Profiling Python

The Rules of Optimization

1. Don't.
2. Don't... yet
3. Profile before optimizing

Identify a Goal

timeit

```
$ python -m timeit -r 5 'import runall; runall.run()'
10 loops, best of 5: 135 msec per loop
```

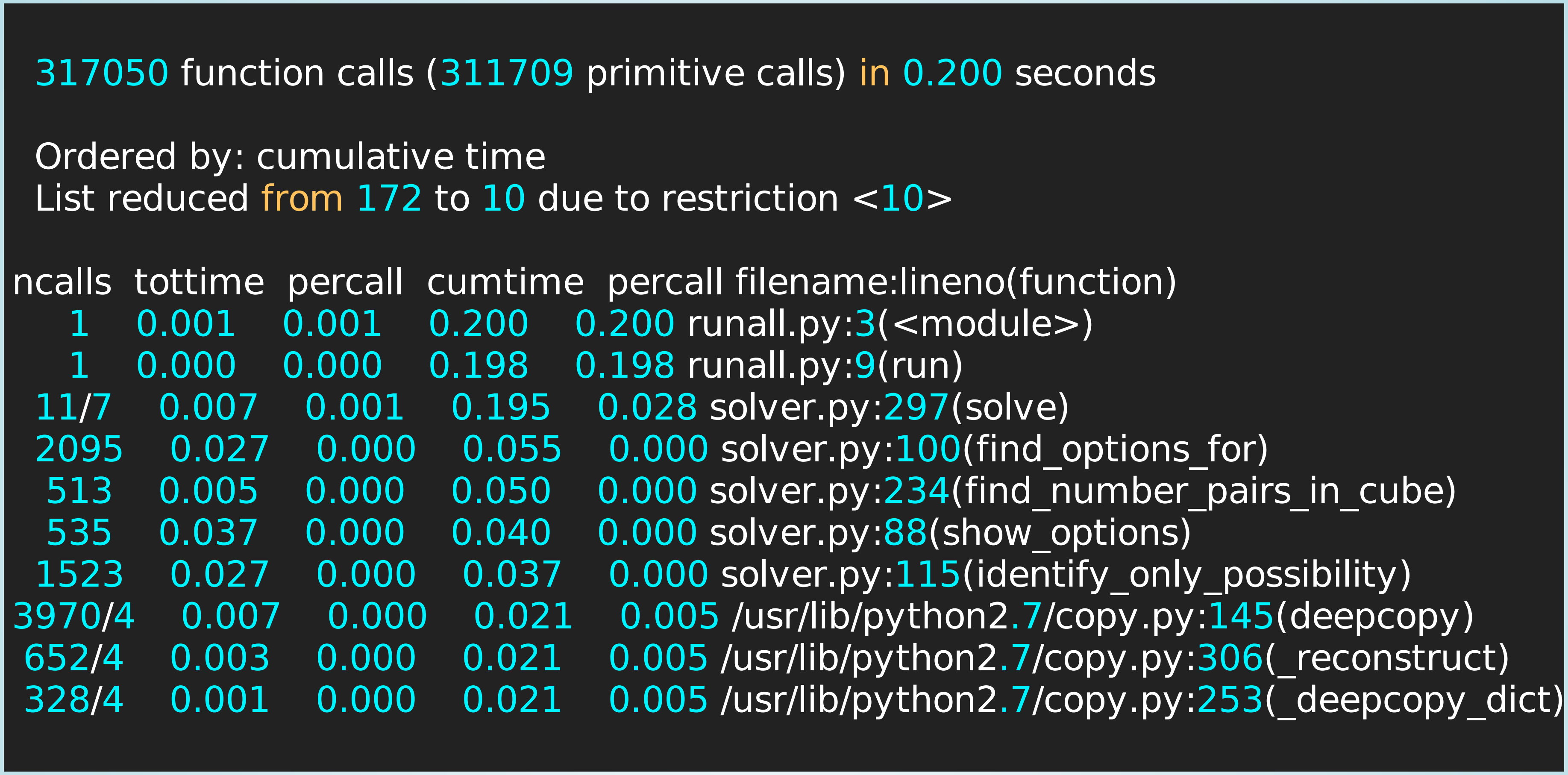
The Tools

profile and **cProfile** for profiling, produce **pstats.Stats**

profile and pstats

http://docs.python.org/library/profile.html

```
$ python -m cProfile -o runall.pstats runall.py
$
$ python -m pstats runall.pstats

runall.pstats% sort cumulative
runall.pstats% stats 10
```

