УДК 004.3+519.681.5

DOI: 10.15407/fmmit2025.40.047

Аналіз реалізації паралельних алгоритмів цифрової фільтрації великих масивів спотворених даних

Ростислав Вдович[1], Михайло Яджак[2,3]

[1] Аспірант, Інститут прикладних проблем механіки і математики ім. Я. С. Підстригача НАН України, вул. Наукова, 3-Б, м. Львів, 79060, Україна, e-mail: rostyslav.r.vdovych@gmail.com

[2] Доктор фізико-математичних наук, старший науковий співробітник, Інститут прикладних проблем механіки і математики ім. Я. С. Підстригача НАН України, вул. Наукова, 3-Б, м. Львів, 79060, Україна,

[3] Доктор фізико-математичних наук, старший науковий співробітник,Львівський національний університет імені Івана Франка, вул. Університетська, 1, м. Львів, 79000, Україна, e-mail: yadzhak_ms@ukr.net

Розглянуто паралельні алгоритми розв'язання одновимірної задачі цифрової фільтрації масивів даних із застосуванням синхронної схеми обчислень. Здійснено програмну реалізацію цих алгоритмів на багатоядерному комп'ютері з використанням мови високого рівня C# та середовища Visual Studio 2022 із залученням бібліотек System та System.Threading. Під час виконання згаданих алгоритмів фільтрації застосовувались інструменти ThreadPool та Parallel.For. Чисельні експерименти проводилися для декількох типів спотворених сигналів з використанням різних способів задання вагових коефіцієнтів. Унаслідок цього одержано суттєве реальне прискорення паралельних алгоритмів, яке підтвердило їх високу ефективність. Одержані результати можуть бути використані під час попереднього опрацювання великих масивів даних у багатьох предметних галузях з допомогою сучасних програмних та апаратних засобів.

Ключові слова: цифрова фільтрація, метод пірамід, паралельний алгоритм, автономні гілки, обмежений паралелізм, прискорення обчислень, багатоядерний комп'ютер.

Вступ. Під час дослідження об'єктів складних ієрархічно-мережевих систем зазвичай використовують значні обсяги вхідних даних про них (характеристики, особливості режимів функціонування, можливі критерії оцінювання тощо) [1–3]. Перед подальшим використанням ці дані потрібно попередньо опрацювати, використовуючи процедуру цифрової фільтрації, щоб усунути завади або хоча б зменшити їх рівень. У більшості випадків таку процедуру потрібно виконувати в режимі реального часу. З цією метою було запропоновано низку ефективних паралельних алгоритмів фільтрації, орієнтованих на різні типи архітектур паралельних обчислювальних засобів (систолічні та квазісистолічні структури [4–6], комп'ютери зі структурно-процедурною організацією обчислень, кластери, гібридні архітектури, комп'ютери з багатоядерними процесорами [7] тощо). Зокрема, в [7] розглянуто паралельні алгоритми фільтрації, що реалізують синхронну схему обчислень, і на їх основі побудовано алгоритми з автономними гілками та обмеженим паралелізмом. Однак, не достатньо дослідженою є проблема аналізу побудованих алгоритмів з використанням різних засобів реалізації паралелізму в

мовах програмування високого рівня, зокрема в *C#*. Цій проблемі і присвячена наша робота. Далі коротко опишемо розглядувану задачу фільтрації та паралельні алгоритми її розв'язання, для яких будемо оцінювати реальне прискорення.

1. Задача цифрової фільтрації

Розглянемо задачу цифрової фільтрації (ЗЦФ), що в загальному випадку полягає у виконанні *C* переобчислень згладжування масиву значень *N* змінних через рухоме вікно розміром *M* [6]. Послідовний алгоритм розв'язання її одновимірного варіанту можна подати у вигляді (1). Зазвичай на практиці виконуються нерівності: $N >> C,\ N >> M$.

$$
\begin{array}{l}
FOR\quad t = 1,\ C\quad DO \\
\{\quad FOR\quad i = 1,\ N\quad DO \\
\{\quad p = 0 \\
FOR\quad s = -m,\ m\quad DO \\
\{\quad p = p + x_{i+s}^{t-1} * f_s\quad \} \\
x_i^t = p\quad \}\quad \}.
\end{array}
\qquad (1)
$$

Тут $M = 2m+1$; x_i^0, x_i^t – відповідно початкове та переобчислене на *t*-му кроці значення змінної x_i; вагові коефіцієнти $f_s\,(s = \overline{-m,m})$, а також «заграничні» значення $x_{1-m}, x_{2-m}, \ldots, x_0$; $x_{N+1}, x_{N+2,}, \ldots, x_{N+m}$ є відомими константами. Для свого виконання наведений алгоритм потребує час $T_1 = t_{op}(2m+1)CN$, де t_{op} – час виконання подвійної операції $a + b * c$ [7].

