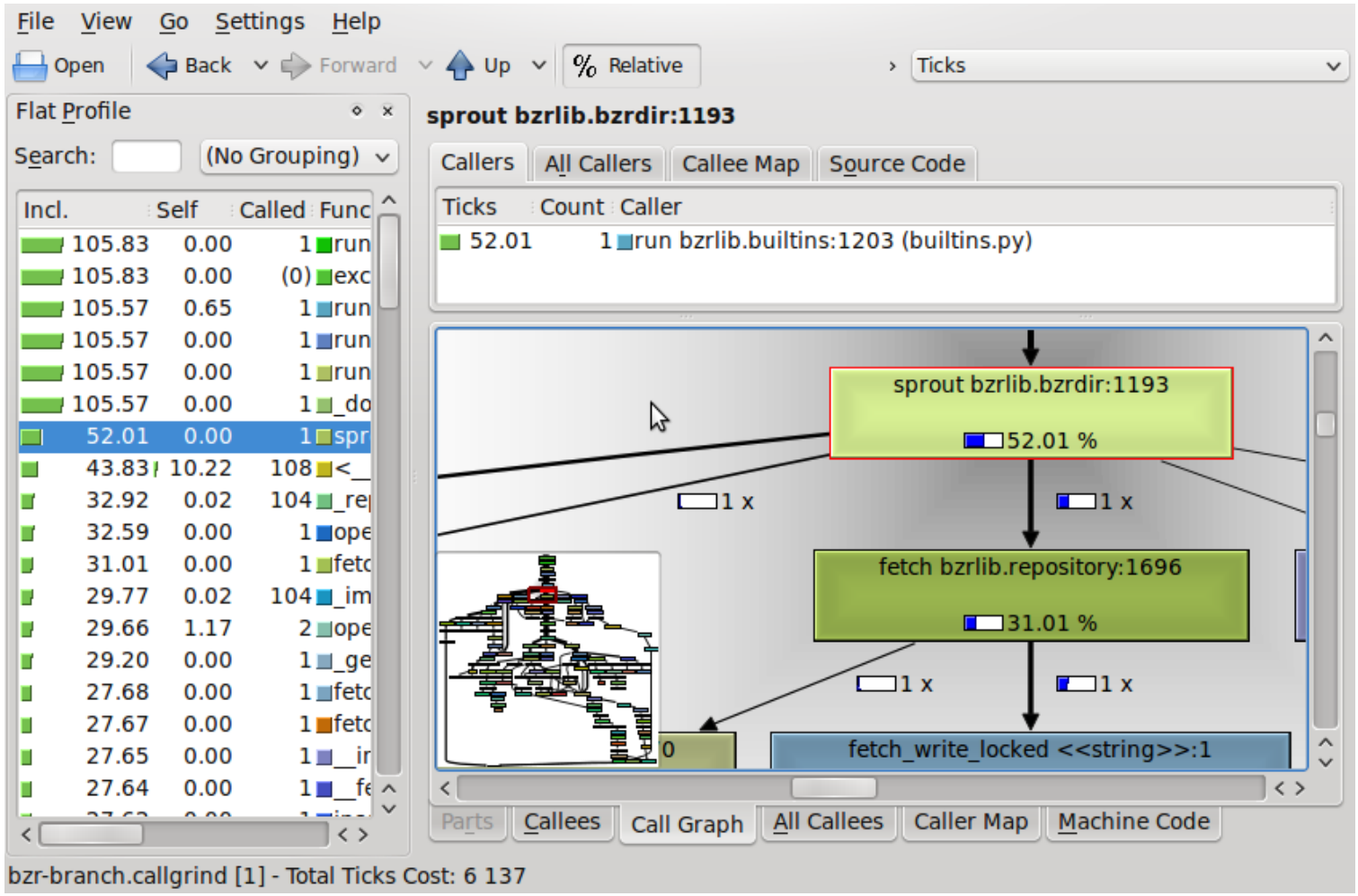
```
10000000 loops, best of 3: 0.117 usec
per loop
```

