УДК 004.052.42
DOI https://doi.org/10.32838/TNU-2663-5941/2020.6-1/28

Шкарупило В.В.
Інститут проблем моделювання в енергетиці імені Г.Є. Пухова Національної академії наук України

Чемерис О.А.
Інститут проблем моделювання в енергетиці імені Г.Є. Пухова Національної академії наук України

Душеба В.В.
Інститут проблем моделювання в енергетиці імені Г.Є. Пухова Національної академії наук України

Кудерметов Р.К.
Національний університет «Запорізька політехніка»

ДОСЛІДЖЕННЯ МУЛЬТИПОТОЧНОЇ РЕАЛІЗАЦІЇ МЕТОДУ ПЕРЕВІРКИ НА МОДЕЛІ ДЛЯ ТЕМПОРАЛЬНОЇ ЛОГІКИ ДІЙ[1]

У наш час мультиядерні обчислювальні системи широко застосовуються при вирішенні науково-прикладних задач різноманітного характеру. Одним із актуальних сценаріїв такого застосування є розв'язання задачі формальної верифікації, орієнтованої на перевірку коректності проєктних рішень, одержуваних, зокрема, під час розроблення систем критичного призначення. Предметна сфера систем критичного призначення є особливо показовою, оскільки канонічні засоби валідації одержуваних рішень – тестування, імітаційне моделювання – не дозволяють охопити усі потенційно можливі сценарії функціонування системи. Дієвим рішенням у цьому разі є використання методів перевірки на моделі, що застосовуються по відношенню не до системи безпосередньо, а до відповідної формальної моделі – специфікації. Застосування названих методів передбачає повний перебір простору станів системи переходів, синтезованої згідно з формальною специфікацією. Такий підхід характеризується експоненційним зростанням простору станів залежно від числа змінних станів. Відповідні супутні часові витрати на верифікацію можна істотно знизити шляхом залучення мультипоточної реалізації методу перевірки на моделі. Залежно від структури специфікації та особливостей реалізації методу перевірки на моделі, корисний ефект від введення паралелізму може істотним чином варіюватися. Це своєю чергою породжує питання доцільності такого введення для того чи іншого випадку. Дослідженню цього питання і присвячено цю роботу, що дозволить сформулювати рекомендації щодо застосування мультипоточної реалізації методу перевірки на моделі для заданого сценарію предметної області.

У якості досліджуваного використано метод TLC, що знайшов широке застосування в індустрії. Розглянуто дві альтернативні реалізації методу, що будуються на перевірці елементів простору станів системи переходів, відповідно, шляхом обходів у ширину і глибину теорії графів. Сформульовано рекомендації до мультипоточного прикладного застосування реалізацій методу.

Ключові слова: *BFS, DFS, TLC, мультипоточність, перевірка на моделі, верифікація, система критичного призначення.*

Постановка проблеми. У наші дні паралельні обчислення широко застосовуються для сприяння вирішенню поставлених науково-технічних задач різного характеру, що включають, зокрема, задачі енергетики, аерокосмічної галузі тощо. Відповідне сприяння проявляється у скороченні супутніх часових витрат [1]. Одержуваний при цьому корисний ефект істотним чином залежить як від програмно-апаратних можливостей застосовуваної обчислювальної платформи, так і від специфіки вирішуваної задачі – її придатності до розпаралелювання. Окремої уваги при цьому заслуговують актуальні сьогодні задачі, вирішувані формальними методами, а саме методами перевірки на моделі [2]. Предметною областю відповідних задач є, зокрема, системи критично призначення – системи, збої і відмови у роботі яких можуть призвести до наслідків критичного масштабу і характеру. Процес розроблення названих систем регламентується, зокрема, наступними стандартами функціональної безпеки: IEC

[1] Дослідження виконано у межах науково-дослідної роботи № 0120U102683 «Розроблення спеціалізованих комп'ютерних технологій моделювання та опрацювання оперативної інформації в задачах енергетики», що проводиться відділом математичного та комп'ютерного моделювання Інституту проблем моделювання в енергетиці ім. Г.Є. Пухова НАН України.

61508 [3], ISO 26262:2018 [4], де рекомендується застосовувати формальні методи. Підхід до застосування названих методів на прикладі космічної галузі вже було висвітлено [5].

