Multithreading czy Multiprocessing

Projekt zaliczeniowy

Wprowadzenie

Multithreading i multiprocessing to dwie techniki pozwalające na jednoczesne wykonywanie wielu zadań. Porównanie czasu trwania tych samych zadań wykonywanych z wykorzystaniem obu technik może prowadzić do ciekawych wniosków.

Opis projektu

Projekt składa się z dwóch części:

1. programu, który sprawdzi czas wykonania określonego zadania na jednym wątku, wielu wątkach i wielu procesach;
2. raportu wygenerowanego przez ten program.

Program po uruchomieniu ma pięciokrotnie obliczyć czas wykonania opisanego niżej **zadania** w następujących wariantach wątków i procesów:

- jednym wątku;
- czterech wątkach;
- czterech procesach;
- procesach, których liczba zależy od liczby dostępnych procesorów w systemie.

Zadanie polega na obliczeniu:

$$f(n)=\sum_{i=1}^{n}((n-i)\cdot i)$$

Notatki:

.....

.....

dla każdej z następujących liczb:

15972490, 80247910, 92031257, 75940266,
97986012, 87599664, 75231321, 11138524,
68870499, 11872796, 79132533, 40649382,
63886074, 53146293, 36914087, 62770938.

Po zebraniu danych program powinien zapisać w pliku `report.html` raport w formacie HTML, w którym zostaną umieszczone:

1. podstawowe informacje o środowisku testowym:
    1. wersja języka Python (funkcja `python_version` z modułu `platform`),
    2. nazwa interpretera (funkcja `python_implementation` z modułu `platform`),
    3. wersja interpretera (zmienna `version` z modułu `sys`),
    4. system operacyjny (funkcja `system` z modułu `platform`),
    5. wersja systemu operacyjnego (funkcja `release` z modułu `platform`),
    6. procesor (funkcja `processor` z modułu `platform`),
    7. liczba procesorów (funkcja `cpu_count` z modułu `os`),
2. czasy wykonań każdego z zadań na poszczególnych wariantach wątków i procesów w formie tabeli (czasy zaokrąglone do trzech miejsc po przecinku),
3. mediany czasów wykonań z poszczególnych wariantów wątków i procesów w formie tabeli (czasy zaokrąglone do trzech miejsc po przecinku),
4. imię i nazwisko autora programu testującego.

Notatki:

.....

.....

Wskazówki

- Do uzyskania informacji o środowisku użyj modułów `os`, `platform`, `sys`.
- W celu zaokrąglenia liczb do trzech miejsc po przecinku i umieszczenia ich w dokumencie możesz użyć formatowania napisów (np. str.format() lub f-string) z odpowiednim specyfikatorem konwersji (format specifier).
- Moduł `timeit`[1] pozwala zmierzyć czas wykonywania fragmentu kodu. Funkcja o nazwie `timeit` zwraca czas w sekundach (typ danych `float`).
- Dokument HTML możesz wygenerować przy pomocy odpowiedniego formatowania i łączenia napisów ze sobą, nie potrzebujesz zewnętrznych bibliotek.
- Zapis danych do pliku zrealizuj z użyciem konstrukcji `with`[2].
- Pamiętaj o użyciu idiomu (semi-standard idiom)[3] `if __name__ == '__main__'`[4], by skrypt był możliwy do zaimportowania i wykonania.
- Wzorzec projektowy builder[5] możesz wykorzystać do tworzenia pliku wynikowego.
- Zgodność kodu źródłowego z dokumentem PEP8 możesz sprawdzić za pomocą programu `pycodestyle` dostępnego w repozytorium PyPi[6].
- Pamiętaj, że katalog wirtualnego środowiska (najczęściej ma nazwę `venv`) nie jest przeznaczony do udostępniania.

---

1 timeit — Measure execution time of small code snippets: https://docs.python.org/3/library/timeit.html
2 The Python Tutorial, Reading and Writing Files: https://docs.python.org/3.9/tutorial/inputoutput.html#reading-and-writing-files
3 PEP 299 -- Special __main__() function in modules, Motivation: https://www.python.org/dev/peps/pep-0299/#motivation
4 What does if __name__ == "__main__": do?: https://stackoverflow.com/questions/419163/what-does-if-name-main-do
5 Builder, Refactoring Guru: https://refactoring.guru/design-patterns/builder
6 Instrukcja użycia pycodestyle: https://pycodestyle.pycqa.org/en/latest/intro.html

Notatki:

.....

.....

