Pracownia Jezyków Skryptowych

PYTHON01 - Raport

Jakub Dziurka

2026

1 Cel ćwiczenia

Celem ćwiczenia było porównanie: przetwarzania sekwencyjnego, wieloprocesowego (multiprocessing), wielowatkowego (threading), w dwóch środowiskach: standardowy Python z GIL, Python 3.13 free-threaded (GIL wyłaczony).

2 Wstep

Program generuje liste N = 100 000 rekordów w postaci słowników:

```
{ "id": i, "text": "losowy tekst..." }
```

Dla każdego rekordu wykonywane sa operacje:

1. zliczenie liczby słów,
2. zliczenie liczby unikalnych liter,
3. obliczenie score" – liczby wystapień liter p, y, t, h, o, n.

Przykład dla rekordu:

```
"python is great"
```

1. liczba słów: 3
2. unikalne litery: p, y, t, h, o, n, i, s, g, r, e, a - 12
3. score: 1

Wyniki zapisywane sa w strukturze:

```
results[id] = {
    "word_count": ...,
    "unique_letters": ...,
    "score": ...,
}
```

3 Wyniki

Wynik dla skryptu z właczonym GIL:

```
GIL = True
Sequential time: 1.413 s
Multiprocessing time: 0.927 s | Czy poprawny?: True
Threading (no lock) time: 1.397 s | Czy poprawny?: True
Threading (with lock) time: 1.421 s | Czy poprawny?: True
```

Wynik dla skryptu z wyłaczonym GIL:

```
GIL = False
Sequential time: 1.713 s
Multiprocessing time: 0.913 s | Czy poprawny?: True
Threading (no lock) time: 0.949 s | Czy poprawny?: True
Threading (with lock) time: 1.166 s | Czy poprawny?: True
```

4 Podsumowanie

W wersji wielowatkowej bez synchronizacji wyniki w cześci testów okazały sie poprawne, nawet przy wyłaczonym GIL, mimo że w teorii w wynikach wersji wielowatkowej bez żadnych blokad moga pojawiać sie nieścisłości.

Czasy wykonania obliczeń sekwencyjnych były bardzo podobne niezależnie od tego, czy program uruchamiany był na Pythonie z GIL, czy na wersji free-threaded. Wynika to z faktu, że w tym wariancie obliczenia wykonywane sa w jednym watku, wiec mechanizm GIL nie ma wpływu na ich przebieg.

W przypadku obliczeń wieloprocesowych zaobserwowano wyraźne przyspieszenie w porównaniu do wersji sekwencyjnej. Program wykorzystywał jednocześnie wiele rdzeni procesora, co było widoczne w narzedziach monitorujacych. Różnice pomiedzy Pythonem z GIL i bez GIL były niewielkie, ponieważ każdy proces działa niezależnie i posiada własny interpreter.

Obliczenia wielowatkowe uruchomione na standardowym Pythonie z GIL nie przyniosły poprawy wydajności, a w niektórych przypadkach były nawet nieco wolniejsze od wersji sekwencyjnej. GIL powoduje, że tylko jeden watek może w danej chwili wykonywać kod Pythona, co eliminuje możliwość równoległego wykonywania obliczeń obciażajacych procesor.

Po wyłaczeniu GIL w Pythonie 3.13 watki mogły wykonywać sie równolegle na różnych rdzeniach procesora, co przełożyło sie na zauważalne skrócenie czasu wykonania programu. Wersja wielowatkowa bez GIL była jednak mniej wydajna niż wariant wieloprocesowy, ponieważ interpreter musi dodatkowo synchronizować dostep do obiektów za pomoca wielu wewnetrznych, drobnoziarnistych blokad.