Окремої уваги заслуговує метод верифікації TLC (TLA Checker), що базується на використанні темпоральної логіки дій TLA (Temporal Logic of Actions) Л. Лемпорта (Leslie Lamport) – лауреата премії Тюрінга [6]. Метод знайшов широке застосування в індустрії: під час перевірки проєктних рішень вебсервісів компанії Amazon [7], у атомній енергетиці – під час перевірки логіки роботи системи захисту атомного реактору [8], під час проєктування системи керування рухом залізничного транспорту тощо [9]. Попри чисельні публікації у напрямі дослідження і застосування названого методу, питання мультипоточної імплементації альтернативних реалізацій методу у контексті одержуваного від цього корисного ефекту лишається невисвітленим. Розвиткові цього напряму і присвячено цю роботу.

Аналіз останніх досліджень і публікацій. Результати попередніх досліджень показали, що корисний ефект від введення мультипоточності до чисельного вирішення різноманітних науково-прикладних задач може бути доволі вагомим [1]. Підтвердженням обґрунтованості таких кроків є підтримка мультипоточності реалізаціями різноманітних методів перевірки на моделі. Показовим є метод TLC: для обчислювальної системи із 384 процесорами було одержано коефіцієнт прискорення, близький до 328 [10]. Своєю чергою, результати попередніх досліджень названого методу, здобуті авторами даної роботи, показують, що доцільність застосування кожної із двох альтернативних реалізацій методу – на основі обходу у ширину (BFS, Breadth-first Search) і у глибину (DFS, Depth-first Search) теорії графів – істотним чином залежить як від числа змінних станів в основі формальної специфікації (ФС), до якої застосовується метод, так і від структури специфікації [11; 12]. Більше того, було показано, що DFS-реалізація методу TLC характеризується істотно гіршою просторовою складністю вирішення задачі формальної верифікації (ФВ), у порівнянні із альтернативною BFS-реалізацією [13]. Питання оцінювання корисного ефекту від залучення мультипоточності залежно від застосовуваної реалізації методу, і, як результат, обґрунтування доцільності такого залучення по відношенню до заданого сценарію предметної області при цьому потребує подальшого розвитку.

Постановка завдання. У роботі ставиться завдання оцінити корисний ефект від залучення мультипоточності до BFS- та DFS-реалізацій методу TLC. Як відповідний показник застосовується коефіцієнт прискорення:

$$\alpha = t_1 / t_{tc} , \qquad (1)$$

де t_1 – час, витрачений на ФВ методом TLC за однопоточної реалізації, t_{tc} – за мультипоточної реалізації.

Під час вирішення поставленого завдання досліджуються дві альтернативні реалізації методу – на основі BFS- і DFS-обходів. Як предмет дослідження розглядається програмний складник системи критичного призначення, застосовуваної у космічній галузі. Це дозволить сформулювати рекомендації стосовно прикладного мультипоточного застосування кожної із реалізацій методу.

Виклад основного матеріалу дослідження. Як предметну сферу розглянемо космічну галузь, де предметом дослідження є система критичного призначення, представлена програмним складником бортового цифрового обчислювального комплексу (БЦОК). Відповідний сценарій предметної сфери полягає у дослідженні алгоритму контролю вихідного стану регістрів модуля пристрою введення / виведення (ПВВ) БЦОК. Фрагмент відповідної блок-схеми алгоритму подано на рис. 1.

