УДК 004.4:681.3.06

Д. І. Харчук
О. К. Колесницький

РЕАЛІЗАЦІЯ ПАРАЛЕЛІЗМУ ТА РОЗПОДІЛЕНИХ ОБЧИСЛЕНЬ В МОВІ ПРОГРАМУВАННЯ ELIXIR

Вінницький національний технічний університет

Анотація
Запропоновано методи реалізації паралелізму та розподілених обчислень в мові програмування Elixir на базі віртуальної машини Erlang та одного екземпляра BEAM.
Ключові слова: процес, Elixir, паралелізм, багатопоточність, BEAM, відмовостійкість.

Abstract
Methods for implementing parallelism and distributed computing in the Elixir programming language based on the Erlang virtual machine and one BEAM instance are offered.
Keywords: process, Elixir, parallelism, multithreading, BEAM, fault tolerance.

Вступ

На сьогоднішній день інформаційні системи стають все більш масштабні. Для обробки великої кількості даних потрібна значна кількість ресурсів та багато часу. Однією з проблем великих систем є обмеження у ресурсах, що в свою чергу призводить до значних затрат часу на обробку даних.

Метою роботи є реалізація паралелізму в мові програмування Elixir на системі з обмеженими ресурсами.

Результати дослідження

Erlang - це платформа для розробки надійних масштабованих систем з невеликим або нульовим часом простою. Гучні слова, але Erlang існує саме для цих цілей. Ідея створення Erlang зародилася в середині 1980-х років в шведському телекомунікаційному гіганті Ericsson і була обумовлена потребою в задоволенні цілей телекомунікаційних систем компанії, в яких ключову роль грали такі властивості, як надійність, швидке реагування, масштабованість і постійна доступність.

Elixir - це альтернативна мова для роботи з віртуальною машиною Erlang, що дозволяє створювати більш зрозумілий, компактний і інформативний код. Написані на Elixir програми запускаються у віртуальній машині BEAM. Одне з найбільш значущих переваг Elixir - це його здатність помітно скоротити кількість зайвого коду, що спрощує його розробку і підтримку [1].

У конкурентних системах Erlang часто використовується такий структурний елемент, як серверний процес. Серверні процеси - це такі собі конкурентні об'єкти, що мають приватний стан і взаємодіють з іншими процесами за допомогою повідомлень. При цьому відмовостійкість системи значно збільшується. Будучи конкурентними, різні процеси можуть працювати паралельно. Типові Erlang-системи в основному будуються на процесах, яких може бути одночасно запущено тисячі або навіть мільйони.

Важливо розуміти різницю між процесом BEAM і процесом операційної системи - перші набагато більш легковагі. Оскільки мова йде переважно про BEAM, то термін процес вживається в значенні процесу BEAM. Два конкурентні процесі можуть бути запущені паралельно, за умови що доступні хоча б два ядра ЦП. На відміну від процесів і потоків виконання операційної системи, процеси BEAM - легкі одиниці, конкурентне виконання яких контролюється віртуальною машиною за допомогою власного планувальника [2].

За замовчуванням кількість планувальників, що використовується віртуальною машиною BEAM, відповідає кількості доступних ядер ЦП. Наприклад, при запуску коду на комп'ютері з чотирьохядерним процесором задіюється чотири планувальника, як показано на рис 1.

