ПАРАЛЕЛЬНЕ ПРОГРАМУВАННЯ У PYTHON ЗА ДОПОМОГОЮ MULTIPROCESSING I SHARED ARRAY

Кривцов Сергій Олегович, аспірант
Національний аерокосмічний університет ім. М.Є. Жуковського «ХАІ»

Однією з особливостей Python є GIL – Global Interpreter Lock. GIL не дозволяє в одному інтерпретатор Python ефективно використовувати більше одного потоку. Існує думка, що однопоточні програми при наявності GIL працюють набагато ефективніше. Але наявність GIL означає, що паралельні обчислення з використанням безлічі потоків і загальної пам'яті неможливі. А це досить сильне обмеження в сучасному світі. В роботі розглянуто підхід до подолання обмежень GIL, заснований на multiprocessing і shared array. Цей спосіб дозволяє досить просто і ефективно використовувати процеси та пам'ять, що розділяється для прозорого паралельного програмування в стилі безлічі потоків і загальної пам'яті.

Як приклад розглянемо наступну задачу. У тривимірному просторі задані N точок v_0, v_1, ..., v_N. Потрібно для кожної пари точок обчислити функцію, залежну від відстані між ними. Результат буде являти собою матрицю NxN зі значеннями цієї функції. В якості опції візьмемо наступну: $f = r^3/12 + r^2/6$. Цей тест, насправді, не такий вже і синтетичний. На обчисленні таких функцій від відстані заснована RBF інтерполяція, яка використовується в багатьох областях обчислювальної математики.

У цій завдачі кожен рядок матриці може обчислюватися незалежно. З кожних кількох рядків матриці сформуємо незалежні роботи і помістимо їх в чергу завдань (рис. 1).

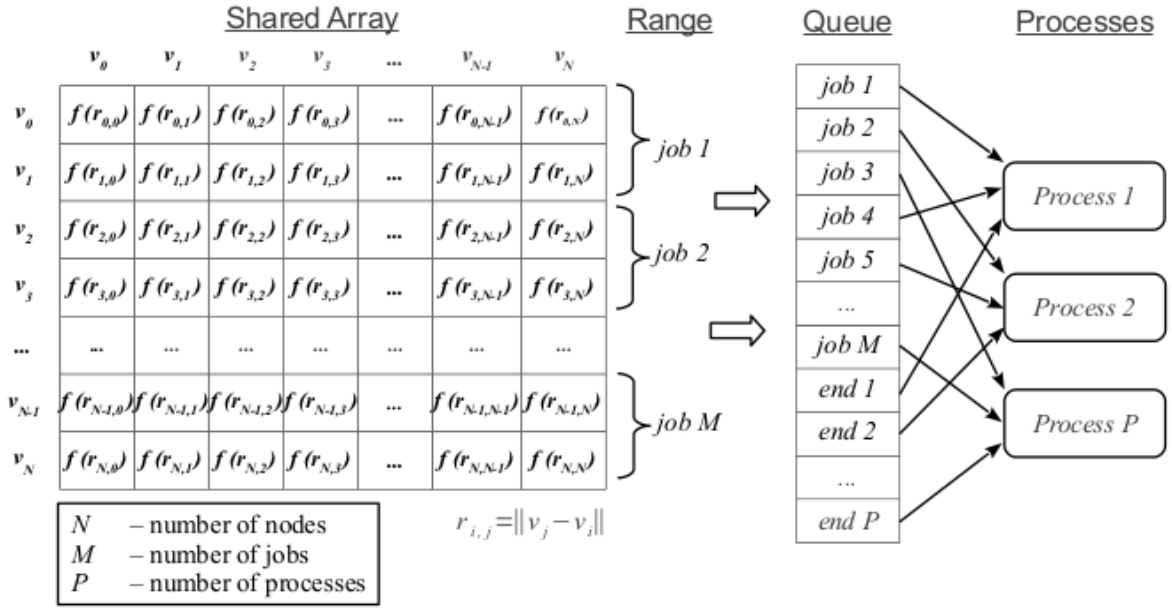

Рисунок 1 – Спосіб паралелизації.

Запустимо кілька процесів. Кожен процес буде брати з черги наступне завдання на виконання, поки не зустріне спеціальне завдання з кодом «end». В цьому випадку процес буде закінчувати свою роботу.

У реалізації на Python у нас будуть два основних методу: mpCalcDistance (nodes) і mpCalcDistance_Worker (nodes, queue, arrD). Метод mpCalcDistance (nodes) приймає на вхід список вузлів, створює область спільної пам'яті, готує чергу завдань і запускає процеси. Метод mpCalcDistance_Worker (nodes, queue, arrD) це обчислювальний метод, який працює у власному потоці. Він приймає на вхід список вузлів, чергу завдань і область спільної пам'яті.

Середовище для виконання тесту: двоядерний процесор, Ubuntu 12.04, 64bit.

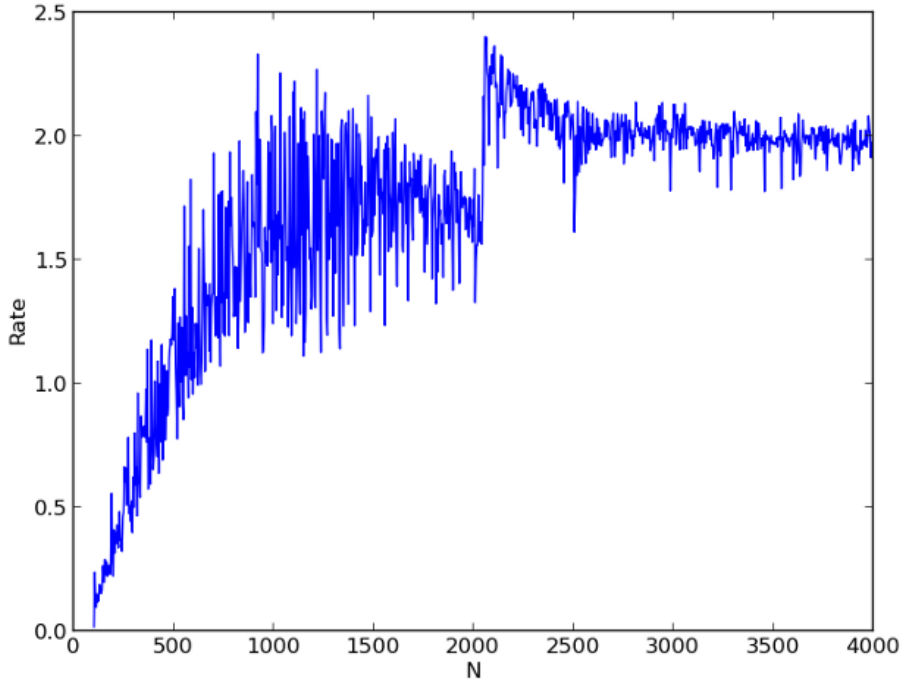

Рисунок 2 – Відношення часу однопотокового розрахунку до двопоточного.

Вочевидь (рис. 2), що починаючи з N = 500 ми отримуємо вже істотне прискорення розрахунків. В районі числа N = 2000 у багатопотоковому розрахунку коефіцієнт прискорення перевищує 2. Це можливо пояснити ефектом кеша. У багатопотоковому варіанті дані для кожного завдання повністю вміщаються в кеш. А в однопоточном вже ні.

Виконано в рамках проєкту Національного фонду досліджень України 2020.02/0404 «Розробка інтелектуальних технологій оцінки епідемічної ситуації для підтримки прийняття управлінських рішень у сфері біобезпеки населення».