Сіциліцин Ю.О.
доктор філософії
старший викладач кафедри інформатики і кібернетики
Мелітопольський державний педагогічний університет
імені Богдана Хмельницького
Пересунько Є.С.
здобувач вищої освіти
Таврійський державний агротехнологічний університет
імені Дмитра Моторного

ВІЗУАЛІЗАЦІЯ ПАРАЛЕЛЬНИХ ОБЧИСЛЕНЬ

Анотація. *У статті були проаналізовані бібліотеки візуалізації мови Python та вибрана бібліотека Seaborn як оптимальний інструмент для розробки застосунку з візуалізації паралельних ообчислень. Був описаний застосунок, який інтегрує Seaborn з Tkinter, який дозволяє студентам створювати та аналізувати паралельні програми у зручному інтерфейсі. Робота відкриває нові можливості для подальших досліджень у галузі візуалізації та навчання паралельного програмування.*

Ключові слова: *Візуалазація навчання, паралельні обчислення, Seaborn, Tkinter.*

Sitsylitsyn Y.O., Peresun'ko Y. S. Y.S. Visualization of parallel computing. *The article analyzed the visualization libraries of the Python language and selected the Seaborn library as the optimal tool for developing a parallel computing visualization application. An application was described that integrates Seaborn with Tkinter, allowing students to create and analyze parallel programs in a user-friendly interface. The work opens up new opportunities for further research in the field of visualization and teaching parallel programming.*

Key words: *Learning visualization, parallel computing, Seaborn, Tkinter.*

Актуальність дослідження. У наш час, коли обробка великих обсягів даних та швидка адаптація оперування такими обсягами даних набувають критичного значення, паралельні обчислення стають все більш важливими. Це обумовлено стрімким розвитком технологій та

ростом потреб обробки даних у реальному часі. Однак початківці у цій області часто мають труднощі з розумінням паралельних парадигм, особливо після вивчення послідовного програмування. Різниця між послідовним та паралельним програмуванням, де виконання декількох процесів одночасно призводить до нового рівня складності, часто викликає непорозуміння й помилки при реалізації на практиці починаючими програмістами. Розуміння ключових концепцій паралельних обчислень, таких як синхронізація, управління ресурсами та масштабування, є критичним для готовності студентів до роботи з паралельним програмуванням.

Таким чином, перехід від послідовного до паралельного мислення вимагає не лише зміни парадигми, але й глибшого розуміння нових викликів та можливостей, які відкриваються перед розробниками програмного забезпечення. Використання візуальних навчальних застосунків, які демонструють роботу паралельних програм у вигляді схем, може стати важливим інструментом на лабораторних заняттях. Ці застосунки перетворюють абстрактні концепції на зрозумілі візуальні образи, що допомагає студентам краще їх розуміти. Вони дають змогу спостерігати за виконанням алгоритмів у реальному часі. Ще більш ефективним буде процес опанування алгоритмами паралельних обчислень за умови використання анімації в реальному часі роботи застосунку.

Метою статті є вибір бібліотеки мови Python, яка найкраще підходить для розробки застосунку візуалізації роботи паралельної програми у реальному часі та розробка базового інтерфейсу застосунку.

Виклад основного матеріалу. Використання візуалізації при навчанні програмуванню знайшло відображення багатьох роботах. Науковці Di Rocco та Ferraro Petrillo розробили систему, призначену для спрощення навчання MapReduce, популярної парадигми розподіленого програмування, за допомогою програмної візуалізації [4]. Дослідники Kurniawati та Kusumaningsih розробили онлайн-репетитор python і lowgarithm - інструменти візуалізації для викладання та вивчення програмування [5].

Розглянемо ключові концепції паралельних обчислень, які за допомогою анімації роботи програми у реальному часі, можуть бути краще зрозумілі та засвоєні студентами. Список концепцій які потрібно вивчити студентам візьмемо з одного з ключових документів який регламентує загальний огляд тем, які слід включити в навчання

студентам спеціальності «Комп'ютерні науки» є «Computer Science Curricula 2023» [3] та статті яка розглядає моделювання навчального контенту для інженерів-програмістів з паралельних обчислень [7].

