интервала и правее его. В области левее рабочего интервала ($Q \rightarrow 0$) наблюдаются качественные отличия, что наиболее вероятно вызвано использованием в модели идеальной жидкости.

3. При расчете напорной характеристики методом гидродинамических особенностей на основе вихревой модели получено лучшее количественное согласование величины напора в номинальном режиме по сравнению с потенциальной.

1. *Моргунов Г.М.* Разработка численного метода пространственного безвихревого потока в гидромашинах / Г.М. Моргунов // Тр. МЭИ. Исследование гидромашин. – 1975. – Вып.25. – С.38-49.
2. *Лифанов И.К.* О методе дискретных вихрей / И.К. Лифанов //ПММ. – 1979. – Т. 43. – №1. – С.184-188.
3. *Кацикаделис Джон Т.* Граничные элементы: теория и приложения / Джон Т. Кацикаделис – М.: Издательство АСВ, 2007. – 336 с.
4. *Косторной С.Д.* Математическое моделирование и расчет трехмерного невязкого течения жидкости в лопастных гидромашинах осевого типа с целью определения их силовых и моментных характеристик / С.Д. Косторной, А.К. Давиденко // Вест. Нац. Техн. Ун-та Украины «Киев. политехн. ин-т»: серия «Машиностроение». – 1999.– № 36. – Т. 2. – С.487-494.
5. *Косторной С.Д.* Расчет пространственного потока в рабочем колесе поворотно-лопастных гидротурбин осевого типа. Часть 1 – Моделирование пространственного течения / С.Д. Косторной, А.К. Давиденко // Вісник Сумського державного університету. – 1996. – №2(6). – С.41-46.
6. *Сеффмен Ф.Дж.* Динамика вихрей / Ф. Дж. Сеффмен. – Москва.: Науч. Мир, 2000. – 367 с.
7. *Дынникова Г.Я.* Расчет трехмерных течений несжимаемой жидкости на основе дипольного представления завихренности. // ДАН. 2011. Т. 437, №1. С.35-38.
8. *Этинберг И.Э.* Гидродинамика гидравлических турбин / И.Э. Этинберг, Е.С. Раухман – Л.: Машиностроение, 1978. – 280 с.

Поступила 19.02.2018р.

УДК 004.896

С.В. Сушко, О.А. Чемерис, Київ

ВПЛИВ РОЗМІРІВ БЛОКІВ РОЗБИТТЯ ОПЕРАТОРІВ ЦИКЛІВ НА ЧАС ВИКОНАННЯ КОМП'ЮТЕРНИХ ПРОГРАМ

Abstract. Article reviews optimization methods of the computational loops. Complicated character of the time processing depending on the tiles sizes is shown. The analysis of the efficiency tiling method is performed relying on the obtained numerical values and pictures.

Актуальність

Паралелізація однопоточної програми полягає в розподілу множини операцій з даними, що розміщені в пам'яті, на множину процесорних елементів деякої конкретної топології. Мультипроцесорна комп'ютерна система має вирішити задачу балансування процесорів для зменшення числа невикористаних процесорів. Такий підхід призводить до зменшення часу виконання програми. Іншою задачею зростання ефективності є мінімізація передач даних поміж вузлами мультипроцесорної комп'ютерної системи. Вирішення цієї задачі призводить до більш ефективного розпаралелювання.

Перетворювання обчислювальних циклів, таких як розбиття циклів та злиття циклів може значно спростити паралелізацію та покращити її ефективність [1]. Паралелізація найефективніша в обчислювальних циклах з великою кількістю ітерацій. Більша кількість ітерацій робить можливим балансування навантаженням з урахуванням наявності більшої кількості задач для розподілення поміж потоків.

Одним з ефективних методів трансформації обчислювальних циклів є метод розбиття на блоки [2]. Цей оптимізаційний метод полягає у розділенні ітераційного простору початкового обчислювального циклу, що може складатись з деяких змінних, на блоки меншого розміру. Розбиття масиву на менші блоки та зберігання їх у кеші призводить до частого використання кешу, зменшення кеш-промахів та зменшення вимог на розмір кешу [3].

Постановка задачі оптимізації програмного забезпечення

Оптимізація програмного забезпечення це процес модифікації або трансформації вихідного програмного коду для досягнення кращої продуктивності (швидкодії, енергії, що необхідна для розрахунків, енергоефективність, розмір пам'яті програм або даних та інші) з тим самим отриманим результатом.

