DOI: https://doi.org/10.15276/ict.02.2025.29
УДК 04.005

Метод побудови GL-моделей складних небазових відмовостійких багатопроцесорних систем

Романкевич Віталій Олексійович[1)]
Д-р техніч. наук, професор, завідувач каф. Системного програмування
та спеціалізованих комп'ютерних систем
ORCID: https://orcid.org/0000-0003-4696-5935; zavkaf@scs.kpi.ua. Scopus Author ID: 57193263058
Морозов Костянтин В'ячеславович[1)]
Канд. техніч. наук, доцент каф. Системного програмування
та спеціалізованих комп'ютерних систем
ORCID: https://orcid.org/0000-0003-0978-6292; mcng@ukr.net. Scopus Author ID: 57222509251
Галицький Даніїл Володимирович[1)]
Аспірант каф. Системного програмування та спеціалізованих комп'ютерних систем
ORCID: https://orcid.org/0009-0004-4421-3443; zipper135401@gmail.com. Scopus Author ID: 58553487600
Єрмоленко Ігор Андрійович[1)]
Аспірант каф. Системного програмування та спеціалізованих комп'ютерних систем
ORCID: https://orcid.org/0009-0008-5298-4888; yermolenkomail@gmail.com.
[1)] Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»,
пр. Перемоги, 37. Київ, 03056, Україна

АНОТАЦІЯ

Робота присвячена методу побудови GL-моделей поведінки складних небазових відмовостійких багатопроцесорних систем в потоці відмов. Такі моделі, зокрема, можуть бути застосовані для розрахунку параметрів надійності останніх методами, що базуються на проведенні статистичних експериментів. Розглядаються системи, поведінка яких може бути описана за допомогою ряду відносно простих умов, таких, для яких доступні методи побудови відповідних GL-моделей (наприклад, наявність не більше ніж заданої кількості відмов серед певної підмножини процесорів системи). Передбачається, що роботоздатність системи забезпечується лише за одночасного виконання всіх цих умов. Зокрема, до таких систем належать так звані ієрархічні системи, що складаються з кількох підсистем, кожна з яких має певний рівень відмовостійкості, а також системи, які містять у своєму складі спеціалізовані процесори різних типів. Відповідно до запропонованого методу спершу будуються моделі для кожної з умов, після чого вони об'єднуються у єдину GL-модель шляхом послідовного об'єднання їх графів через деякі вершини (ці вершини об'єднуються в одну, а решта вершин та ребра графів копіюються). При цьому, послідовність моделей, а також вершини для об'єднання графів можуть бути обрані довільним чином. Завдяки цьому, за необхідності, може бути обрана більш зручна структура графа побудованої GL-моделі. Наведено приклад, що демонструє застосування запропонованого методу. При цьому розглянуто два варіанти вибору послідовності об'єднання моделей, а також вершин, через які об'єднуються їхні графи. Проведені експерименти підтверджують, що незважаючи на різну структуру графів отриманих моделей, їх поведінка на однакових вхідних векторах дійсно збігається і відповідає поведінці в потоці відмов відмовостійкої багатопроцесорної системи, для якої вони були побудовані. Відзначено також, що запропонований метод не накладає жодних обмежень на способи побудови GL-моделей для кожної з умов (можуть бути одночасно поєднані моделі різних типів), умови не обов'язково мають відповідати базовим системам (тобто, відмові не більш ніж деякої кількості довільних процесорів), а графи відповідних їм моделей можуть не бути циклічними.

Ключові слова: відмовостійкі багатопроцесорні системи; GL-моделі; небазові системи; поведінка в потоці відмов; системи керування; ієрархічні системи; оцінка надійності; статистичні експерименти

Актуальність. В останні десятиріччя автоматизація різноманітних процесів набуває дедалі більшого поширення [1, 2]. Це дозволяє з одного боку звільнити людину від необхідності виконання монотонної роботи, а з іншого – знизити вплив людського фактору на виконання таких завдань. Деякі завдання взагалі не можуть бути виконані людиною через обмеженість її фізіологічних можливостей (наприклад, швидкість реакції), або ж через небажаність/неможливість її безпосереднього перебування на об'єкті (наприклад, космічні місії, військові безпілотні системи тощо). Одним із ключових вузлів таких об'єктів та систем стає їхня система керування (СК), що на базі даних, отриманих від різноманітних датчиків формує відповідні керуючі сигнали для виконавчих пристроїв [1, 2].