Ось декілька ключових концепцій, які на нашу думку можуть бути відображені через анімацію:

– Паралельність vs Конкурентність. Анімація може ілюструвати відмінності між паралельним виконанням (одночасне виконання на різних фізичних ядрах) та конкурентністю (одночасне виконання в одному ядрі).
– Синхронізація потоків. Візуалізація механізмів синхронізації, таких як мютекси та семафори, та їх вплив на взаємодію між потоками.
– Взаємодія та умови гонки. Через анімацію можна показати, як умови гонки виникають при неконтрольованому доступі до спільних ресурсів, та як їх можна уникнути.
– Балансування навантаження та розподіл роботи. Анімація може показати, як завдання розподіляються між різними потоками або процесорами, та як це впливає на загальну продуктивність.
– Алгоритми паралельних обчислень. Наприклад, можна візуалізувати паралельне сортування або алгоритми пошуку, показуючи одночасну обробку даних.
– Ідентифікація та налагодження проблем. Анімація може допомогти в ідентифікації складних проблем, таких як взаємоблокування або неефективне використання ресурсів.
– Розподілені системи та кластери. Візуалізація взаємодії між вузлами в мережі, показуючи, як дані передаються та обробляються у розподілених системах.

У галузі програмування існує різноманіття мов, що пропонують різні бібліотеки для створення паралельних програм. Серед них варто зазначити Fortran, C, C++, Python та Java, кожна з яких має свої особливості у синтаксисі та можливостях. Незважаючи на різноманіття, їх спільною метою є ефективна реалізація паралельних алгоритмів, але підходи та інструменти для досягнення цієї мети можуть значно відрізнятися.

Згідно з дослідженням [7], для успішної розробки паралельних програм на будь-якій з цих мов, студентам потрібно не лише досконало знати обрану мову програмування, але й розуміти специфіку її бібліотек для паралельної обробки. Це ставить перед студентами великий виклик, адже вони мають одночасно опанувати як основи мови, так і складніші

аспекти паралельного програмування. Враховуючи ці виклики, ми пропонуємо розробити застосунок, який буде використовувати спрощену формальну мову для написання паралельних програм [2]. Такий підхід має на меті не лише спростити сам процес написання паралельних програм, але й зробити його більш доступним та зрозумілим для студентів, які лише починають свій шлях у світі паралельних обчислень. Використання візуалізації значно поліпшить процес навчання, роблячи його більш захоплюючим та інтерактивним, а також дозволить студентам розвивати практичні навички в паралельному програмуванні, не занурюючись вглиб складностей мов програмування.

Враховуючи усі розглянуті фактори та потреби, ми вирішили обрати мову Python як інструмент для розробки згаданого застосунку. Цей вибір обумовлений декількома ключовими перевагами Python, які роблять його ідеальним для цієї задачі.

По-перше, Python має обширний вибір бібліотек та фреймворків, які можна використовувати для створення анімацій, візуалізацій та інтерфейсу користувача, що є ключовими складовими нашого додатку [6]. По-друге, Python підтримує об’єктно-орієнтоване програмування, що дозволяє організувати додаток у вигляді модулів та класів, спрощуючи його розширення та підтримку [6]. По-третє, Python є кросплатформеною мовою програмування, що стане ключовою перевагою нашого додатку. Це означає, що створений додаток буде працювати на різних операційних системах, таких як Windows, macOS та Linux, без потреби значних змін у коді або адаптації під конкретну платформу [6]. В четверте, Python славиться своєю зрозумілістю та простотою синтаксису, що знижує бар’єр для вступу для студентів, які в подальшому можуть приєднатися до розробки та розширення функціональності цього додатку [6].

Виходячи з цього розробка застосунку на мові Python дасть можливість студентам використовувати застосунок на будь-якому пристрої з будь-якою операційною системою, що значно полегшує доступ до навчальних матеріалів.

Таким чином, використання Python для розробки нашого навчального застосунку не тільки спростить процес його створення, але й значно розширить його потенційну аудиторію, роблячи навчання більш доступним та ефективним.

Для поглибленого вивчення цієї тематики можна звернутися до масових відкритих онлайн курсів як це зроблено у роботи [1].