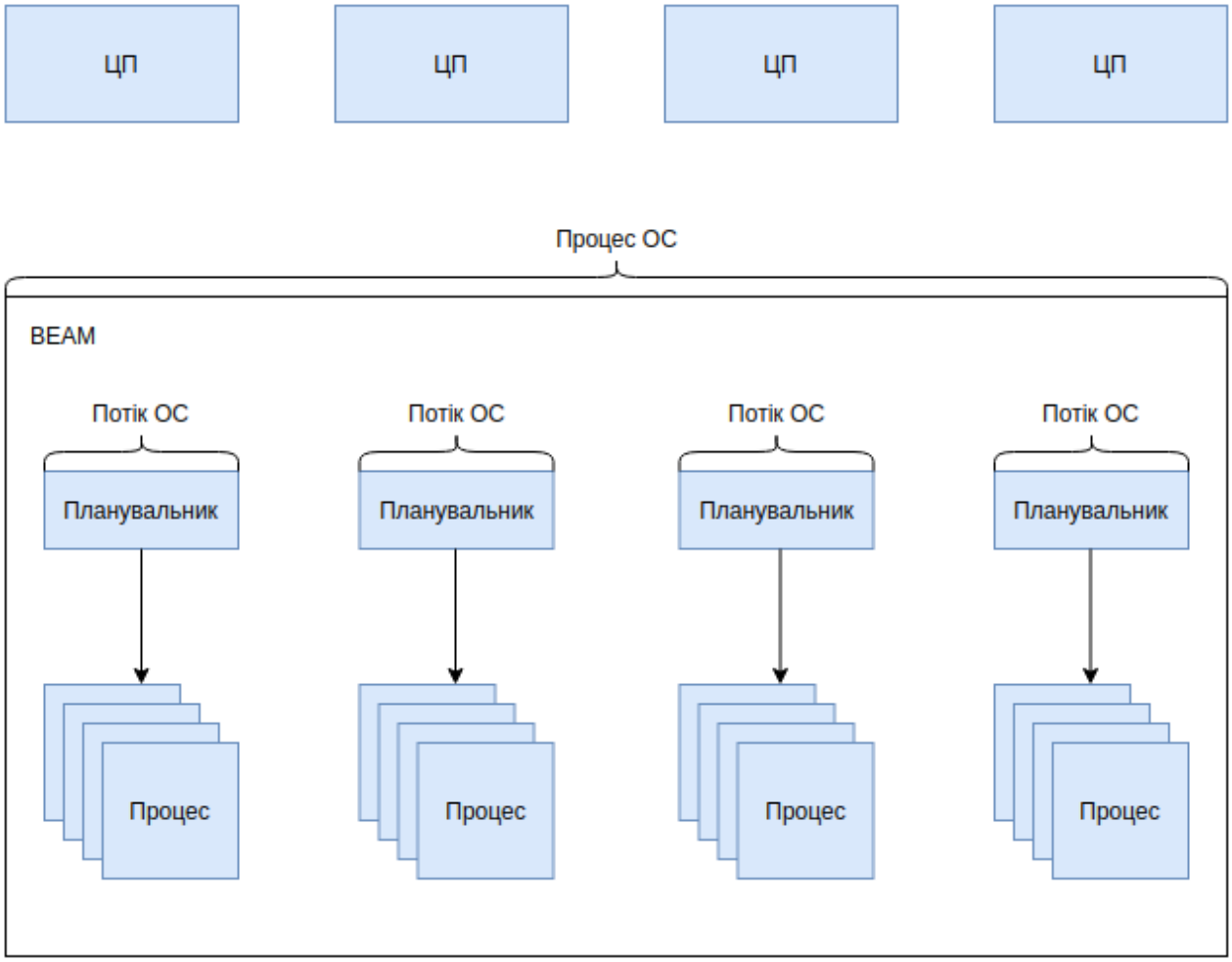

Рис.1. BEAM - окремий процес ОС, що використовує кілька потоків для впорядкування великої кількості процесів

На створення одного процесу витрачається не більше двох мікросекунд, а початковий об'єм пам'яті, який він займає, становить всього декілька кілобайт. Для порівняння: потоки ОС зазвичай споживають пару мегабайтів тільки для зберігання стека. Звідси випливає, що існує можливість створити велику кількість таких процесів; теоретично граничне значення для віртуальної машини досягає 134 млн [3].

Зокрема, це може бути необхідно при розробці системи на стороні сервера для реалізації управління різними завданнями, що виконуються одночасно. Використовуючи для кожного завдання окремий процес, можна задіяти всі можливі ресурси ЦП та по максимуму розпаралелити виконання коду.

Висновки

Встановлено, що паралелізм та розподілені обчислення в мові програмування Elixir реалізовані за рахунок процесів, що запускаються у віртуальній машині BEAM. Вони є досить легкими та не ресурсозатратними, що дозволяє одночасно обробляти значну кількість даних, рівномірно використовуючи ресурси ЦП.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Elixir School [Електронний ресурс] – Режим доступу до ресурсу: https://elixirschool.com/ru/lessons/advanced/otp-concurrency/

2. Saša Jurić. Elixir in Action — M. : Concurrency primitives, 2019. — 149-179 с.

3. Martin Logan, Eric Merritt, and Richard Carlsson. Erlang and OTP in Action — M: The Erlang/OTP platform, 2010. — 13-25 с.

Харчук Денис Ігорович — студент групи 2КН-16б, факультет інформаційних технологій та комп'ютерної інженерії, Вінницький національний технічний університет, Вінниця, e-mail: kharchuk.dennis@gmail.com

Науковий керівник: **Колесницький Олег Костянтинович** — кандидат технічних наук, доцент кафедри комп'ютерних наук, факультет інформаційних технологій та комп'ютерної інженерії, Вінницький національний технічний університет, Вінниця

Kharchuk Denys Igorovich — student of Information Technologies and Computer Engineering Department, Vinnytsia National Technical University, Vinnytsia, e-mail: kharchuk.dennis@gmail.com

Supervisor: **Kolesnytskyy Oleg K.** — Candidate of Science (Engineering), associate professor of Computer Science Department, Informations Technologies and Computer Engineering Faculty, Vinnytsia National Technical University, Vinnytsia