Існують системи, неправильна робота яких може призводити до значних негативних наслідків, таких як суттєві матеріальні втрати, загроза здоров'ю або життю людей, загроза

добробуту та стабільності держави тощо [3, 4, 5]. До таких систем загалом, і, до їхніх СК зокрема висуваються підвищені вимоги щодо надійності. Крім того, часто керування ними вимагає вирішення задач, що мають значну розрахункову складність. Тому системи керування такими об'єктами доцільно реалізовувати на базі так званих відмовостійких багатопроцесорних систем (ВБС), які складаються з великої кількості процесорів (що, зокрема, дозволяє досягти високих рівнів продуктивності), та залишаються в роботоздатному стані у випадку виходу з ладу деяких із них (що забезпечує високі показники надійності) [6, 7, 8].

Розробнику ВБС важливо мати можливість оцінки рівня надійності створюваної ним системи. Ця задача не завжди є простою, зокрема, через те що ВБС для СК можуть мати складну та неоднорідну структуру, складатися з процесорів різних типів, які вирішують специфічні задачі тощо.

Методи розрахунку параметрів надійності ВБС можна умовно поділити на дві групи [9, 10]. Методи першої групи базуються на побудові складних аналітичних виразів, що з одного боку часто дозволяє оцінити параметри надійності ВБС із високою точністю, а з іншого – ці методи є неуніверсальними: для нового типу систем зазвичай доводиться створювати новий метод [11, 12, 13, 14, 15, 16]. Методи другої групи дозволяють розрахувати параметри надійності ВБС шляхом проведення статистичних експериментів з моделями її поведінки в потоці відмов [17, 18, 19, 20]. Вони є універсальними, проте, точність розрахунку в загальному випадку залежить від кількості проведених експериментів. Тому зменшення складності експерименту (зокрема, за рахунок спрощення моделі) дозволяє з одного боку зменшити час розрахунків, а з іншого – підвищити їх точність.

У ролі моделей поведінки ВБС в потоці відмов можуть використовуватися GL-моделі [20, 21], що поєднують в собі властивості графів та булевих функцій. GL-модель являє собою неорієнтований граф, кожному ребру якого відповідає деяка булева реберна функція, що залежить від так званого вектора стану системи – булевого вектора, кожен елемент якого відповідає стану одного з процесорів системи (1 – процесор є роботоздатним, 0 – відмовив). Якщо реберна функція приймає нульове значення, відповідне ребро виключається з графа. Зв'язність графа на деякому векторі відповідає стану системи в потоці відмові для заданої конфігурації станів її процесорів: зв'язний граф – система є роботоздатною, незв'язний граф – система є несправною. Побудова GL-моделей ВБС може виконуватися різними способами [22, 23, 24, 25].

Окремої уваги заслуговують так звані базові системи, здатні зберігати роботоздатність за умови відмови не більше ніж певної кількості своїх процесорів. Базова ВБС, яку позначатимемо як $K(m, n)$ складається з n процесорів та є стійкою до виходу з ладу не більш ніж m будь-яких із них. GL-моделі базових систем можуть бути побудовані на основі циклічних графів [22, 23], що, зокрема, дозволяє зменшити складність процедури оцінки їх зв'язності. Проте ВБС керування досить часто бувають небазовими, тобто, такими, що є стійкими до деяких відмов певної кратності і одночасно нестійкими до інших відмов тієї ж кратності.

Побудова GL-моделі небазової системи може відбуватися шляхом модифікації деякої базової системи, зокрема, в результаті зміни структури її графа (наприклад, проведення додаткових ребер) і/або зміни виразів її реберних функцій. Такий підхід є особливо зручним у випадках, коли ВБС не суттєво відрізняється від базової, тобто, в більшості випадків поводить себе як базова, і лише в деяких окремих ситуаціях (при певних комбінаціях справних та несправних процесорів) її поведінка відрізняється: вона виходить з ладу або залишається роботоздатною [24].