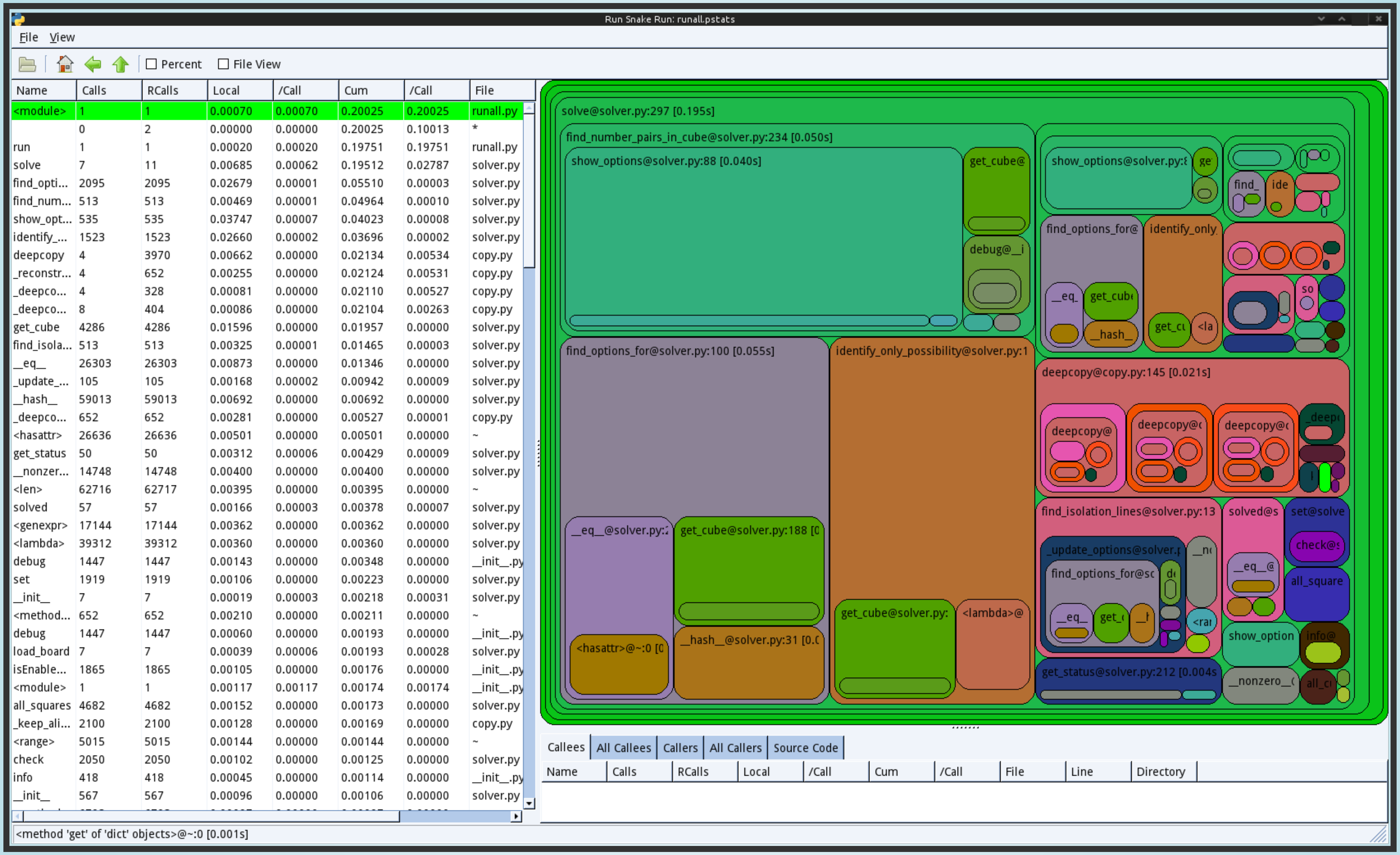


```
  317050 function calls (311709 primitive calls) in 0.200 seconds

  Ordered by: cumulative time
  List reduced from 172 to 10 due to restriction <10>

ncalls  tottime  percall  cumtime  percall filename:lineno(function)
     1    0.001    0.001    0.200    0.200 runall.py:3(<module>)
     1    0.000    0.000    0.198    0.198 runall.py:9(run)
  11/7    0.007    0.001    0.195    0.028 solver.py:297(solve)
  2095    0.027    0.000    0.055    0.000 solver.py:100(find_options_for)
   513    0.005    0.000    0.050    0.000 solver.py:234(find_number_pairs_in_cube)
   535    0.037    0.000    0.040    0.000 solver.py:88(show_options)
  1523    0.027    0.000    0.037    0.000 solver.py:115(identify_only_possibility)
3970/4    0.007    0.000    0.021    0.005 /usr/lib/python2.7/copy.py:145(deepcopy)
 652/4    0.003    0.000    0.021    0.005 /usr/lib/python2.7/copy.py:306(_reconstruct)
 328/4    0.001    0.000    0.021    0.005 /usr/lib/python2.7/copy.py:253(_deepcopy_dict)
```