Оптимізаційна задача [4] для кожного і-го обчислювального циклу може бути визначена за формулою:

$$F_i = \arg\min(f(\vec{M}_i, \vec{P}_i)), \quad (1)$$

де F_i – параметр, що потрібно оптимізувати, M – вектор оптимізаційних методів, P – вектор параметрів оптимізаційних векторів.

argmin() визначається як пошук такого аргумента (одного або декількох) функції, при якому досягається мінімальне значення функції:

$$\arg\min_x(f(x)) \in \{x \mid \forall y : f(x) \le f(y)\}, \quad (2)$$

Оптимізаційна задача для всієї програми, що складається з N обчислювальних циклів може бути визначена за формулою:

$$F = \arg\min(\sum_{i=1}^{N} f(\vec{M}_i, \vec{P}_i)), \quad (3)$$

Всі операції виконуються на цілочисельному базисі, що визначає та обмежує індекси змінних циклу. Вони визначають розмірності та розміри ітераційного простору.

Визначення. Ітераційний простір це множина всіх цілочисельних векторів $I = (I_1, I_2, \ldots, I_n)$, що задовольняють нерівності:

$$L_i \le x_i \le \mathrm{U}_i, i = 1..n, \tag{4}$$

Нерівність (4) визначає межі циклу, що обмежують ітераційний простір опуклим багатогранником.

Як було показано в [5] та [6] розбиття на блоки та розпаралелювання часто призводять до покращення в термінах швидкодії та енергоефективності. Проте вказані дослідження не розкривають, як саме впливає розмір блоку розбиття на вказані характеристики і якщо впливає, то в якій мірі та який характер цієї залежності. Щоб перевірити цей вплив було проведено ряд експериментів.

Експерименти

Для перевірки впливу розмірів блоків на час виконання програм було вибрано фреймворк Pluto [7]. Він включає як можливість розпаралелювання коду, так і різноманітні методи розбиття на блоки, які можливо використовувати поодинці або сумісно з розпаралелюванням. Для експериментів було вибрано два режими роботи оптимізаційної програми: tile а також tile, innerpar та parallel одночасно. Другий режим це більш просунутий метод розбиття на блоки з одночасним розпаралелюванням, що використовує всі ядра за допомогою бібліотеки OpenMP.

Джерелом тестових програм було обрано Polybench test [8], що містить близько трьох десятків вживаних алгоритмів з різних галузей обробки даних. Загалом 17 тестових програм пройшли тестування у двох вищезгаданих режимах. Для мінімізації помилки вимірювання для кожного режиму виконувалось п’ять вимірів. Найбільше та найменше значення не враховувалися, а середнє значення поміж решти трьох приймалося за отримане значення. Тести проводилися на настільному ПК с чотириядерним процесором Intel Core i5-4670K під керуванням операційної системи Ubuntu 14.04 LTS.

Pluto використовує розбиття на блоки по двом вимірам. Для отримання репрезентативних результатів проводилися виміри при різних наборах обох параметрів у визначеному діапазоні значень. При попередньому тестуванні на одному тесті було визначено, що другий параметр має значний вплив, у той час як зміна першого не сильно впливає на результат. Тому, зважаючи на довгий час виконання тестування, було прийнято рішення обмежити діапазони розмірів блоків [16; 32] та [64; 1024] для першого та другого блоку відповідно.

Деякі з отриманих результатів приведено на рис. 1 – 5. Графіки показують залежність часу виконання від двох розмірів блоків розбиття.