Проте деякі реальні ВБС можуть суттєво відрізнятися від базових. Наприклад, для кожного типу процесорів в системі може існувати своя максимально допустима кратність відмов. Крім того, можуть мати місце і додаткові обмеження, наприклад, максимально допустима загальна кратність відмов, або ж максимально допустима кратність відмов для

певної підмножини процесорів. Зокрема, це стосується систем, що складаються з декількох різних підсистем: кожна підсистема може поводити себе як базова, але поведінка системи в цілому при цьому значно відрізнятиметься від базової. В [25] запропоновано метод побудови GL-моделей для таких ієрархічних систем. Проте, отримані цим методом моделі також є ієрархічними і, відповідно, досить складними: спершу доводиться виконувати розрахунок для кількох допоміжних моделей, після чого виконується розрахунок для GL-моделі системи.

Метою дослідження є створення методу побудови GL-моделей складних небазових ВБС спеціального виду, а саме таких, поведінка яких у потоці відмов може бути описана за допомогою множини відносно простих умов (наприклад, вихід з ладу не більш ніж деякої кількості процесорів із певної підмножини процесорів системи), для кожної з яких тим чи іншим способом може бути побудована окрема GL-модель. При цьому передбачається, що роботоздатність системи забезпечується лише за одночасного виконання всіх зазначених умов.

Відповідно до запропонованого методу модель такої системи може бути побудована наступним чином.

1. Побудувати окремі GL-моделі для кожної з умов (позначимо їх $\Omega = \{M_1, M_2, \ldots, M_k\}$).
2. Вилучити довільну модель M_i з множини Ω, $M = M_i$ – модель, що будується.
3. Якщо множина Ω – порожня, перейти до п. 8.
4. Вилучити довільну модель M_j з множини Ω.
5. Обрати довільну вершину α в графі моделі M та довільну вершину β в графі моделі M_j.
6. Об'єднати графи моделей M та M_j через вершини α та β (тобто, вершини α та β графів моделей M та M_j перетворити на одну спільну вершину, а решту вершин та всі ребра обидвох моделей скопіювати), в результаті чого буде отримано нову модель M.
7. Перейти до п. 3.
8. Модель M і є GL-моделлю системи, яку потрібно було побудувати.

Легко помітити, що отримана запропонованим методом GL-модель показуватиме роботоздатний стан системи тоді і тільки тоді, коли виконуватиметься кожна з умов, тобто коли графи кожної з моделей $M_1, M_2, \ldots, M_k$ залишатимуться зв'язними. Дійсно, з одного боку зв'язність кожного з графів моделей $M_1, M_2, \ldots, M_k$ призводить і до зв'язності графа моделі M. З іншого боку, об'єднання графів моделей через спільну вершину не призводить до появи додаткових шляхів, які б дозволили зберегти зв'язність графа моделі M у випадку втрати її хоча б одним із графів моделей $M_1, M_2, \ldots, M_k$.

Варто також відзначити, що метод дозволяє вибирати порядок об'єднання моделей, а також вершини для об'єднання їх графів довільним чином. Тобто, для однієї і тієї ж ВБС може бути отримана велика кількість альтернативних варіантів GL-моделі. Завдяки цьому, зокрема, може бути отримана більш зручна структура її графа (наприклад, для швидшого виконання процедури оцінки зв'язності, або для простішої подальшої модифікації моделі у разі потреби).

В якості прикладу побудуємо GL-модель системи, що містить 12 процесорів та є роботоздатною тоді й лише тоді, коли одночасно виконуються наступні умови:

1. Серед процесорів 1…6 є не більш ніж 2 несправних.
2. Серед процесорів 7…10 є не більш ніж 1 несправний.
3. Серед процесорів 4…8 є не більш ніж 1 несправний.
4. Серед процесорів 1, 3, 7, 8, 9 є не більш ніж 2 несправних.
5. В усій системі є не більш ніж 3 несправних процесори.