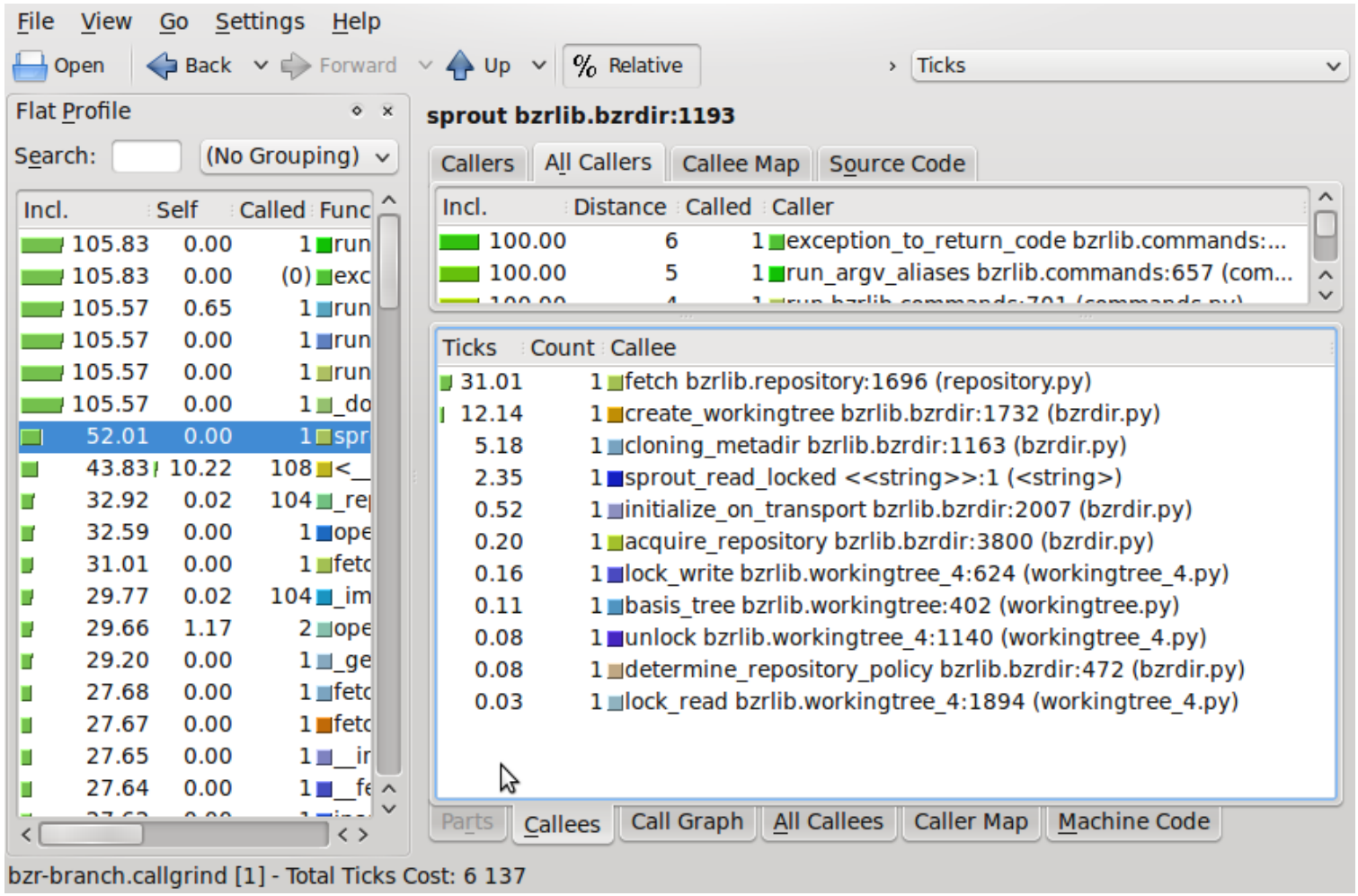
lsprof

lsprof + kcachegrind == awesome


/tmp/bzr.callgrind
File View Go Settings Help
Open Reload Force Dump Up Back Forward Show Absolute Costs
Ticks
Flat Profile
Search:
(No Grouping)
Incl. Self Called Func
102.22 0.00 2 run
102.22 0.00 1 run
102.22 0.00 (0) exe
101.35 0.00 1 run
101.35 0.00 1 ign
89.90 0.04 1 sho
82.64 0.00 1 cha
82.14 1.42 1 _co
82.14 0.00 1 cor
82.14 0.00 1 cor
65.63 0.07 79 _wa
65.44 65.44 78 <n
11.43 0.00 1 tre
11.43 0.00 1 op
11.43 0.00 1 int
10.53 0.00 1 op
10.18 0.00 1 op
9.88 0.00 2 <c
9.88 0.00 1 _op
9.01 0.00 4 op
8.09 0.02 50 sha
8.09 0.00 50 sha
exception_to_return_code bzrlib.commands:719
Callers All Callers Source Code Callee Map
<method 'read_dir' of '_readdir_pyx.UTF8DirReader'...
<...
run bzrlib.builtins:278
99.15 %
1 x
show_tree_status bzrlib.status:37
87.95 %
show_tree_status bzrlib.status:37
2 x
2 x
1 x
unlock bzrlib.workingtree_4:1153
1.07 %
changes_from bzrlib.tree:66
80.84 %
Caller Map Parts Call Graph Callees All Callees Assembly Code
bzr.callgrind [1] - Total Ticks Cost: 4 586