Visualization

RunSnakeRun

http://www.vrplumber.com/programming/runsnakerun/


Run Snake Run: runall.pstats
File
View
Percent
File View
solve@solver.py:297 [0.195s]
find_number_pairs_in_cube@solver.py:234 [0.050s]
show_options@solver.py:88 [0.040s]
get_cube@
debug@__i
show_options@solver.py:
find_options_for@
identify_only
find_options_for@solver.py:100 [0.055s]
identify_only_possibility@solver.py:1
deepcopy@copy.py:145 [0.021s]
deepcopy@
find_isolation_lines@solver.py:13
solved@s
set@solve
check@s
all_square
_update_options@solver.
find_options_for@sc
__eq__@solver.py:
get_cube@solver.py:188 [0
<hasattr>@~:0 [0
__hash__@solver.py:31 [0.0
get_cube@solver.py:
<lambda>@
get_status@solver.py:212 [0.004s
show_option
info@
__nonzero__(
Callees
All Callees
Callers
All Callers
Source Code

Name	Calls	RCalls	Local	/Call	Cum	/Call	File
<module>	1	1	0.00070	0.00070	0.20025	0.20025	runall.py
	0	2	0.00000	0.00000	0.20025	0.10013	*
run	1	1	0.00020	0.00020	0.19751	0.19751	runall.py
solve	7	11	0.00685	0.00062	0.19512	0.02787	solver.py
find_opti...	2095	2095	0.02679	0.00001	0.05510	0.00003	solver.py
find_num...	513	513	0.00469	0.00001	0.04964	0.00010	solver.py
show_opt...	535	535	0.03747	0.00007	0.04023	0.00008	solver.py
identify_...	1523	1523	0.02660	0.00002	0.03696	0.00002	solver.py
deepcopy	4	3970	0.00662	0.00000	0.02134	0.00534	copy.py
_reconstr...	4	652	0.00255	0.00000	0.02124	0.00531	copy.py
_deepco...	4	328	0.00081	0.00000	0.02110	0.00527	copy.py
_deepco...	8	404	0.00086	0.00000	0.02104	0.00263	copy.py
get_cube	4286	4286	0.01596	0.00000	0.01957	0.00000	solver.py
find_isola...	513	513	0.00325	0.00001	0.01465	0.00003	solver.py
__eq__	26303	26303	0.00873	0.00000	0.01346	0.00000	solver.py
update...	105	105	0.00168	0.00002	0.00942	0.00009	solver.py
__hash__	59013	59013	0.00692	0.00000	0.00692	0.00000	solver.py
_deepco...	652	652	0.00281	0.00000	0.00527	0.00001	copy.py
<hasattr>	26636	26636	0.00501	0.00000	0.00501	0.00000	~
get_status	50	50	0.00312	0.00006	0.00429	0.00009	solver.py
__nonzer...	14748	14748	0.00400	0.00000	0.00400	0.00000	solver.py
<len>	62716	62717	0.00395	0.00000	0.00395	0.00000	~
solved	57	57	0.00166	0.00003	0.00378	0.00007	solver.py
<genexpr>	17144	17144	0.00362	0.00000	0.00362	0.00000	solver.py
<lambda>	39312	39312	0.00360	0.00000	0.00360	0.00000	solver.py
debug	1447	1447	0.00143	0.00000	0.00348	0.00000	__init__.py
set	1919	1919	0.00106	0.00000	0.00223	0.00000	solver.py
__init__	7	7	0.00019	0.00003	0.00218	0.00031	solver.py
<method...	652	652	0.00210	0.00000	0.00211	0.00000	~
debug	1447	1447	0.00060	0.00000	0.00193	0.00000	__init__.py
load_board	7	7	0.00039	0.00006	0.00193	0.00028	solver.py
isEnable...	1865	1865	0.00105	0.00000	0.00176	0.00000	__init__.py
<module>	1	1	0.00117	0.00117	0.00174	0.00174	__init__.py
all_squares	4682	4682	0.00152	0.00000	0.00173	0.00000	solver.py
_keep_ali...	2100	2100	0.00128	0.00000	0.00169	0.00000	copy.py
<range>	5015	5015	0.00144	0.00000	0.00144	0.00000	~
check	2050	2050	0.00102	0.00000	0.00125	0.00000	solver.py
info	418	418	0.00045	0.00000	0.00114	0.00000	__init__.py
__init__	567	567	0.00096	0.00000	0.00106	0.00000	solver.py

