УДК 519.63 К

Катерина Зайчук, Ірина Лавренюк
Університет митної справи та фінансів, м. Дніпро

ОСОБЛИВОСТІ ВИКОРИСТАННЯ ПАРАЛЕЛЬНИХ ОБЧИСЛЕНЬ В СУЧАСНІЙ РОЗРОБЦІ

У даній роботі поставлено завдання розглянути навіщо необхідно використовувати потоки, проблеми, які можуть виникнути під час використання потоків та паралельних обчислень. Також у даній роботі наявний опис деяких методів вирішення таких проблем.

Ключові слова: Потоки, паралельні обчислення, «вузькі місця», IDE Visual Studio, інструменти профілювання.

Kateryna Zaichuk, Iryna Lavreniuk

FEATURES OF THE USE OF PARALLEL COMPUTING IN MODERN DEVELOPMENT

In this work, the task is: to consider why it is necessary to use threads, problems that can arise while using threads and parallel computing. Also in this work there is a description of some methods of solving such problems.

Keywords. Threads, parallel computing, "Bottlenecks", IDE Visual Studio, profiling tools.

На сьогоднішній день загальною тенденцією є поява задач, які потребують великої кількості ресурсів для свого вирішення. Певною мірою розвиток технологій здатен забезпечити деякі потужності, але нажаль, цього виявляється недостатньо. Наразі апаратне забезпечення розвивається зі швидкістю, яка нижча за швидкість появи задач. Тому необхідно змінювати наявний підхід до їх розв'язку, тобто, ефективніше використовувати наявні ресурси. Одним з таких рішень є паралельні обчислення.

Також треба відмітити, що для розв'язання задачі за допомогою паралельних обчислень, необхідно, щоб така задача допускала можливість розбиття на підзадачі. Такі підзадачі виконуються кожна у своєму потоці або на своєму процесорі в розподіленій системі. На рисунку 1 показана схема паралельних обчислень.

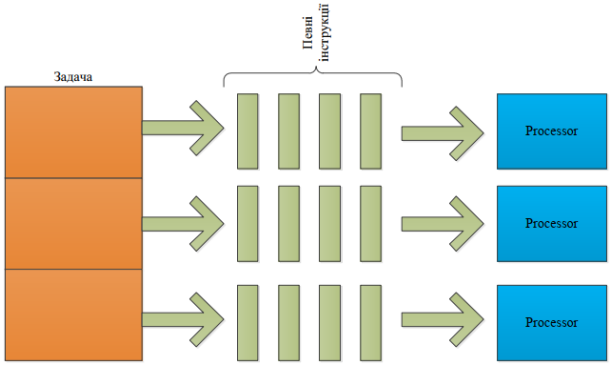

Рис. 1. Схема паралельних обчислень

Оскільки паралельна програма представляє собою множину взаємодіючих паралельних процесів (потоків), то основною метою паралельних обчислень є прискорення рішення обчислювальних задач. При цьому необхідно пам'ятати, що

паралельні програми мають наступні особливості, які необхідно враховувати при написанні коду:

- здійснюється управління роботою безлічі процесів;
- організовується обмін даними між процесами;
- втрачається детермінізм поведінки через асінхронність доступу до даних;
- переважають нелокальних та динамічні помилки;
- з'являється можливість тупикових ситуацій;
- виникають проблеми масштабованості програми і балансування завантаження обчислювальних вузлів.

Не виключена ситуація коли при використанні паралельних обчислень бажана ефективність не досягається. У такому разі можна сказати, що з'явилися «вузькі місця» процесу функціонування. Згідно з, [1] такі вузькі місця можна об'єднати у 3 групи:

- На будь-якому паралельному пристрої потоки даних обробляються не однаково. Деякі потоки реалізуються ефективно, деякі – не дуже.
- Не в кожній програмі обов'язково існують фрагменти які ефективно реалізуються на певному, конкретному пристрої.
- Залежність від використаної в компіляторі технології відображення програм у машинний код [1].

Для вирішення проблеми «вузьких місць» програмісти, найчастіше аналізують інформацію про процес виконання, зібрану під час його виконання.

Якщо ж під час аналізу «вузьких місць» не було виявлено тих частин що працюють найдовше використовують інструменти для профілювання. Коли такі місця будуть виявлені програміст їх модифікує. Таким чином досягається бажана продуктивність системи [2].

Наприклад у IDE Visual Studio є вбудований інструментарій для профілювання коду. Також, передбачено декілька засобів для збору та аналізу продуктивності. Та в більшості випадків рекомендується користуватись параметрами майстра аналізу продуктивності за замовчуванням. За допомогою цього майстра можливо збирати статистику програми, яка дозволить виявити наявність проблем у коді.

Якщо існують певні «загальні» проблеми кодування, то їх можна побачити у вікні помилок Visual Studio. За допомогою таких попереджень існує можливість написати більш ефективний код. Це рішення напряму залежить від програміста.

Звіти інструментів профілювання можливо дивитись на різноманітних рівнях, починаючи від коду написаного програмістом, закінчуючи процесами. Більше того, у таких звітах наявні дані про виконання програми. Наприклад – дерево виклику всієї програми [3].

Література

1. Воеводин В. В. Параллельные Вычисления / В.В. Воеводин, В.В. Воеводин. – Санкт-Петербург: "БХВ-Петербург", 2002. – 608 с.

2. Параллельные заметки №1 – технология OpenMP [Електроний ресурс] URL: https://www.viva64.com/ru/b/0053/ (Дата звернення 12.04.2019).

3. Getting Started with Profiling Tools [Електроний ресурс] URL: https://docs.microsoft.com/en-us/previous-versions/visualstudio/visual-studio-2012/bb385749(v=vs.110) (Дата звернення 12.04.2019)