File View Go Settings Help
Open
Back
Forward
Up
% Relative
Ticks
Flat Profile
Search:
(No Grouping)
Incl. Self Called Func
105.83 0.00 1 run
105.83 0.00 (0) exc
105.57 0.65 1 run
105.57 0.00 1 run
105.57 0.00 1 run
105.57 0.00 1 _do
52.01 0.00 1 spr
43.83 10.22 108 <_
32.92 0.02 104 _re
32.59 0.00 1 ope
31.01 0.00 1 fetc
29.77 0.02 104 _im
29.66 1.17 2 ope
29.20 0.00 1 _ge
27.68 0.00 1 fetc
27.67 0.00 1 fetc
27.65 0.00 1 __ir
27.64 0.00 1 __fe
run_argv_aliases bzrlib.commands:657
Callers
All Callers
Callee Map
Source Code
Ticks Count Caller
105.83 1 exception_to_return_code bzrlib.commands:903 (comman...
sprout bzrlib.bzrdir:1193
52.01 %
1 x
1 x
cloning_metadir bzrlib.bzrdir:1163
5.18 %
Parts
Callees
Call Graph
All Callees
Caller Map
Machine Code
bzr-branch.callgrind [1] - Total Ticks Cost: 6 137


File View Go Settings Help
Open Back Forward Up % Relative Ticks
Flat Profile
Search: (No Grouping)
Incl. Self Called Func
105.83 0.00 1 run
105.83 0.00 (0) exc
105.57 0.65 1 run
105.57 0.00 1 run
105.57 0.00 1 run
105.57 0.00 1 _do
52.01 0.00 1 spr
43.83 10.22 108 <_
32.92 0.02 104 _re
32.59 0.00 1 ope
31.01 0.00 1 fetc
29.77 0.02 104 _im
29.66 1.17 2 ope
29.20 0.00 1 _ge
27.68 0.00 1 fetc
27.67 0.00 1 fetc
27.65 0.00 1 __ir
27.64 0.00 1 __fe
sprout bzrlib.bzrdir:1193
Callers All Callers Callee Map Source Code
Ticks Count Caller
52.01 1 run bzrlib.builtins:1203 (builtins.py)
sprout bzrlib.bzrdir:1193
52.01 %
1 x
1 x
fetch bzrlib.repository:1696
31.01 %
1 x
1 x
0
fetch_write_locked <<string>>:1
Parts Callees Call Graph All Callees Caller Map Machine Code
bzr-branch.callgrind [1] - Total Ticks Cost: 6 137


File View Go Settings Help
Open Back Forward Up % Relative Ticks
Flat Profile
Search: (No Grouping)
Incl. Self Called Func
105.83 0.00 1 run
105.83 0.00 (0) exc
105.57 0.65 1 run
105.57 0.00 1 run
105.57 0.00 1 run
105.57 0.00 1 _do
52.01 0.00 1 spr
43.83 10.22 108 <_
32.92 0.02 104 _re
32.59 0.00 1 ope
31.01 0.00 1 fetc
29.77 0.02 104 _im
29.66 1.17 2 ope
29.20 0.00 1 _ge
27.68 0.00 1 fetc
27.67 0.00 1 fetc
27.65 0.00 1 __ir
27.64 0.00 1 __fe
sprout bzrlib.bzrdir:1193
Callers All Callers Callee Map Source Code
Incl. Distance Called Caller
100.00 6 1 exception_to_return_code bzrlib.commands:...
100.00 5 1 run_argv_aliases bzrlib.commands:657 (com...
Ticks Count Callee
31.01 1 fetch bzrlib.repository:1696 (repository.py)
12.14 1 create_workingtree bzrlib.bzrdir:1732 (bzrdir.py)
5.18 1 cloning_metadir bzrlib.bzrdir:1163 (bzrdir.py)
2.35 1 sprout_read_locked <<string>>:1 (<string>)
0.52 1 initialize_on_transport bzrlib.bzrdir:2007 (bzrdir.py)
0.20 1 acquire_repository bzrlib.bzrdir:3800 (bzrdir.py)
0.16 1 lock_write bzrlib.workingtree_4:624 (workingtree_4.py)
0.11 1 basis_tree bzrlib.workingtree:402 (workingtree.py)
0.08 1 unlock bzrlib.workingtree_4:1140 (workingtree_4.py)
0.08 1 determine_repository_policy bzrlib.bzrdir:472 (bzrdir.py)
0.03 1 lock_read bzrlib.workingtree_4:1894 (workingtree_4.py)
Parts Callees Call Graph All Callees Caller Map Machine Code
bzr-branch.callgrind [1] - Total Ticks Cost: 6 137