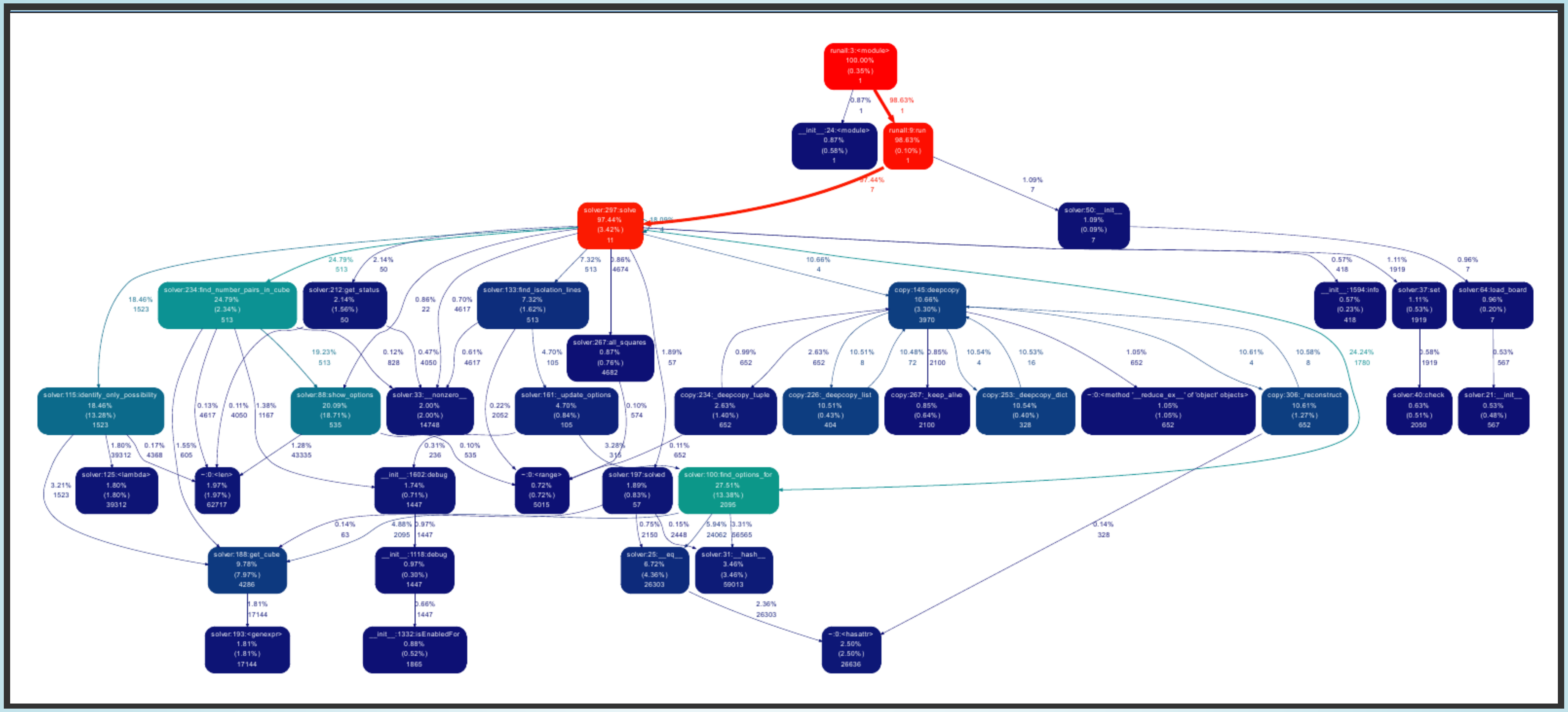
Name	Calls	RCalls	Local	/Call	Cum	/Call	File	Line	Directory

<method 'get' of 'dict' objects>@~:0 [0.001s]

Call Graph

http://code.google.com/p/jrfonseca/wiki/Gprof2Dot

```
$ gprof2dot -f pstats runall.pstats | dot -Tpdf -o output.pdf
```


runall:3:<module> 100.00% (0.35%) 1
__init__:24:<module> 0.87% (0.58%) 1
runall:9:run 98.63% (0.10%) 1
solver:297:solve 97.44% (3.42%) 11
solver:50:__init__ 1.09% (0.09%) 7
solver:234:find_number_pairs_in_cube 24.79% (2.34%) 513
solver:212:get_status 2.14% (1.56%) 50
solver:133:find_isolation_lines 7.32% (1.62%) 513
solver:267:all_squares 0.87% (0.76%) 4682
copy:145:deepcopy 10.66% (3.30%) 3970
__init__:1594:info 0.57% (0.23%) 418
solver:37:set 1.11% (0.53%) 1919
solver:64:load_board 0.96% (0.20%) 7
solver:115:identify_only_possibility 18.46% (13.28%) 1523
solver:88:show_options 20.09% (18.71%) 535
solver:33:__nonzero__ 2.00% (2.00%) 14748