Кожна з наведених вище умов відповідає поведінці у потоці відмов деякої базової системи. Побудуємо GL-моделі для кожної з них, а саме модель $K_1(2, 6)$ для множини процесорів 1…6, модель $K_2(1, 4)$ для множини процесорів 7…10, модель $K_3(1, 5)$ для множини процесорів 4…8, модель $K_4(2, 5)$ для множини процесорів 1, 3, 7, 8, 9 та модель $K_5(3, 12)$ для множини процесорів 1…12. Елементи вектора стану системи позначатимемо як

x_i, де i – номер відповідного процесора. Для побудови цих моделей скористаємося методом, наведеним у [22].

Модель $K_1(2, 6)$ базується на циклічному графі з 5 вершинами (позначеними як α_1, α_2, α_3, α_4, α_5) та 5 ребрами (рис. 1). Вона матиме наступні реберні функції:

$f_1^1 = x_1 \vee x_2$;
$f_2^1 = x_1 x_2 \vee x_3$;
$f_3^1 = x_1 x_2 x_3 \vee x_4 x_5 x_6$;
$f_4^1 = x_4 \vee x_5$;
$f_5^1 = x_4 x_5 \vee x_6$.

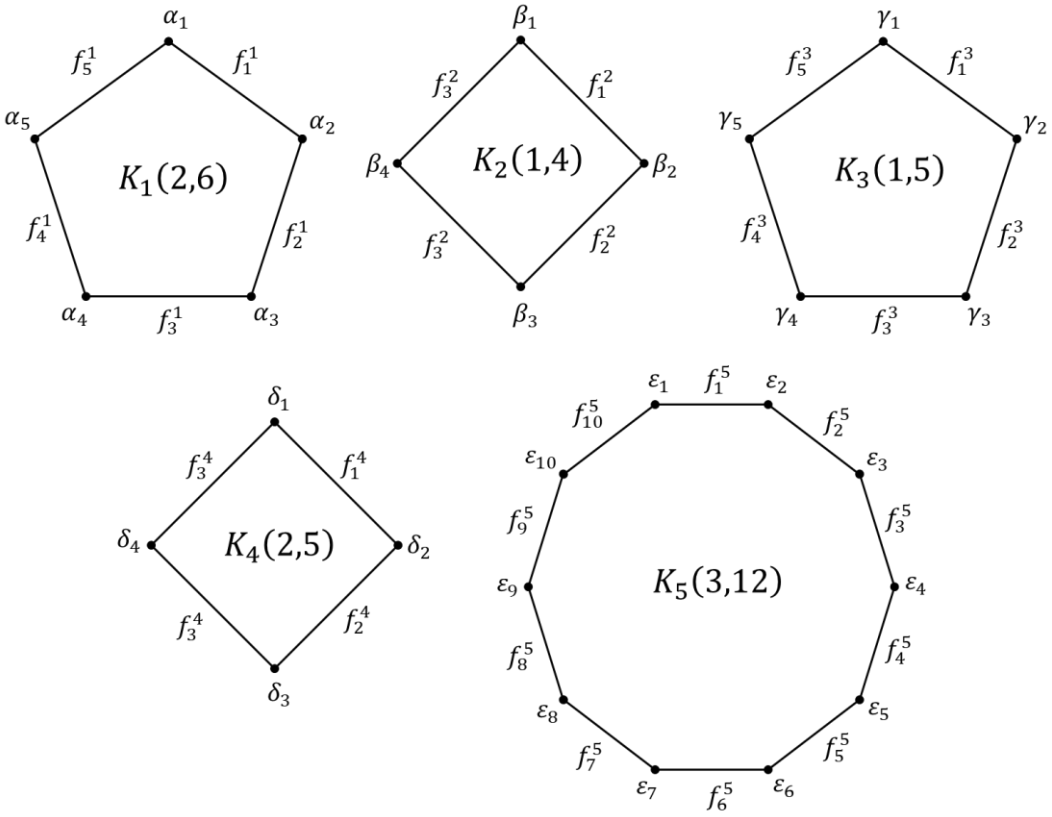

Рис. 1. **GL-моделі $K_1(2, 6)$, $K_2(1, 4)$, $K_3(1, 5)$, $K_4(2, 5)$ та $K_5(3, 12)$**

