Profiling protokol

Petr Salaba (xsalab00)

Profilování kódu odhalilo, že nejvíce času zabere čtení vstupu a psaní na výstup, ale také je důležité nezanedbat funkce, které se při vykonávání programu provedou mnohokrát, v našel případě sčítání a umocňování.

Při optimalizaci kódu by se mělo zaměřit na vysoce využívané funkce a funkce u kterých je v profilování jasné, že jsou napsány neefektivně.

Testovali jsme počítání výběrové směrodatné odchylky, výsledky běhu funkcí byly zaznamenány do tabulek.

```
34.18072881350277
         69 function calls in 0.000 seconds

   Ordered by: standard name

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
        1    1.9µs    1.9µs  202.7µs  202.7µs <string>:1(<module>)
        2    5.0µs    2.5µs    6.7µs    3.3µs cp1252.py:22(decode)
       18    1.5µs    0.1µs    1.5µs    0.1µs mathlib.py:132(add)
        2    0.2µs    0.1µs    0.2µs    0.1µs mathlib.py:145(subtract)
        1    0.4µs    0.4µs    0.4µs    0.4µs mathlib.py:158(multiply)
        2    0.6µs    0.3µs    0.6µs    0.3µs mathlib.py:171(divide)
       11    4.6µs    0.4µs    4.6µs    0.4µs mathlib.py:189(power)
        1    3.9µs    3.9µs    3.9µs    3.9µs mathlib.py:207(root)
        1    6.4µs    6.4µs   26.3µs   26.3µs stddev.py:15(calculate_standard_deviation)
       10    2.5µs    0.2µs    6.8µs    0.7µs stddev.py:19(<lambda>)
        1   44.6µs   44.6µs   79.1µs   79.1µs stddev.py:25(read_and_calculate)
        2    1.7µs    0.8µs    1.7µs    0.8µs {built-in method _codecs.charmap_decode}
        2    6.1µs    3.0µs   14.4µs    7.2µs {built-in method _functools.reduce}
        1   20.5µs   20.5µs  223.2µs  223.2µs {built-in method builtins.exec}
        2    0.3µs    0.1µs    0.3µs    0.1µs {built-in method builtins.len}
        1  121.7µs  121.7µs  121.7µs  121.7µs {built-in method builtins.print}
        1    0.2µs    0.2µs    0.2µs    0.2µs {method 'disable' of '_lsprof.Profiler' objects}
       10    1.3µs    0.1µs    1.3µs    0.1µs {method 'split' of 'str' objects}
```

Figure 1: Výstup profilování pro 10 vstupních čísel

```
15.774048499374164
         519 function calls in 0.000 seconds

   Ordered by: standard name

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
        1    2.3µs    2.3µs  312.1µs  312.1µs <string>:1(<module>)
        2    5.0µs    2.5µs    7.4µs    3.7µs cp1252.py:22(decode)
      198   11.0µs    0.1µs   11.0µs    0.1µs mathlib.py:132(add)
        2    0.3µs    0.1µs    0.3µs    0.1µs mathlib.py:145(subtract)
        1    0.5µs    0.5µs    0.5µs    0.5µs mathlib.py:158(multiply)
        2    0.6µs    0.3µs    0.6µs    0.3µs mathlib.py:171(divide)
      101   27.5µs    0.3µs   27.5µs    0.3µs mathlib.py:189(power)
        1    5.4µs    5.4µs    5.4µs    5.4µs mathlib.py:207(root)
        1    5.6µs    5.6µs  104.1µs  104.1µs stddev.py:15(calculate_standard_deviation)
      100   19.0µs    0.2µs   46.2µs    0.5µs stddev.py:19(<lambda>)
        1  101.7µs  101.7µs  223.5µs  223.5µs stddev.py:25(read_and_calculate)
        2    2.4µs    1.2µs    2.4µs    1.2µs {built-in method _codecs.charmap_decode}
        2   34.1µs   17.0µs   91.3µs   45.6µs {built-in method _functools.reduce}
        1   35.3µs   35.3µs  347.4µs  347.4µs {built-in method builtins.exec}
        2    0.2µs    0.1µs    0.2µs    0.1µs {built-in method builtins.len}
        1   86.3µs   86.3µs   86.3µs   86.3µs {built-in method builtins.print}
        1    0.2µs    0.2µs    0.2µs    0.2µs {method 'disable' of '_lsprof.Profiler' objects}
      100   10.2µs    0.1µs   10.2µs    0.1µs {method 'split' of 'str' objects}
```

Figure 2: Výstup profilování pro 100 vstupních čísel

```
86.74121217372942
         5019 function calls in 0.002 seconds

   Ordered by: standard name

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
        1    6.7µs    6.7µs 1803.0µs 1803.0µs <string>:1(<module>)
        2    5.6µs    2.8µs   17.2µs    8.6µs cp1252.py:22(decode)
     1998  126.8µs    0.1µs  126.8µs    0.1µs mathlib.py:132(add)
        2    0.2µs    0.1µs    0.2µs    0.1µs mathlib.py:145(subtract)
        1    0.4µs    0.4µs    0.4µs    0.4µs mathlib.py:158(multiply)
        2    0.5µs    0.2µs    0.5µs    0.2µs mathlib.py:171(divide)
     1001  262.9µs    0.3µs  262.9µs    0.3µs mathlib.py:189(power)
        1    5.7µs    5.7µs    5.7µs    5.7µs mathlib.py:207(root)
        1   11.6µs   11.6µs  888.2µs  888.2µs stddev.py:15(calculate_standard_deviation)
     1000  164.8µs    0.2µs  427.3µs    0.4µs stddev.py:19(<lambda>)
        1  646.3µs  646.3µs 1647.3µs 1647.3µs stddev.py:25(read_and_calculate)
        2   11.6µs    5.8µs   11.6µs    5.8µs {built-in method _codecs.charmap_decode}
        2  315.3µs  157.6µs  869.4µs  434.7µs {built-in method _functools.reduce}
        1   30.2µs   30.2µs 1833.2µs 1833.2µs {built-in method builtins.exec}
        2    0.1µs    0.0µs    0.1µs    0.0µs {built-in method builtins.len}
        1  149.0µs  149.0µs  149.0µs  149.0µs {built-in method builtins.print}
        1    0.1µs    0.1µs    0.1µs    0.1µs {method 'disable' of '_lsprof.Profiler' objects}
     1000   95.5µs    0.1µs   95.5µs    0.1µs {method 'split' of 'str' objects}
```

Figure 3: Výstup profilování pro 1000 vstupních čísel