solver:161:_update_options 4.70% (0.84%) 105
copy:234:_deepcopy_tuple 2.63% (1.40%) 652
copy:226:_deepcopy_list 10.51% (0.43%) 404
copy:267:_keep_alive 0.85% (0.64%) 2100
copy:253:_deepcopy_dict 10.54% (0.40%) 328
~:0:<method '__reduce_ex__' of 'object' objects> 1.05% (1.05%) 652
copy:306:_reconstruct 10.61% (1.27%) 652
solver:40:check 0.63% (0.51%) 2050
solver:21:__init__ 0.53% (0.48%) 567
solver:125:<lambda> 1.80% (1.80%) 39312
~:0:<len> 1.97% (1.97%) 62717
__init__:1602:debug 1.74% (0.71%) 1447
~:0:<range> 0.72% (0.72%) 5015
solver:197:solved 1.89% (0.83%) 57
solver:100:find_options_for 27.51% (13.38%) 2095
solver:188:get_cube 9.78% (7.97%) 4286
__init__:1118:debug 0.97% (0.30%) 1447
solver:25:__eq__ 6.72% (4.36%) 26303
solver:31:__hash__ 3.46% (3.46%) 59013
solver:193:<genexpr> 1.81% (1.81%) 17144
__init__:1332:isEnabledFor 0.88% (0.52%) 1865
~:0:<hasattr> 2.50% (2.50%) 26636