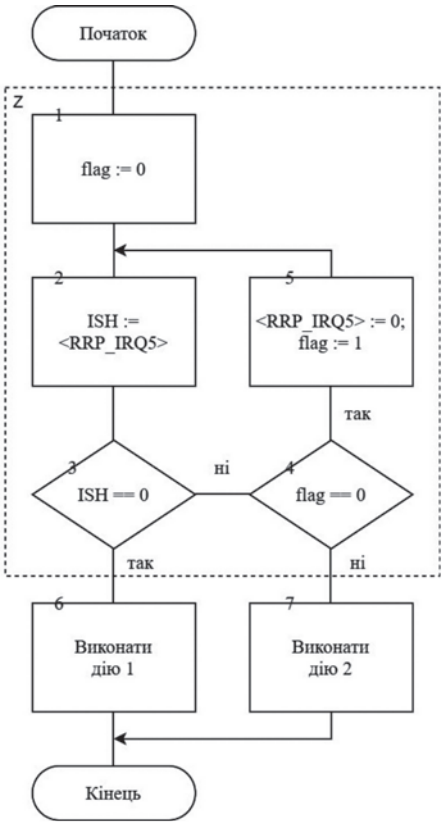

Рис. 1. Фрагмент блок-схеми алгоритму контролю вихідного стану регістрів ПВВ

На рис. 1 пунктирною областю (позначено як Z) окреслено фрагмент блок-схеми, що повторюється 15 разів: $5 \cdot 15 = 75$ блоків. Блоки 6 і 7 при цьому демонструють альтернативні заключні кроки алгоритму. Після 15 зазначених фрагментів вони одержують, відповідно, номери 76 і 77. Кожен наступний фрагмент відрізняється від попереднього діями, виконуваними у блоках 2 і 5, де скобками < i > позначено опосередковану адресацію.

На рис. 1 фігурують три змінних стану: "flag", "ISH", "RRP_IRQ5". Окрім того, аби імітувати виконання / невиконання блоків 6 і 7, введено додаткову змінну "done": $s(done) \in \{0,1,2\}$, де $s \in S$ – стан системи переходів (СП), представленої структурою Кріпке [2].

Експериментальні дослідження проведено на програмно-апаратній платформі наступної конфігурації: середовище виконання – Java Runtime Environment (64 bit, build 1.8.0_251-b08); версія реалізації методу TLC – 2.14 (від 10 липня 2019 р.); центральний процесор – 4 ядра, 8 потоків, частота – 3,8 ГГц; обсяг ОП – 16 ГБ.

Методика проведення досліджень:

– провести ФВ ФС, фрагмент якої подано на рис. 1, спочатку BFS-, а потім – DFS-реалізацією методу TLC, що дасть змогу визначити глибину обходу СП – необхідний параметр для застосування DFS-реалізації методу;

– виміряти та порівняти часові витрати на використання одно- і мультипоточної реалізацій методу, розрахувавши коефіцієнти прискорення для мультипоточних реалізацій.

Характеристики ФС, використовуваної у якості вихідних даних для методу: число змінних станів – 15; глибина обходу простору станів СП (у вершинах) – 42; загальне число станів СП – $|S| = 184302$.

Здобуті результати подано у табл. 1, де t_{BFS} – часові витрати на ФВ BFS-реалізацією методу TLC, t_{DFS} – DFS-реалізацією методу. Кожне табличне значення є середнім арифметичним 10 замірів.

Із табл. 1 видно, що введення мультипоточності в BFS-реалізацію методу супроводжується незначним ефектом – близько 26,2% для 8 потоків. При цьому для розглянутого випадку, в абсолютному вираженні BFS-реалізація методу є від 12,549 до 6,739 разів ефективнішою, у порівнянні із DFS-альтернативою. Це можна обґрунтувати як специфікою структури досліджуваної ФС, так і особливостями як алгоритмічного складника, так і програмних реалізацій відповідних обходів. Відомо, що для BFS-обходу зазвичай застосовується структура даних «черга», що функціонує згідно з правилами FIFO (First In First Out), а для DFS-обходу – стек, де діє правило LIFO (Last In First Out). Асимптотична обчислювальна складність для обох варіацій є однаковою [14].

Графічне подання здобутих даних представлене на рис. 2, де показано залежність значень коефіцієнтів прискорення від кількості обчислювальних потоків.

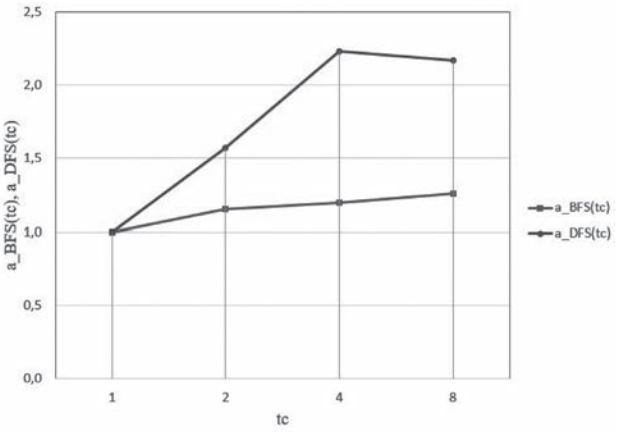

Рис. 2. Графік залежності значень коефіцієнтів прискорення від кількості обчислювальних потоків

Під час побудови рис. 2 застосовано кусочно-лінійну інтерполяцію.