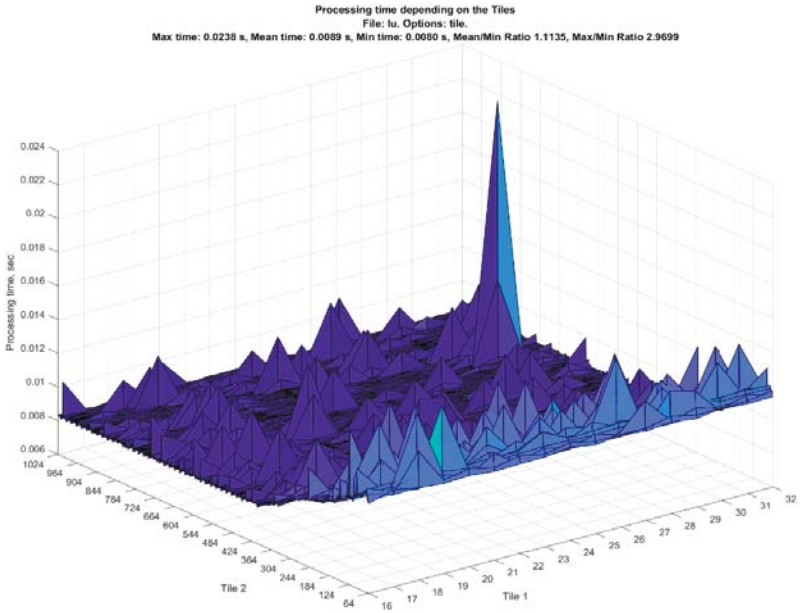

Рис. 1. Залежність 1 часу виконання програми

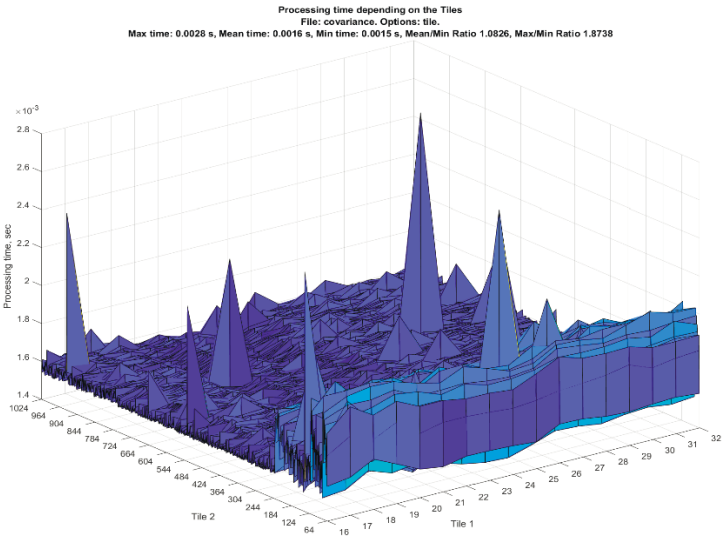

Рис. 2. Залежність 2 часу виконання програми

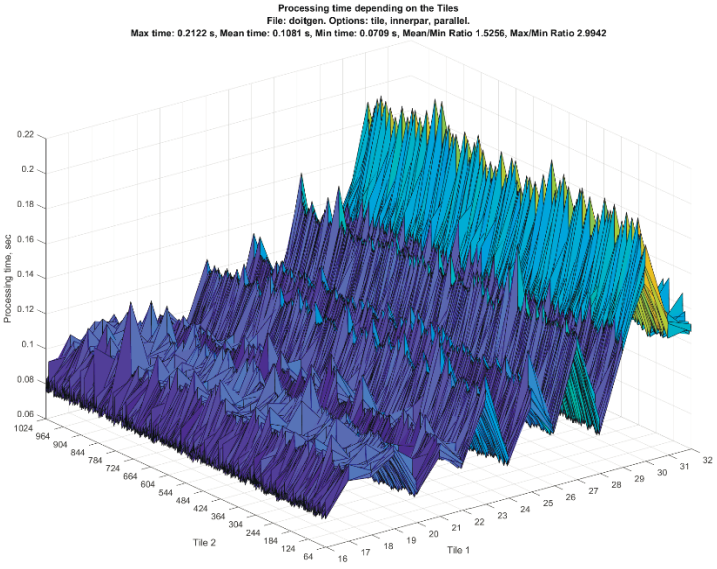

Рис. 3. Залежність 3 часу виконання програми

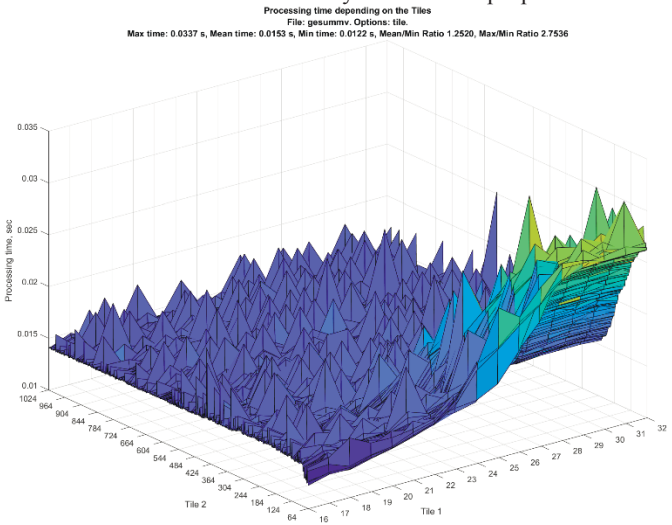

Рис. 4. Залежність 4 часу виконання програми

В деяких випадках, як зображено на рисунку 5, час виконання програм може значно зростати на деяких значеннях. Це викликано процесами в середині операційної системи, що виконує фонові процеси під час вимірювання часу виконання основних програм. Такі значення є хибними, вони зменшуються при повторному вимірі і тому не мають впливати на загальні висновки.

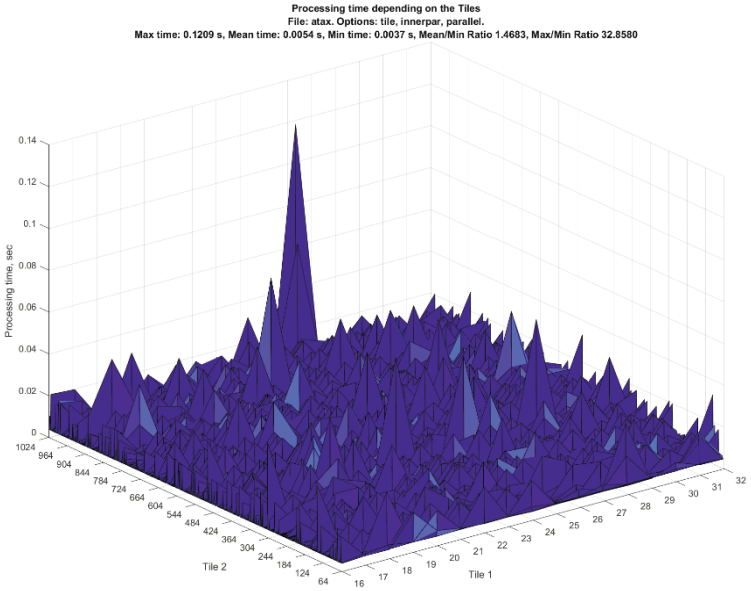

Рис. 5. Залежність 5 часу виконання програми

Для числового аналізу отриманих даних було введено дві метрики вимірюваних значень – відношення середнього значення до мінімального та відношення максимального значення до мінімального. Такі оцінки дозволяють оцінити, в скільки разів правильний підбір розмірів блоків розбиття може покращити час виконання в середньому випадку і в найгіршому випадку. Такі оцінки приведено у табл. 1.

Таблиця 1

Оцінки можливого прискорення часу виконання програм

Тестові програми	Опція Tile		Опції Tile + Innerpar + Parallel	