2. Паралельні алгоритми фільтрації

Для розв'язання сформульованої одновимірної ЗЦФ розглянемо паралельний алгоритм (2), який реалізує синхронну схему обчислень. Згідно з цим алгоритмом [4], для переобчислення значень змінних на *j*-му кроці беруться значення, переобчислені виключно на (*j*-1)-му кроці. При цьому на кожному кроці здійснюється синхронізація паралельних гілок.

$$
\begin{array}{l}
FOR\quad t = 1,\ C\quad DO \\
\{\quad FOR\quad i = 1,\ N\quad DO\quad PAR \\
\{\quad p_i = 0 \\
FOR\quad s = -m,\ m\quad DO \\
\{\quad p_i = p_i + x_{i+s} * f_s\quad \} \\
x_i = p_i\quad \}\quad \}.
\end{array}
\qquad (2)
$$

У наведеному алгоритмі типом паралелізму *PAR* є *SIM* (simultaneous). Якщо знехтувати часом на забезпечення синхронізації, то (2) реалізуватиме обчислення за час $T_s = t_{op}(2m+1)C$, а його прискорення $S = N$.

У [7] на підставі використання ідей методу пірамід для розпаралелювання циклів стосовно (1) запропоновано паралельний алгоритм фільтрації (3) з автоно-

мними N гілками. Результатом його роботи є переобчислені значення N змінних на C-му кроці. Кожна паралельна гілка складається з ітерацій, які впливають на результуючу. Однак, в алгоритмі (3) спостерігається дублювання обчислень на низці ітерацій, що робить його неекономним. Попри це, в багатьох випадках цей алгоритм є доволі ефективним.

$$
\begin{aligned}
&FOR \quad k = 1,\ N \quad DO \quad PAR \\
&\{ \quad FOR \quad t = 1,\ C \quad DO \\
&\{ \quad FOR \quad i = \max\{1,\ m(t-C)+k\},\ \min\{N,\ m(C-t)+k\} \quad DO \\
&\{ \quad p = 0 \\
&FOR \quad s = -m,\ m \quad DO \\
&\{ \quad p = p + x_{i+s}^{t-1} * f_s \quad \} \\
&x_i^t = p \quad \} \quad \} \quad \}.
\end{aligned}
\tag{3}
$$

У даному разі *PAR* є *AUTON* (autonomous), тобто конструкція (3) задає паралельне виконання N гілок, які не взаємодіють між собою. Час функціонування цього алгоритму обчислюється за формулою $T_p = t_{op}(2m+1)C(1+m(C-1))$, а його прискорення $S_N = N/(1+m(C-1))$.

Дещо зменшити частку дублювань обчислень у паралельних гілках дозволяє використання алгоритму фільтрації (4) з обмеженим паралелізмом, який враховує, що $N > P$, де P – кількість реально виконуваних паралельних гілок [7]. Гілки в (4) суттєво завантажені корисною (обчислювальною) роботою. Цей алгоритм орієнтований на заданий обсяг доступних ресурсів (кількість процесорів, ядер, обчислювальних вузлів, обсяг оперативної пам'яті) у наявному паралельному комп'ютері.

$$
\begin{aligned}
&FOR \quad k = 1,\ P \quad DO \quad AUTON \\
&\{ \quad FOR \quad t = 1,\ C \quad DO \\
&\{ \quad FOR \quad i = \max\{1,\ m(t-C)+(k-1)N/P+1\}, \\
&\qquad\qquad \min\{N,\ m(C-t)+kN/P\} \quad DO \\
&\{ \quad p = 0 \\
&FOR \quad s = -m,\ m \quad DO \\
&\{ \quad p = p + x_{i+s}^{t-1} * f_s \quad \} \\
&x_i^t = p \quad \} \quad \} \quad \}.
\end{aligned}
\tag{4}
$$

Час виконання паралельного алгоритму (4) обчислюємо за такою формулою: $T_{po} = t_{op}(2m+1)C(N/P + m(C-1))$. Прискорення цього алгоритму $S_P \approx P$ з урахуванням певних співвідношень [7] між параметрами задачі фільтрації. Для простоти викладу тут вважаємо, що N є кратним до P, однак це обмеження не є суттєвим для опису та роботи паралельного алгоритму.