Для створення анімацій паралельного виконання програм на мові Python, ми розглянемо наведену нижче набір бібліотек, які славляться своїми можливостями у сфері візуалізації та анімації:

– Matplotlib дозволяє створювати як статичні, так і анімовані візуалізації, роблячи її ідеальним інструментом для відтворення динамічних процесів у паралельних програмах.
– Seaborn може бути корисною для підкреслення ключових аспектів даних у паралельних програмах.
– Plotly особливо корисна для розробки інтерактивних візуалізацій, що дозволяють користувачам легко взаємодіяти з представленими даними.
– Bokeh дозволяє створювати докладні інтерактивні діаграми та графіки.
– Pygame дозволяє створювати складні анімаційні сцени, що ідеально підходять для відображення процесів у паралельних програмах.
– PyOpenGL – це Python-інтерфейс для OpenGL, який можна використовувати для створення візуалізацій з використанням 3D-графіки.

Для розробки застосунку, що візуалізує роботу паралельної програми, докладно розглянемо кожну з вищезазначених бібліотек:

1. Matplotlib:

Переваги: Велика гнучкість у створенні різноманітних графіків і діаграм. Підтримує анімацію та інтерактивність, що є корисним для динамічного відображення процесів.

Недоліки: Може бути складною для новачків через велику кількість налаштувань та опцій.

2. Seaborn:

Переваги: Вибудовує більш привабливі та інформативні графіки на базі Matplotlib. Підходить для візуалізації складних наборів даних.

Недоліки: Менш гнучка для дуже специфічних або налаштованих візуалізацій.

3. Plotly:

Переваги: Дозволяє створювати високоякісні інтерактивні візуалізації. Ідеальна для створення складних інтерактивних візуалізацій.

Недоліки: Вимагає інтернет-з'єднання для деяких функцій.

4. Bokeh:

Переваги: Зосереджена на створенні інтерактивних візуалізацій для веб-середовищ. Велика гнучкість у налаштуваннях.

Недоліки: Може бути складною у використанні для новачків.

5. Pygame:

Переваги: Чудово підходить для розробки анімацій та графічних інтерфейсів. Хороший вибір для більш гейміфікованих або інтерактивних проектів.

Недоліки: Орієнтована переважно на ігри, тому може бути не такою зручною для типових наукових візуалізацій.

6. PyOpenGL:

Переваги: Дозволяє використовувати 3D-графіку для створення візуалізацій. Ідеальна для демонстрації складних тривимірних процесів.

Недоліки: Вимагає глибшого розуміння 3D-графіки та OpenGL.

Розглянемо, як обрані бібліотеки відповідають встановленим критеріям оцінки для розробки застосунку анімації роботи паралельної програми:

Функціональні можливості. Matplotlib та Seaborn пропонують широкий спектр функцій для детальної візуалізації, в той час як Plotly та Bokeh забезпечують високий рівень інтерактивності. Pygame та PyOpenGL є потужними для створення динамічних, інтерактивних та 3D-візуалізацій.

Простота використання. Matplotlib та Seaborn можуть бути складними для новачків, але пропонують детальну документацію. Plotly та Bokeh більш інтуїтивні для користувачів з попереднім досвідом роботи з візуалізаціями.

Швидкодія та ефективність. Matplotlib та Seaborn мають добру продуктивність для стандартних візуалізацій. Plotly та Bokeh, хоча вони інтерактивні, можуть вимагати більше ресурсів, особливо при використанні у веб-середовищі. Pygame та PyOpenGL підходять для високопродуктивних анімацій, але вимагають більшого ресурсу комп’ютера.

Сумісність та інтеграція. Всі бібліотеки добре інтегруються з Python і можуть бути комбіновані для різних цілей. Всі раніше розглянуті бібліотеки є відкритими та безкоштовними, що робить їх доступними для освітнього використання.

У результаті аналізу була вибрана бібліотека Seaborn для використання при розробці застосунку. По-перше, вона працює поверх бібліотеки Matplotlib, що дозволяє нам використовувати всі переваги Matplotlib разом з додатковими можливостями Seaborn. По-друге,

Seaborn має широкий набір стилів та параметрів, що дозволяють швидко створювати естетично привабливі графіки та візуалізації. По-третє, ця бібліотека добре інтегрується з Tkinter, що дозволяє нам легко відображати графіки та візуалізації у вікнах нашого додатку, створених з Tkinter. Це робить Seaborn чудовим вибором для наших потреб у візуалізації даних у додатку. Крім того, Seaborn надає зручний інтерфейс для створення діаграм та графіків, що дозволяє легко відображати результати роботи нашого додатку у вигляді інформативних та зрозумілих візуалізацій. Таким чином, Seaborn відповідає нашим потребам у візуалізації даних та інтеграції з інтерфейсом користувача, роблячи її оптимальним вибором для нашого додатку.