PStats

sort_stats

    cumulative, time, call count, primitive call count

print_stats

    with an optional regex filter, or limit

print_callers

    of functions which match a regex

print_callees

    of functions which match a regex

Optimize

Follow the numbers

```
ncalls  tottime  cumtime    filename:lineno(function)
    1    0.001    0.200    runall.py:3(<module>)
    1    0.000    0.198    runall.py:9(run)
 11/7    0.007    0.195    solver.py:297(solve)
 2095    0.027    0.055    solver.py:100(find_options_for)
  513    0.005    0.050    solver.py:234(find_number_pairs_in_cube)
  535    0.037    0.040    solver.py:88(show_options)
 1523    0.027    0.037    solver.py:115(identify_only_possibility)
```

hmm

```
>>> p.print_callees('find_options_for')

   ncalls  tottime  cumtime
    24062    0.008    0.012  solver.py:25(__eq__)
    56565    0.007    0.007  solver.py:31(__hash__)
     2095    0.008    0.010  solver.py:188(get_cube)
```

the code

```
def find_options_for(self, r, c, index):
    other_index = self.cols if index == self.rows else self.rows

    options = index[r][c].options
    options -= set(index[r])
    options -= set(other_index[c])
    options -= set(self.get_cube(r, c, index))
    return options
```

next

```
ncalls  tottime  cumtime    filename:lineno(function)
     1    0.001    0.200    runall.py:3(<module>)
     1    0.000    0.198    runall.py:9(run)
  11/7    0.007    0.195    solver.py:297(solve)
  2095    0.027    0.055    solver.py:100(find_options_for)
   513    0.005    0.050    solver.py:234(find_number_pairs_in_cube)
   535    0.037    0.040    solver.py:88(show_options)
  1523    0.027    0.037    solver.py:115(identify_only_possibility)
```

Aha!

```
>>> p.print_callees('find_number_pairs_in_cube')

ncalls  tottime  cumtime
  1167    0.001    0.003  .../logging/__init__.py:1602(debug)
   828    0.000    0.000  solver.py:33(__nonzero__)
   513    0.036    0.039  solver.py:88(show_options)
   605    0.003    0.003  solver.py:188(get_cube)
  4617    0.000    0.000  {len}
   542    0.000    0.000  {method 'append' of 'list' objects}
```

the code

```
def find_number_pairs_in_cube(self, row_min, col_min):
    ...
    log.debug("Current state of game board:\n%s\n%s",
              self, self.show_options())
    ...
```

the solution

```
class BoardPresenter(object):

    def __init__(self, board):
        self.board = board
    def __str__(self):
        ...

def find_number_pairs_in_cube(self, row_min, col_min):
    ...
    log.debug("Current state of game board:\n%s",
              BoardPresenter(self))
    ...
```

re-timeit

```
$ python -m timeit 'import runall; runall.run()'
10 loops, best of 3: 106 msec per loop
```

repeat

```
ncalls  tottime  cumtime  filename:lineno(function)
     1    0.001    0.164  runall.py:3(<module>)
     1    0.000    0.161  runall.py:9(run)
  11/7    0.007    0.158  solver.py:303(solve)
  2095    0.028    0.057  solver.py:106(find_options_for)
  1523    0.028    0.039  solver.py:121(identify_only_possibility)
3970/4    0.006    0.021  .../copy.py:145(deepcopy)
 652/4    0.002    0.021  .../copy.py:306(_reconstruct)
 328/4    0.001    0.021  .../copy.py:253(_deepcopy_dict)
 404/8    0.001    0.020  .../copy.py:226(_deepcopy_list)
  4286    0.016    0.020  solver.py:194(get_cube)
```