Модель $K_2(1, 4)$ ґрунтується на циклічному графі з 4 вершинами (β_1, β_2, β_3, β_4) і 4 ребрами (рис. 1). Її реберні функції мають вигляд:

$f_1^2 = x_7$;
$f_2^2 = x_8$;
$f_3^2 = x_9$;
$f_4^2 = x_{10}$.

Модель $K_3(1, 5)$ побудована на циклічному графі з 5 вершинами (γ_1, γ_2, γ_3, γ_4, γ_5) і 5 ребрами (рис. 1). Реберні функції цієї моделі матимуть наступний вигляд:

$f_1^3 = x_4$;
$f_2^3 = x_5$;
$f_3^3 = x_6$;
$f_4^3 = x_7$;
$f_5^3 = x_8$.

Модель $K_4(2, 5)$, базується на циклічному графі з 4 вершинами (δ_1, δ_2, δ_3, δ_4) і 4 ребрами (рис. 1). Вона матиме наступні реберні функції:

$f_1^4 = x_1 \vee x_3$;
$f_2^4 = x_1 x_3 \vee x_7$;

$f_3^4 = x_1x_3x_7 \vee x_8x_9$;
$f_4^4 = x_8 \vee x_9$.

Нарешті модель $K_5(3, 12)$ ґрунтується на циклічному графі з 10 вершинами (ε_1, ε_2, ε_3, ε_4, ε_5, ε_6, ε_7, ε_8, ε_9, ε_{10}) і 10 ребрами (рис. 1). Її реберні функції матимуть вигляд:

$f_1^5 = x_1 \vee x_2 \vee x_3$;
$f_2^5 = (x_1 \vee x_2)(x_1x_2 \vee x_3) \vee x_4x_5x_6$;
$f_3^5 = x_1x_2x_3 \vee (x_4 \vee x_5)(x_4x_5 \vee x_6)$;
$f_4^5 = x_4 \vee x_5 \vee x_6$;
$f_5^5 = (x_1 \vee x_2)(x_1x_2 \vee x_3)(x_1x_2x_3 \vee x_4x_5x_6)(x_4 \vee x_5)(x_4x_5 \vee x_6) \vee x_7x_8x_9x_{10}x_{11}x_{12}$;
$f_6^5 = x_1x_2x_3x_4x_5x_6 \vee (x_7 \vee x_8)(x_7x_8 \vee x_9)(x_7x_8x_9 \vee x_{10}x_{11}x_{12})(x_{10} \vee x_{11})(x_{10}x_{11} \vee x_{12})$;
$f_7^5 = x_7 \vee x_8 \vee x_9$;
$f_8^5 = (x_7 \vee x_8)(x_7x_8 \vee x_9) \vee x_{10}x_{11}x_{12}$;
$f_9^5 = x_7x_8x_9 \vee (x_{10} \vee x_{11})(x_{10}x_{11} \vee x_{12})$;
$f_{10}^5 = x_{10} \vee x_{11} \vee x_{12}$.

Далі для побудови GL-моделі системи, що розглядається, виконаємо послідовне об'єднання побудованих вище моделей. Наприклад, першою можемо обрати модель $K_1(2, 6)$, після чого об'єднаємо її із моделлю $K_2(1, 4)$ через вершини α_2 та β_4 (отриману в результаті об'єднання вершину позначимо ω_1). Далі отриману модель об'єднаємо із моделлю $K_3(1, 5)$ через вершини β_2 та γ_5 (відповідну вершину нової моделі позначимо ω_2). Після цього виконаємо об'єднання із моделлю $K_4(2, 5)$ через вершини α_4 та δ_1, позначивши отриману в результаті об'єднання вершину ω_3. Нарешті, об'єднаємо отриману модель із моделлю $K_5(3, 12)$ через вершини γ_4 та ε_2 (аналогічно до попередніх випадків відповідну вершину нової моделі позначимо ω_4). Отримана в результаті вищенаведених об'єднань GL-модель (рис. 2) відповідає поведінці в потоці відмов розглянутої системи, що було підтверджено проведеними з нею експериментами.