	Mean/Min	Max/Min	Mean/Min	Max/Min
2mm	1.0254	2.4929	1.5102	43.7708
3mm	1.0277	1.4658	1.3991	13.7480
atax	1.1341	1.7816	1.4683	32.8580
bicg	1.1716	2.3085	1.4212	46.5144
cholesky	1.0571	1.5511	2.7493	28.6245
cov	1.0826	1.8730	2.0335	254.5727
doitgen	1.6403	2.2937	1.5256	2.9942
durbin	1.0128	1.5814	1.0157	2.0391
gemm	1.1397	1.7401	1.2237	2.4104

gemver	1.2147	1.5456	1.2490	5.8665
gesummv	1.2520	2.7536	1.2958	13.7526
gramsch	1.6527	2.6383	2.0542	16.2835
lu	1.1135	2.9699	2.2271	48.8295
mvt	1.2086	2.4393	1.3390	8.2386
symm	1.0255	1.6794	1.0208	1.5740
syr2k	1.2027	1.9711	1.5365	15.8442
syrk	1.0466	2.2875	1.6282	28.8474

Висновки

Як можна бачити по отриманим даним, розмір блоків загалом суттєво впливає на час виконання програм.

Час однопотокового розбиття на блоки, отриманий при використанні випадкових значень блоків, зазвичай більше до 20% чим мінімальний час. В деяких випадках цей час може бути більше до 65%.

Час багатопотокового розбиття на блоки, отриманий при використанні випадкових значень блоків, зазвичай більше до 40% мінімального часу. В деяких випадках цей час може бути більшим в два рази. В обох конфігураціях найгірший час виконання у випадку, коли розміри блоків були обрано дуже невдало може бути покращено в декілька разів.