Przykładowe raporty

Raport 1 (z maszyny rzeczywistej):

Multithreading/Multiprocessing benchmark results

Execution environment

Python version: 3.8.5
Interpreter: CPython
Interpreter version: 3.8.5 (default, Jul 28 2020, 12:59:40) [GCC 9.3.0]
Operating system: Linux
Operating system version: 5.4.0-48-generic
Processor: x86_64
CPUs: 8

Test results

The following table shows detailed test results:

Execution:	1 thread (s)	4 threads (s)	4 processes (s)	processes based on number of CPUs (s)
1	13.774	13.678	3.942	4.038
2	13.366	13.735	4.048	3.760
3	13.579	13.636	4.006	3.902
4	13.491	13.654	4.144	3.786
5	13.573	13.753	4.162	4.266

Summary

The following table shows the median of all results:

Execution:	1 thread (s)	4 threads (s)	4 processes (s)	processes based on number of CPUs (s)
Median:	13.573	13.678	4.048	3.902

App author: Jan Kowalski

Notatki:

.....

.....

Raport 2 (z maszyny wirtualnej):

Multithreading/Multiprocessing benchmark results

Execution environment

Python version: 3.7.5
Interpreter: CPython
Interpreter version: 3.7.5 Stackless 3.7 (tags/v3.7.5-slp:f7925f2a02, Oct 20 2019, 15:28:53) [MSC v.1916 64 bit (AMD64)]
Operating system: Windows
Operating system version: 10
Processor: Intel64 Family 6 Model 58 Stepping 9, GenuineIntel
CPUs: 4

Test results

The following table shows detailed test results:

Execution:	1 thread (s)	4 threads (s)	4 processes (s)	processes based on number of CPUs (s)
1	44.850	45.656	14.006	12.946
2	45.160	45.785	13.057	13.269
3	45.637	46.578	13.515	13.091
4	45.928	44.839	13.069	14.712
5	46.119	48.441	15.113	14.515

Summary

The following table shows the median of all results:

Execution:	1 thread (s)	4 threads (s)	4 processes (s)	processes based on number of CPUs (s)
Median:	45.637	45.785	13.515	13.269

App author: Jan Kowalski

Notatki:

.....

.....

Punktacja

Warunki wstępne do uzyskania punktów:

- projekt jest wynikiem samodzielnej pracy,
- w projekcie używaj tylko modułów standardowych – nie używaj modułów pochodzących spoza domyślnej instalacji Pythona,
- wygenerowany raport jest zapisany w formacie HTML.

Obowiązująca punktacja (w nawiasach podano możliwe do zdobycia liczby punktów):

- raport zawiera informacje o środowisku testowym:
  - (0pkt albo 1pkt) program poprawnie raportuje wersję języka Python,
  - (0pkt albo 1pkt) program poprawnie raportuje używany interpreter,
  - (0pkt albo 1pkt) program poprawnie raportuje wersję interpretera,
  - (0pkt albo 1pkt) program poprawnie raportuje nazwę systemu operacyjnego,
  - (0pkt albo 1pkt) program poprawnie raportuje wersję systemu operacyjnego,
  - (0pkt albo 1pkt) program poprawnie raportuje informacje o procesorze,
  - (0pkt albo 1pkt) program poprawnie raportuje liczbę dostępnych procesorów,
- raport zawiera informacje o poszczególnych wykonaniach:
  - (0pkt albo 3pkt) program poprawnie raportuje czasy obliczeń na jednym wątku,
  - (0pkt albo 3pkt) program poprawnie raportuje czasy obliczeń na czterech wątkach,
  - (0pkt albo 3pkt) program poprawnie raportuje czasy obliczeń na czterech procesach,

Notatki:

.....

.....

    - (0pkt albo 3pkt) program poprawnie raportuje czasy obliczeń na liczbie procesów zależnej od liczby dostępnych procesorów,
- raport zawiera podsumowanie:
    - (0pkt albo 1pkt) program poprawnie raportuje medianę z czasów obliczeń na jednym wątku,
    - (0pkt albo 1pkt) program poprawnie raportuje medianę z czasów obliczeń na czterech wątkach,
    - (0pkt albo 1pkt) program poprawnie raportuje medianę z czasów obliczeń na czterech procesach,
    - (0pkt albo 1pkt) program poprawnie raportuje medianę z czasów obliczeń na liczbie procesów zależnej od liczby dostępnych procesorów,
- (0pkt albo 1pkt) raport zawiera informacje o autorze programu,
- (0pkt albo 2pkt) w projekcie wykorzystywany jest idiom `if __name__ == '__main__'`,
- (0pkt albo 1pkt) kod programu jest zgodny z dokumentem PEP8 (program `pycodestyle` nie wypisuje komunikatów o błędach),
- przesłany raport został wygenerowany na interpreterze:
    - (0pkt albo 3pkt) CPython,
    - (0pkt albo 3pkt) PyPy, link: https://www.pypy.org/
    - (0pkt albo 3pkt) Stackless Python, link: http://www.stackless.com/

Notatki:

.....

.....