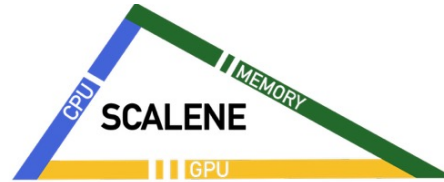


scalene matmult.py – line 27 takes the most time so optimise for speed here, or explore alternative methods rather than nested for loops



► AI optimization options

Time: Python | native | system

Memory: Python | native

Memory timeline: (max: 0.000 MB, growth: 0.0%)



hover over bars to see breakdowns; click on COLUMN HEADERS to sort.

show all | hide all | only display profiled lines ☒

▼/Users/jackwhite/Documents/Uppsala PhD/python-course/day2-bestpractices-1/matmult.py: % of time = 100.0% (1.995s) out of 1.995s.

TIME	MEMORY average	MEMORY peak	MEMORY timeline	MEMORY activity	COPY	LINE PROFILE (click to reset order)
						/Users/jackwhite/Documents/Uppsala PhD/python-course/day2-bestpractices-1/matmult.py
						14 Y.append([random.randint(0,100) for r in range(N+1)])
						22 ⚡ for i in range(len(X)):
						24 ⚡ for j in range(len(Y[0])):
						26 ⚡ for k in range(len(Y)):
						27 ⚡ result[i][j] += X[i][k] * Y[k][j]
						29 ⚡ for r in result:
						30 ⚡ print(r)

time (Python) 84.7%

## euler72.py using cProfile – it was too fast to use scalene

```
day2-bestpractices-1 — zsh — 131x42
(base) jackwhite@Jacks-MacBook-Pro-2 day2-bestpractices-1 % python -m cProfile euler72.py
[
    226 function calls in 0.001 seconds
]

Ordered by: cumulative time

ncalls  tottime  percall  cumtime  percall filename:lineno(function)
    1    0.000    0.000    0.001    0.001 {built-in method builtins.exec}
    1    0.000    0.000    0.001    0.001 euler72.py:1(<module>)
    1    0.000    0.000    0.001    0.001 <frozen importlib._bootstrap>:1165(_find_and_load)
    1    0.000    0.000    0.001    0.001 <frozen importlib._bootstrap>:1120(_find_and_load_unlocked)
    1    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:666(_load_unlocked)
    1    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:566(module_from_spec)
    3    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:233(_call_with_frames_removed)
    1    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap_external>:1231(create_module)
    1    0.000    0.000    0.000    0.000 {built-in method _imp.create_dynamic}
    1    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:1054(_find_spec)
    1    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap_external>:1496(find_spec)
    1    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap_external>:1464(_get_spec)
    4    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap_external>:1604(find_spec)
   16    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap_external>:126(_path_join)
    5    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap_external>:1421(_path_importer_cache)
    5    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap_external>:140(_path_stat)
    5    0.000    0.000    0.000    0.000 {built-in method posix.stat}
    1    0.000    0.000    0.000    0.000 {built-in method posix.getcwd}
    1    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:169(__enter__)
    1    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap_external>:1599(_get_spec)
    1    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:493(_init_module_attrs)
    1    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap_external>:159(_path_isfile)
   16    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap_external>:128(<listcomp>)
    1    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap_external>:150(_path_is_mode_type)
    1    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:179(_get_module_lock)
    1    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap_external>:778(spec_from_file_location)
    1    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:920(find_spec)
    1    0.000    0.000    0.000    0.000 __init__.py:89(find_spec)
    1    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:173(__exit__)
   19    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:244(_verbose_message)
    7    0.000    0.000    0.000    0.000 {built-in method builtins.getattr}
    1    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap_external>:1239(exec_module)
    1    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:392(cached)
    1    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:100(acquire)
    1    0.000    0.000    0.000    0.000 {built-in method _imp.find_frozen}
   32    0.000    0.000    0.000    0.000 {method 'rstrip' of 'str' objects}
```

```
day2-bestpractices-1 — zsh — 131x60
(base) jackwhite@Jacks-MacBook-Pro-2 day2-bestpractices-1 % kernprof -l -v euler72.py
30397485.0
Wrote profile results to euler72.py.lprof
Timer unit: 1e-06 s

Total time: 0.00151 s
File: euler72.py
Function: gen_primes at line 4

Line #      Hits          Time    Per Hit   % Time  Line Contents
=====
   4
   5          1          1.0        1.0     0.1    @profile
   6          1          0.0        0.0     0.0    def gen_primes(n):
   7          1          0.0        0.0     0.0        l = range(2,n)
   8          999        71.0        0.1     4.7        primes = []
   9          998        65.0        0.1     4.3        for j in range(0,len(l)):
  10          2968       242.0        0.1    16.0            p = True
  11          2967       460.0        0.2    30.5            for d in primes:
  12          167        12.0        0.1     0.8                if(d > sqrt(l[j])):
  13          2800       428.0        0.2    28.3                    break
  14          830        42.0        0.1     2.8                    if(l[j] % d == 0):
  15          830        65.0        0.1     4.3                        p = False
  16          998       100.0        0.1     6.6                        break;
  17          168        24.0        0.1     1.6                if(p):
  18                                primes.append(l[j])
  19          1          0.0        0.0     0.0            return primes

Total time: 0.049632 s
File: euler72.py
Function: factorize at line 21

Line #      Hits          Time    Per Hit   % Time  Line Contents
=====
  21
  22          1          1.0        1.0     0.1    @profile
  23          9999       650.0        0.1     1.3    def factorize(n,primes):
  24          9999       822.0        0.1     1.7        factors = []
  25          96347      8612.0        0.1    17.4        init_n = n
  26          118736    16168.0        0.1    32.6        for p in primes:
  27          22389     2466.0        0.1     5.0            while(n%p == 0):
  28          22389     3158.0        0.1     6.4                n = n/p
  29          96347    13426.0        0.1    27.1                factors.append(p)
  30          9999      897.0        0.1     1.8                if(p > sqrt(n)):
  31          9999     1135.0        0.1     2.3                    break
  32          9596     1416.0        0.1     2.9                if(n > 1):
  33          9999      882.0        0.1     1.8                    factors.append(n)
                                return factors

Total time: 0.116235 s
File: euler72.py
Function: fast_phi at line 50

Line #      Hits          Time    Per Hit   % Time  Line Contents
=====
  50
  51          1          1.0        1.0     0.1    @profile
  52          9999    102411.0       10.2    88.1    def fast_phi(n,primes):
  53          9999     1240.0        0.1     1.1        factors = factorize(n,primes)
  54          31985     4348.0        0.1     3.7        phi = factors[0]-1
  55          21986     3397.0        0.2     2.9        for i in range(1,len(factors)):
                                if(factors[i] == factors[i-1]):
```

euler72.py using  
line\_profiler – line 52,  
i.e. calling the factorize  
function is time-costly.