Варто відмітити те, що характер залежностей дуже неоднорідний. Не можна виділити один чи декілька шаблонів залежностей часу виконання від розмірів блоків. Цей факт суттєво ускладнює знаходження оптимальних значень. Найкращі значення блоків розбиття не можуть бути визначені до тестування або на основі деяких значень. Пошук оптимальних значень розмірів блоків розбиття являє собою окрему задачу, що являє собою практичну цінність з урахуванням отриманих значень.

1. *Kodary K., Mignotte A.* “Loop fusion for memory space optimization”, The 14th International Symposium on System Synthesis, 2001, Proceedings, P.95-100, 2001.
2. *Xue J.* Loop Tiling for Parallelism. Kluwer Academic Publishers. 2000.
3. *Lam M.S., Rothberg E.E., Wolf M.E.* The cache performance and optimizations of blocked algorithms. In Proceedings of the 4th International Conference on Architectural Support for Programming Languages and Operating Systems, P.63-74, April 1991.
4. *Карманов В.* Г. Математическое программирование: Учеб. пособие. – 5-е изд., стереотип. – М.: ФИЗМАТЛИТ, 2004. – 264 с. – ISBN 5-9221-0170-6.
5. *Чемерис А., Сушко С.*. Исследование быстродействия и энергопотребления при автоматической оптимизации методами разбиения на блоки и распараллеливания для вычислений на платформе x64 / А. Чемерис, С. Сушко // Збірник наукових праць Інституту проблем моделювання в енергетиці ім. Г.Є. Пухова. – К.: ІПМЕ ім. Г.Є. Пухова НАН України, 2017. – №80. – С. 52–60.
6. *Chemeris A., Lazorenko D., Sushko S.*. Influence of Software Optimization on Energy Consumption of Embedded Systems – Green IT Engineering: Components, Networks and Systems Implementation. Springer, 2017.

7. *Uday Bondhugula* Effective Automatic Parallelization and Locality Optimization Using The Polyhedral Model. – The Ohio State University, 2010.
8. *Pouchet L.N.*, The polyhedral benchmark suite [Online]. Available: http://web.cs.ucla.edu /*pouchet/software/polybench/ 22 Sept 2016

Поступила 22.02.2018р.

УДК 004.032.26 : 004.056.55

І.Г. Цмоць, д.т.н., НУ «Львівська політехніка», Львів
Ю.В. Цимбал, к.т.н., НУ «Львівська політехніка», Львів
О.В. Скорохода, к.т.н., НУ «Львівська політехніка», Львів
В.М. Хавалко, к.т.н., НУ «Львівська політехніка», Львів
Т.В. Теслюк, НУ «Львівська політехніка», Львів

АПАРАТНА РЕАЛІЗАЦІЯ НЕЙРОМЕРЕЖЕВИХ ЗАСОБІВ ШИФРУВАННЯ-ДЕШИФРУВАННЯ ІНФОРМАЦІЙНИХ ПОТОКІВ ДАНИХ

Abstract. The “model of successive geometric transformations” paradigm has been adapted for the implementation of parallel-streaming neural network encryption-decryption of data in real time. A model and structure of a parallel-streaming neural-like network for the mode have been developed.

Keywords*:* intensive data stream; neural networks; geometric transformations model

Постановка проблеми

Розвиток новітніх інформаційних технологій та засобів комунікацій, забезпечують все більш широкі можливості доступу до інформаційних ресурсів і переміщення великих масивів даних на необмежені відстані. Широке впровадження інформаційних технологій робить закономірною та актуальною проблему захисту передачі інформації з використанням криптографічних методів, які забезпечують шифрування готової до передачі інформації. Зашифрована інформація передається каналом зв’язку до санкціонованого користувача, який після її отримання виконує дешифрування за допомогою зворотного перетворення. Криптографічні перетворення здійснюються шляхом використання спеціальних алгоритмів. Для шифрування та дешифрування потоків даних у реальному часі пропонується використати нейроподібні мережеві алгоритми, ключем в яких використовуються архітектура мережі, вагові коефіцієнти та коди маскування. Забезпечити реальний час шифрування та дешифрування інтенсивних потоків можна шляхом НВІС-реалізації відповідних алгоритмів. Для синтезу нейроподібних елементів і нейроподібних мереж реального часу необхідно