3. Опис програмної реалізації

Для аналізу було вибрано алгоритми (2) та (4) і реалізовано їх на комп'ютері з шестиядерним процесором Intel Core i5-9600 KF на мові програмування високого

рівня *C#* у середовищі Visual Studio 2022 із залученням бібліотек System та System.Threading. Кожен із цих паралельних алгоритмів фільтрації був виконаний двома способами із застосуванням відповідно ThreadPool та Parallel.For.

Чисельний експеримент проводився для різних наборів значень параметрів задачі фільтрації. Для кожного з цих наборів розглядалось по декілька масивів вхідних даних. Як вхідні дані для алгоритмів фільтрації використовувались спотворені (переважно імпульсними завадами) сигнали різних типів: широкий прямокутний імпульс, пилоподібний, гармонійний та трикутний сигнали. Загалом ми розглядали завади з рівномірним та нормальним розподілом. Для вибору вагових коефіцієнтів використовувались відомі [8] вікна Бартлета (Bartlett), Блекмана (Blackman), Даніеля (Daniell), Парзена (Parzen) та Хемінга (Hamming). Зауважимо, що у цьому випадку (за виключенням вікна Даніеля) переважно вага більш далеких від центру вікна точок буде поступово зменшуватися майже до нуля на межі вікна. До того ж вагові коефіцієнти вибираються такими, щоб їх сума дорівнювала 1.

4. Аналіз результатів чисельних експериментів

Реальний час виконання паралельних алгоритмів (2) та (4) ми порівнювали з часом виконання відповідного послідовного алгоритму (1), еквівалентного до них за інформаційним графом [7].

Таблиця 1

Прискорення паралельних алгоритмів (2) та (4)

C	*m* = 3			
	N = 120000		*N* = 1200000	
3	3,453	4,149	3,144	3,762
	4,719	4,641	4,268	4,295
4	3,446	4,186	3,207	3,805
	4,982	4,879	4,400	4,457
5	3,469	4,164	3,129	3,843
	4,938	5,054	4,455	4,420
6	3,444	4,168	3,149	3,756
	5,048	4,884	4,522	4,524
7	3,477	4,179	3,188	3,753
	4,893	4,788	4,512	4,541
8	3,341	4,127	3,146	3,780
	5,061	5,001	4,510	4,477

Результати проведених чисельних експериментів для заданих значень параметрів ЗЦФ *N*, *m*, *C* подані у таблицях 1, 2. Кожному набору цих значень відповідає чотири значення реального прискорення паралельних алгоритмів. Зокрема, у верхньому рядку маємо прискорення алгоритму (2), одержане із застосуванням ThreadPool та Parallel.For відповідно, а у нижньому рядку – прискорення (4), одержане із застосуванням тих же засобів.

Таблиця 2

Прискорення паралельних алгоритмів для різних значень *m*

m	*N* = 1200000

	$C = 5$		$C = 10$	
1	3,217	3,536	3,502	4,113
	4,091	4,023	4,551	4,568
2	3,360	3,921	3,559	4,245
	4,529	4,515	4,941	4,873
3	3,027	3,539	3,456	4,107
	4,473	4,482	4,932	4,804
4	3,229	4,000	3,495	4,201
	4,421	4,289	4,606	4,645
5	3,258	3,759	3,486	4,251
	4,636	4,684	5,154	4,904
6	3,286	3,882	3,558	4,319
	4,861	4,854	5,143	5,058
7	3,290	3,977	3,542	4,224
	5,039	4,775	5,161	5,171

Результати обчислень засвідчили, що прискорення паралельних алгоритмів (2) та (4) під час їх реалізації на комп'ютері з багатоядерним процесором є суттєвим. При цьому важко надати перевагу якомусь із розглянутих способів реалізації паралелізму. Однак, в більшості випадків прискорення алгоритму з обмеженим паралелізмом (4) є більшим, ніж прискорення паралельного алгоритму (2), в якому використовується синхронізація гілок.