Тепер розглянемо алгоритм роботи нашого додатку. На початку роботи нашого додатку студент користується текстовим полем для написання псевдокоду програми. Після цього в областях визначення кількості процесів та процесорів встановлюються параметри віртуального паралельного середовища, і програма запускається. Після запуску в області візуалізації студент спостерігає анімований процес роботи паралельної програми, що дозволяє зрозуміти її динаміку та взаємодію між процесами (Рис 1).

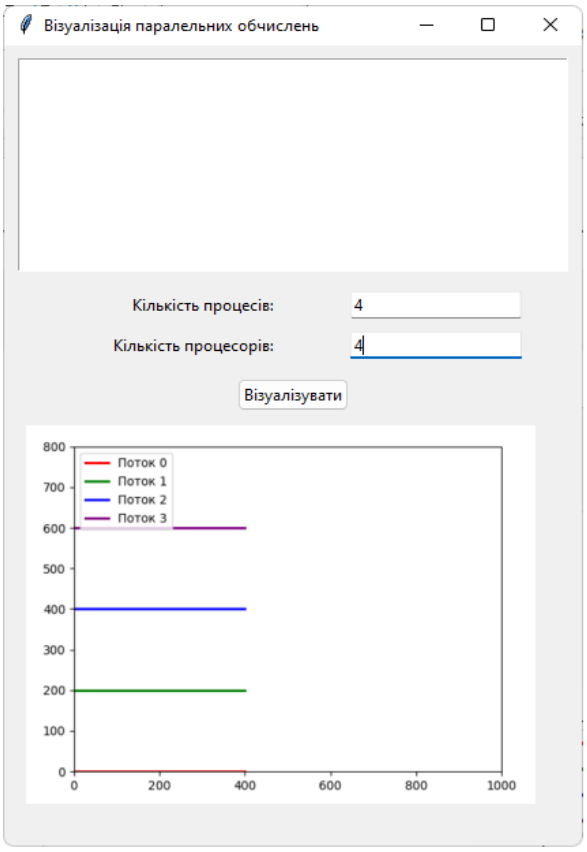

Рис. 1. Програма візуалізації паралельних обчислень

Висновки. Після ретельного аналізу різних бібліотек візуалізації даних ми обрали Seaborn для нашого додатку. Її інтеграція з Matplotlib та простота використання разом із Tkinter роблять її ідеальним вибором. Наш додаток містить зручний інтерфейс, що дозволяє студентам створювати та візуалізувати паралельні програми, а також взаємодіяти з ними у реальному часі. Він оптимально підходить для використання в межах лабораторних та практичних робіт. Можливості Seaborn разом з іншими інструментами надають потужний фреймворк для розвитку нових методів візуалізації, що можуть сприяти глибшому розумінню та вивченню паралельного програмування. Такий підхід відкриває шлях для подальших вдосконалень у навчанні та дослідженнях у цій важливій області комп'ютерних наук.

Література

1. Мірошниченко М. Ю. Використання МООС у якості додаткового засобу для вивчення об'єктно-орієнтованого програмування. *Українські студії в європейському контексті*: зб. наук. пр. 2023. № 7. С. 293–299.
2. Сіциліцин Ю. О. Проєктування візуального середовища для навчання студентів паралельного програмування. *Педагогічні науки: теорія та практика*. 2021. № 2(3). С. 116–120.
3. Computer Science Curricula 2023. URL: https://csed.acm.org/wp-content/uploads/2023/03/Version-Beta-v2.pdf.
4. Di Rocco L., Ferraro Petrillo U., Palini F. Using software visualization to support the teaching of distributed programming. *The Journal of Supercomputing*. 2023. № 79. pp. 3974–3998.
5. Kurniawati A. Kusumaningsih A., Sophan M. K. Visualization Code Tools for Teaching and Learning Introductory Programming. *The 2nd International Conference on Informatics for Development 2018 "DIGITAL OF THINGS"*. 2018. pp. 99–103.
6. Python 3.12.2 documentation. URL: https://docs.python.org/3/.
7. Sitsylitsyn Y. O. et al. Modeling training content for software engineers in parallel computing. *J. Phys.: Conf*. Ser. 2023. 2611 012017.