Search Google for how to get calltree format output from lsprof
:(

strace

```
$ strace -e open python -c ""
open("/etc/ld.so.cache", O_RDONLY)      = 3
```

... 165 lines later ...

```
open("/usr/lib/python2.6/encodings/utf_8.so",
O_RDONLY|O_LARGEFILE) = -1 ENOENT (No such file or
directory)
open("/usr/lib/python2.6/encodings/utf_8module.so"
, O_RDONLY|O_LARGEFILE) = -1 ENOENT (No such file
or directory)
open("/usr/lib/python2.6/encodings/utf_8.py",
O_RDONLY|O_LARGEFILE) = 3
open("/usr/lib/python2.6/encodings/utf_8.pyc",
O_RDONLY|O_LARGEFILE) = 4
```

perf

(apt-get install linux-tools)

```
$ perf stat bzr status
```

... bzr output elided ...

```
 Performance counter stats for './bzr --no-plugins st':

     292.926379  task-clock-msecs          #      0.873 CPUs
            134  context-switches          #      0.000 M/sec
              0  CPU-migrations            #      0.000 M/sec
           3953  page-faults               #      0.013 M/sec
      489067925  cycles                    #   1669.593 M/sec
      477048124  instructions              #      0.975 IPC
        9805418  cache-references          #     33.474 M/sec
         494308  cache-misses              #      1.687 M/sec

    0.335459688  seconds time elapsed
```

```
$ perf record bzr status
[ perf record: Woken up 1 times to write data ]
[ perf record: Captured and wrote 0.129 MB perf.data (~5628
samples) ]

$ perf report
Overhead  Command         Shared Object  Symbol
........  .......  ...................  ......

   6.88%   python  /usr/bin/python2.6   [.] PyString_FromFormatV
   3.72%   python  /usr/bin/python2.6   [.] vgetargs1
   3.58%   python  /usr/bin/python2.6   [.] PyString_Format
   3.10%   python  /usr/bin/python2.6   [.] PyUnicodeUCS4_
                                             DecodeUnicodeEscape
   3.03%   python  .../libc-2.11.1.so   [.] __GI_memcpy
   1.95%   python  /usr/bin/python2.6   [.] product_dealloc
```

bzr's import profiler

```
$ bzr --profile-imports rocks
It sure does!
  cum inline name                                                  @ file:line
 17.3   1.4 bzrlib.commands                                       @ __main__:132
  8.5   0.9 + [Option]bzrlib.option                               @ bzrlib.commands:51
  7.7   2.8 ++optparse                                            @ bzrlib.option:20
  4.1   4.1 +++textwrap                                           @ optparse:77
  0.7   0.5 +++ [gettext]gettext                                  @ optparse:90
  0.3   0.1 ++++struct                                            @ gettext:49
  0.1   0.1 +++++ [*]_struct                                      @ struct:1
  6.3   1.0 + [disable_plugins, load_plugins]bzrlib.plugin @ bzrlib.commands:52
  5.3   2.1 ++ [osutils]bzrlib                                    @ bzrlib.plugin:36
```

... etc ...

Traceback sampling

e.g.

```
from signal import signal, SIGUSR1
import sys
from traceback import print_stack

def dump_stack(*args):
    print_stack(file=sys.stderr)

signal(SIGUSR1, dump_stack)
```

Recurring theme:

Make it really easy to get performance stats from your program.

```
sudo tc qdisc add dev lo root
  netem delay 500ms
```

```
(and
sudo tc qdisc del dev lo root)
```

tcptrace

```
a->b:                                  b->a:
  total packets:                 16      total packets:                 16
  ack pkts sent:                 15      ack pkts sent:                 16
  pure acks sent:                13      pure acks sent:                 2
  sack pkts sent:                 0      sack pkts sent:                 0
  dsack pkts sent:                0      dsack pkts sent:                0
  max sack blks/ack:              0      max sack blks/ack:              0
  unique bytes sent:            450      unique bytes sent:          18182
  actual data pkts:               1      actual data pkts:              13
  actual data bytes:            450      actual data bytes:          18182
  rexmt data pkts:                0      rexmt data pkts:                0

                                                    …

  ttl stream length:            450 bytes  ttl stream length:          18182 bytes
  missed data:                    0 bytes  missed data:                    0 bytes
  truncated data:               420 bytes  truncated data:             17792 bytes
  truncated packets:              1 pkts   truncated packets:            13 pkts
  data xmit time:             0.000 secs   data xmit time:             0.149 secs
  idletime max:               103.7 ms     idletime max:               99.9 ms
  throughput:                  1113 Bps    throughput:                44957 Bps
```

```
cpufreq-set -g performance
```

Thank you for your time!

Links, slides — http://bemusement.org/pycon2010

Mail me — andrew@bemusement.org

Stick around — Need for Speed, Graeme Cross