Висновки. У роботі розглянуто деякі паралельні алгоритми (з синхронізацією та автономними гілками) розв'язання одновимірної ЗЦФ. Здійснено реалізацію паралельного алгоритму, що використовує синхронну схему обчислень, та алгоритму з обмеженим паралелізмом на комп'ютері з багатоядерним процесором із застосуванням сучасних програмних засобів. Одержано реальне прискорення обчислень, яке підтверджує високу ефективність цих алгоритмів на практиці. Результати роботи можуть бути використані як під час аналізу паралельних алгоритмів розв'язання алгоритмічно складних задач, так і для попереднього опрацювання в режимі реального часу великих масивів вхідних даних у різних предметних галузях, зокрема під час моделювання об'єктів і процесів у біологічних [9], природних [10] та штучних [11–13] складних системах, зокрема системах з ієрархічно-мережевою структурою [1, 3, 14].

Література

1. *Ravasz E., Barabasi A.-L.* Hierarchical organization in complex networks. *Physical Review E*. 2003. Vol. 67, N 2. 026112.
2. *Polishchuk O., Polishchuk D., Tyutyunnyk M., Yadzhak M*. Big Data Processing in complex hierarchical network systems. *arXiv: 1603.00633* [*physics.data-an*]. 2016. 7 p.
3. *Поліщук О. Д., Яджак М. С.* Моделі та методи комплексного дослідження складних мережевих систем та міжсистемних взаємодій. Львів: Інститут прикладних проблем механіки і математики ім. Я. С. Підстригача НАН України, 2023. 385 с.
4. *Valkovskii V. A.* An optimal algorithm for solving the problem of digital filtering. *Pattern Recognition and Image Analysis*. 1994. Vol. 4, N 3. P. 241–247.
5. *Jadzhak M. S.* On Optimal in One Class Algorithm for Solving Three-Dimensional Digital Filtering Problem. *Journal of automation and information sciences*. 2001. Vol. 33, N 1. P. 51–63.
6. *Яджак М. С.* Високопаралельні алгоритми та засоби для розв'язання задач масових арифметичних і логічних обчислень. *Автореф. дис. ... д. ф.-м. н.: спеціальність 01.05.03 – математичне та*

програмне забезпечення обчислювальних машин і систем. Київ: КНУ імені Тараса Шевченка, 2009. 33 с.
7. *Яджак М. С.* Паралельні алгоритми цифрової фільтрації даних. *Кібернетика та системний аналіз*. 2023. Т. 59, № 1. С. 46–56.
8. *Oppenheim A. V., Schafer R. W., Buck J. R.* Discrete-Time Signal Processing'. Upper Saddle River. New Jersey, 1999. 893 p.
9. *Lesne A.* Complex networks: from graph theory to biology. *Letters in Mathematical Physi*cs. 2006. Vol. 78, N 3. P. 235–262.
10. *Valdez L. D., Braunstein L. A., Havlin S.* Epidemic spreading on modular networks: the fear to declare a pandemic. *arXiv: 1909.09695v2* [*physics.soc-ph*]. 23 March 2020. 38 p.
11. *Berbyuk V. E., Demidyuk M. V.* Parametric optimization in problems of dynamics and control of motion of an elastic manipulator with distributed parameters. *Mechanics of Solids*. 1986. Vol. 21, is. 2. P. 78–86.
12. *Берб'юк В. Є., Демидюк М. В., Литвин Б. А.* Параметрична оптимізація ходи та пружних характеристик пасивних приводів двоногого крокуючого робота. *Вісник Київського університету. Серія: Кібернетика*. 2002. № 3. С.17–20.
13. *Jackson M. O.* Social and economic networks. Princeton: Princeton University Press, 2010. 520 p.
14. Hierarchy in natural and social sciences (ed. by Pumain D.). Springer. Printed in the Netherlands, 2006. 246 p.

Analysis of the implementation of parallel algorithms for digital filtering of large arrays of distorted data

Vdovych Rostyslav, Yadzhak Mykhailo

Parallel algorithms for solving a one-dimensional problem of digital filtering of data arrays using a synchronous calculation scheme are considered. These algorithms are implemented on a multi-core computer using the high-level language C# and the Visual Studio 2022 environment with the involvement of the System and System.Threading libraries. The ThreadPool and Parallel.For tools were used when executing the aforementioned filtering algorithms. Numerical experiments were conducted for different types of distorted signals using different methods of setting weight coefficients. As a result, a significant real speed up of parallel algorithms was obtained, which confirmed their high efficiency. The results obtained can be used during the preliminary processing of large data arrays in many subject areas using modern software and hardware tools.

Key words: digital filtering, pyramids method, parallel algorithm, autonomous branches, limited parallelism, speed up of computations, multi-core computer.

Отримано 4 07 2025 р.