З рис. 2 видно, що застосування мультипоточної DFS-реалізації методу супроводжується кращим ефектом: для $tc = 4$ одержано прискорення $\alpha = 2,230$. При цьому для $tc = 8$ здобуто гірший результат – $\alpha = 2,172$. Це можна пояснити специфікою тестової платформи – чотири обчислювальних

Таблиця 1

Результати дослідження мультипоточних реалізацій

№ з/п	tc	Реалізація методу				t_{DFS} / t_{BFS}
		BFS		DFS		
		t_{BFS}, с	α	t_{DFS}, с	α	
1	1	3,552	1,000	44,575	1,000	12,549
2	2	3,072	1,156	28,384	1,570	9,240
3	4	2,966	1,198	19,989	2,230	6,739
4	8	2,815	1,262	20,526	2,172	7,292

ядра з реалізацією технології "Simultaneous Multithreading", що дозволяє одночасно виконувати два програмних потоки на одному обчислювальному ядрі процесора [1].

Отже, введення мультипоточності до DFS-реалізації методу TLC супроводжується істотно більшим корисним ефектом, у порівнянні із альтернативною BFS-реалізацією. Такий крок є обґрунтованим у разі співставних часових витрат для однопоточних BFS- і DFS-реалізацій методу.

Висновки. Таким чином, у роботі було проведено дослідження мультипоточної реалізації методу перевірки на моделі TLC на прикладі BFS- і DFS-варіацій. Здобуті результати полягають у наступному:

1. На прикладі сценарію розглянутої предметної області показано, що DFS-реалізація методу TLC забезпечує ліпші значення показника прискорення під час застосування мультипоточності. Найбільше значення названого показника було зафіксовано для чотирьохпоточної реалізації методу. Воно склало 2,230. При цьому для восьмипоточної реалізації методу було одержано гірший результат – 2,172.

2. У разі BFS-реалізації методу найліпший корисний ефект від залучення мультипоточності склав істотно менше значення – 1,262 – для восьмипоточної реалізації методу.

Отже, у разі співставності часових витрат для однопоточних BFS-і DFS-реалізацій методу TLC, рекомендується застосовувати мультипоточність саме для DFS-реалізації методу.

Озвучені результати можна пояснити наступними чинниками: специфікою програмної реалізації методів обходу вершин графу-системи переходів під час проведення перевірки на моделі; відмінністю концептуальних складників в основі методів обходу; обмеженнями апаратного складника тестової платформи (чотири обчислювальних ядра із підтримкою мультипоточності); структурою блок-схеми алгоритму, залученої у якості вихідних даних. У зв'язку із цим, подальші дослідження спрямовано на узагальнення рекомендацій стосовно доцільності введення мультипоточності для тієї чи іншої реалізації методу TLC, враховуючи структуру вихідних даних.

Список літератури:

1. Тіменко А.В., Шкарупило В.В., Скрупський С.Ю., Смолій В.В. Дослідження шляхів підвищення пропускної спроможності підсистеми пам'яті сучасної обчислювальної системи. *Вчені записки ТНУ імені В.І. Вернадського*. 2020. Том 31 (70), Ч. 1, № 2. С. 208–212. DOI: https://doi.org/10.32838/2663-5941/2020.2-1/32.
2. Clarke E.M., Grumberg O., Kroening D., Peled D., Veith H. Model checking: 2nd ed. Massachusetts: The MIT Press, 2018.
3. IEC 61508 Edition 2.0. Functional safety of electrical/electronic/programmable electronic safety-related systems. [Approved: April 2010]. URL: https://www.iec.ch/functionalsafety/standards/page2.htm. (Accessed: 19.11.2020).
4. ISO 26262:2018. Road vehicles. Functional safety. Part 1: Vocabulary. [Published: December 2018]. URL: https://www.iso.org/standard/68383.html (Accessed: 19.11.2020).
5. Шкарупило В.В., Євдокимов В.Ф., Душеба В.В. Застосування формальних методів для перевірки систем критичного призначення. *Вчені записки ТНУ імені В.І. Вернадського*. 2019. Том 30 (69), Ч. 1, № 6. С. 188–193.
6. Lamport L. *Specifying systems: The TLA+ language and tools for hardware and software engineers*. Boston : Addison-Wesley, 2002. 382 p.
7. Newcombe C., Rath T., Zhang F., Munteanu B., Brooker M., Deardeuff M. How Amazon web services uses formal methods. *Communications of the ACM*. 2015. Vol. 58, No. 4. P. 66-73. DOI: https://doi.org/10.1145/2699417
8. Pakonen A., Buzhinsky I. Verification of fault tolerant safety I&C systems using model checking. *Industrial Technology, ICIT 2019*: 2019 IEEE International Conference (Melbourne, Australia, 2019). 2019. P. 969–974. DOI: https://doi.org/10.1109/ICIT.2019.8755014
9. Resch S., Paulitsch M. Using TLA+ in the Development of a Safety-Critical Fault-Tolerant Middleware. *Software Reliability Engineering Workshops* : Proc. 2017 IEEE International Symposium (Toulouse, France, 23–26 October 2017). P. 146-152. DOI: https://doi.org/10.1109/ISSREW.2017.43.
10. Lamport L. Checking a multithreaded algorithm with +CAL. *Distributed Computing, DISC'06* : Proceedings of the 20th international conference (Stockholm, Sweden, September 18-20, 2006). P. 151-163. DOI: https://doi.org/10.1007/11864219_11.
11. Shkarupylo V. V., Tomičić I., Kasian K. M. The investigation of TLC model checker properties. *Journal of Information and Organizational Sciences*. 2016. Vol. 40, No. 1. P. 145–152.
12. Shkarupylo V. V., Tomičić I., Kasian K. M., Alsayaydeh J. A. J. An Approach to increase the Effectiveness of TLC Verification with Respect to the Concurrent Structure of TLA+ Specification. *International Journal of Software Engineering and Computer Systems*. 2018. Vol. 4, No. 1. P. 48–60. DOI: https://doi.org/10.15282/ijsecs.4.1.2018.4.0037

13. Шкарупило В.В., Чемерис О.А., Душеба В.В. Оцінювання просторової складності задачі формальної верифікації, вирішуваної методом перевірки на моделі. *Вчені записки ТНУ імені В.І. Вернадського*. 2020. Том 31 (70) № 5. С. 147–151.

14. Cormen T.H., Leiserson C.E., Rivest R.L., Stein C. Introduction to algorithms: 3rd ed. Cambridge, Massachusetts: The MIT Press, 2009. 1320 p.

Shkarupylo V.V., Chemerys O.A., Dusheba V.V., Kudermetov R.K.

RESEARCH ON MULTITHREADED IMPLEMENTATION OF MODEL CHECKING METHOD FOR TEMPORAL LOGIC OF ACTIONS

Nowadays, multicore computing systems are broadly applied while diverse scientific-practical tasks resolving. One of the topical scenarios of such application is formal verification task solving – to check the correctness of project solutions obtained, in particular, during the development of safety-critical systems. The domain of safety-critical systems is among the demonstrative ones, because canonical instruments, e.g., testing, simulation, do not provide an opportunity to encompass all possible scenarios of system functioning. An effective solution in this case is model checking techniques usage. Named techniques are intended to be applied not to the system directly, but with respect to corresponding formal model, i.e. specification. Practical implementation of these methods implies traversing through the elements of total set of states of transition system, generated from specification. Such approach is coupled with an exponential growth of state space from the number of state variables. Corresponding verification-related time costs can be significantly reduced – by way of multithreaded implementation of model checking technique. Depending on specification structure and peculiarities of model checking technique implementation, the effect of multithreading bringing can vary significantly. This situation, in its turn, raises the question on the expediency of such step with respect to a particular scenario. Current work is devoted to a research in this direction – to formulate the recommendations to multithreaded implementation of model checking technique with respect to a specified applicability domain.

Because of being broadly adopted in industry, the TLC method has been examined in research. Two alternative method implementations grounding on checking the elements of state space by way of Breadth-first- and Depth-first-searches have been considered. Recommendations to practical multithreaded application of these implementations have been formulated.

Key words: *BFS, DFS, TLC, multithreading, model checking, verification, safety-critical system.*