5. Taming False Alarms from a Domain-Unaware C Analyzer by a Bayesian Statistical Post Analysis / J.Yungbum, S. Jaeho, Y. Kwangkeun, K. Jaehwang. – Seoul, 2005. – 127 с.

6. Csallner C. DSD-Crasher: A hybrid analysis tool for bug finding [Електронний ресурс] / C. Csallner, Y. Smaragdakis, X. Tao. – 2008. – Режим доступу до ресурсу: https://dl.acm.org/doi/10.1145/1348250.1348254

7. Alikhashashneh E. Using Machine Learning Techniques to Classifyand Predict Static Code Analysis Tool Warnings / E. Alikhashashneh, R. Rajeev. – Indianapolis, 2019. – 52 с.

8. Kathleen G. Feature Set Selection for Improved Classification of Static Analysis Alerts / Goeschel Kathleen., 2019. – 120 с.

Світенко Г.М.

студент,

Харківський національний університет радіоелектроніки

ПАРАЛЕЛІЗМ У PYTHON

Паралельні обчислення – це форма обчислень, в якої два або більше обчислення виконуються за один і той же проміжок часу. Паралельні обчислення ґрунтуються на тому, що великі задачі можна розділити на кілька менших завдань, кожне з яких можна вирішити незалежно від інших. Застосування паралельних обчислень покликано зменшити загальний час виконання програми.

Паралелізм є окремим випадком організації паралельних обчислень, коли в один момент часу розв’язуються декілька задач. Для виконання паралельних програм необхідним є комп’ютер з багатоядерним процесором або з кількома процесорами.

Паралельні обчислення у Python [1; 2] можуть бути реалізовані з використанням трьох модулів:

– multiprocessing – для виконання операції, які є ресурсномісткими;

– asyncio – для виконання операції, що пов’язані з введенням та виведенням даних;

– threading – універсальний модуль, що підходить для усіх інших варіантів операцій.

У доповіді розглядається спосіб розпаралелювання будь-якої типової логіки за допомогою модуля multiprocessing.

Модуль multiprocessing було додано до Python у версії 2.6. Його авторами є Джессі Ноллер та Ричард Одкерк. Цей модуль надає методи для створення процесів та взаємодії із ними за допомогою API. Multiprocessing допускає локальну та віддалену спільну обробку даних та надає можливість обходити обмеження Global Interpreter Lock (GIL), що означає можливість одночасно використовувати декілька процесорів комп'ютеру.

Програми з використанням multiprocessing можливо використовувати на комп'ютерах з операційними системами Windows та Unix.

GIL – це механізм інтерпретатору мови програмування Python, для синхронізації виконання потоків, таким чином, що одночасно може виконуватися лише один власний потік, навіть якщо програма виконується на багатоядерному процесорі. Цей механізм використовується для уникнення конкурентного доступу до спільних структур даних.

Дослідження прискорення виконання програми завдяки використанню паралельних обчислень проводилися на прикладі обчислення числа Пі за методом Монте-Карло. Ідея знаходження числа Пі за методом Монте-Карло полягає у переборі точок певної площини на відповідність заданим критеріям.

Загальний алгоритм виглядає наступним чином:

визначимо квадрат із стороною довжиною 2R;

впишемо до квадрату коло з радіусом R;

почнемо випадковим чином ставити крапки всередині квадрату;

знаючи, що геометрична ймовірність того, що крапка потрапить до середини кола дорівнює відношенню площі кола до площі квадрата, отримаємо формулу для знаходження числа Пі – $\pi \approx 4M / N$, де M – кількість точок, що належить колу, N – загальна кількість точок.

Досліджувався час виконання програм, які були написані на мовах програмування Python та C++, за вказаним алгоритмом в залежності від кількості численних експериментів. Результати досліджень наведено у таблиці 1.

На теперішній час Python займає другу позицію у глобальному рейтингу мов програмування TIOBE, та є лідером серед мов програмування для обробки даних, машинного навчання та розробки комп'ютерного інтелекту. Простота сприйняття та використання зробила проект нідерландського програміста Гвідо ван Россума всесвітньо

відомим та дуже популярним серед розробників та науковців. Але використання паралелізму у Python, у порівнянні з компільованими мовами програмування, на кшталт C++ – не має сенсу. Адже компільовані мови виконують паралельні обчислення значно швидше.

Таблиця 1

Результати роботи програм на Python та C++

Мова	Кількість точок	Час послідовного виконання, с	Час паралельного виконання, с	Прискорення
Python	100	$7 \cdot 10^{-5}$	0.35	$0.2 \cdot 10^{-3}$
	1000	$6 \cdot 10^{-4}$	0.34	$1.8 \cdot 10^{-3}$
	10 000	$6.5 \cdot 10^{-3}$	0.35	$18.5 \cdot 10^{-3}$
	100 000	$63 \cdot 10^{-3}$	0.36	0.18
	1 000 000	0.63	0.53	1.8
	10 000 000	6.5	2	3.25
	100 000 000	66.5	23	2.89
C++	100	$2.63 \cdot 10^{-5}$	$3.2 \cdot 10^{-3}$	$8.2 \cdot 10^{-3}$
	1000	$0.16 \cdot 10^{-3}$	$8.31 \cdot 10^{-5}$	1.92
	10 000	$1.7 \cdot 10^{-3}$	$0.7 \cdot 10^{-3}$	2.42
	100 000	$17 \cdot 10^{-3}$	$8.7 \cdot 10^{-3}$	1.95
	1 000 000	0.22	$59 \cdot 10^{-3}$	3.72
	10 000 000	1.9	0.38	5
	100 000 000	16.7	3.6	4.63

В той же час, серед інтерпретованих мов програмування, Python є однією з небагатьох мов, що підтримують паралелізм. Отже, простота використання та чисельна бібліотека наукових і математичних модулів Python, допомагає цій мові займати лідируючі позиції у сфері обробки даних, де можливість виконувати паралельні обчислення на великих об’ємах інформації є безсумнівним плюсом.

Список використаних джерел:

1. Python.org. URL: https://www.python.org/ (дата звернення: 19.11.2020).

2. Python Documentation. URL: https://docs.python.org/ (дата звернення: 24.11.2020).