the code

```
def identify_only_possibility(self, r, c):
    target = self.rows[r][c]

    related = self.rows[r], self.cols[c], self.get_cube(r, c, self.rows)
    for related_list in related:
        others_options = set()

        for square in ifilterfalse(lambda s: s is target, related_list):
            others_options |= set(square.options)

        options = target.options - others_options
        if len(options) == 1:
            return options
    return False
```

```python
option_filter = functools.partial(operator.is_, target)

def get_other_options(related_list):
    return set(itertools.chain.from_iterable(
                square.options for square in
                ifilterfalse(option_filter, related_list)))

def get_related_lists():
    yield self.rows[r]
    yield self.cols[c]
    yield self.get_cube(r, c, self.rows)

for related_list in get_related_lists()
    options = target.options - get_other_options(related_list)
    ...
```

and again

```
ncalls  tottime  cumtime  filename:lineno(function)
     1    0.001    0.164  runall.py:3(<module>)
     1    0.000    0.161  runall.py:9(run)
  11/7    0.007    0.158  solver.py:303(solve)
  2095    0.028    0.057  solver.py:106(find_options_for)
  1523    0.028    0.039  solver.py:121(identify_only_possibility)
3970/4    0.006    0.021  .../copy.py:145(deepcopy)
 652/4    0.002    0.021  .../copy.py:306(_reconstruct)
 328/4    0.001    0.021  .../copy.py:253(_deepcopy_dict)
 404/8    0.001    0.020  .../copy.py:226(_deepcopy_list)
  4286    0.016    0.020  solver.py:194(get_cube)
```

deepcopy

```
>>> p.print_callers('deepcopy')

 ncalls  tottime  cumtime
1022/72    0.002    0.020  /usr/lib/python/copy.py:226(_deepcopy_list)
    652    0.001    0.002  /usr/lib/python/copy.py:234(_deepcopy_tuple)
1312/16    0.002    0.021  /usr/lib/python/copy.py:253(_deepcopy_dict)
  980/8    0.002    0.021  /usr/lib/python/copy.py:306(_reconstruct)
      4    0.000    0.021  solver.py:303(solve)
```

the code

```
class Square(object):
    def __init__(self, num):
        self.value = int(num)
        self.options = ...

class SudokuBoard(object):
    def __init__(self, initial_state=None):
        self.rows = self.load_board(initial_state)
        ...

...
new_board = copy.deepcopy(board)
```

the solution

```
class Square(object):
    def __init__(self, num, options=()):
        self.value = int(num)
        self.options = set(options) or ...

class SudokuBoard(object):
    def __init__(self, initial_state=None):
        ...
    def clone(self):
        return type(self)(initial_state=self.get_state())


  ...
  new_board = board.clone()
```

re-timeit

```
$ python -m timeit 'import runall; runall.run()'
10 loops, best of 3: 91.8 msec per loop
```

Outcome

From 135ms to 92ms by making 2 small changes

Common Speedups

- deepcopy
- loops
- dynamic variable lookup
- eager evaluation

Limitations

- timing accuracy
- threads (and processes)
- overhead
- garbage collection

Sampling

```
@contextlib.contextmanager
def profile_section(filename):
    profiler = cProfile.Profile()
    profiler.enable()
    yield
    profiler.disable()
    profiler.dump_stats(filename)
```

```
class ProfiledThread(threading.Thread):

    def run(self):
        with profile_section(threading.get_ident()):
            ...
```

Merging

```
import pstats

merged_stats = pstats.Stats('1.profile',
                            '2.profile',
                            '3.profile')

merged_stats.add('4.profile')
```

- docs.python.org/library/profile.html
- RunSnakeRun
- Gprof2Dot