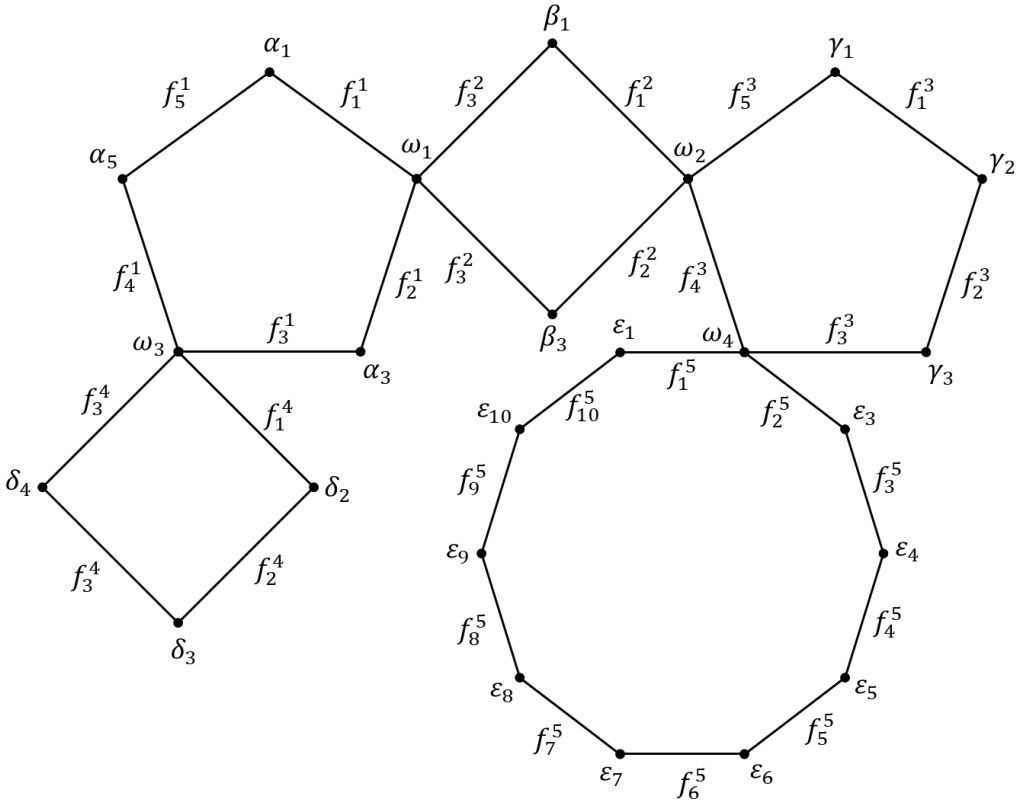

Рис. 2. **GL-модель системи, побудована запропонованим методом**

Відзначимо також, що, як вже було сказано вище, порядок об'єднання моделей, а також вершини через які об'єднуються їхні графи можуть обиратися довільним чином. Так,

наприклад, першою можна було б обрати модель $K_5(3, 12)$, після чого об'єднати її з моделлю $K_4(2, 5)$ через вершини ε_9 та δ_2 (ω_1). Далі виконати об'єднання із моделлю $K_2(1, 4)$ через вершини ε_4 і β_4 (ω_2), та з моделлю $K_1(2, 6)$ через вершини ε_7 і α_1 (ω_3). Та, зрештою, об'єднати її із моделлю $K_3(1, 5)$ через вершини ε_2 і γ_4 (ω_4). Отримана в результаті таких перетворень GL-модель (рис. 3) хоча й відрізняється від попередньої структурою свого графа, але, як показують експерименти, демонструє збіжну поведінку з попередньою моделлю на однакових вхідних векторах.

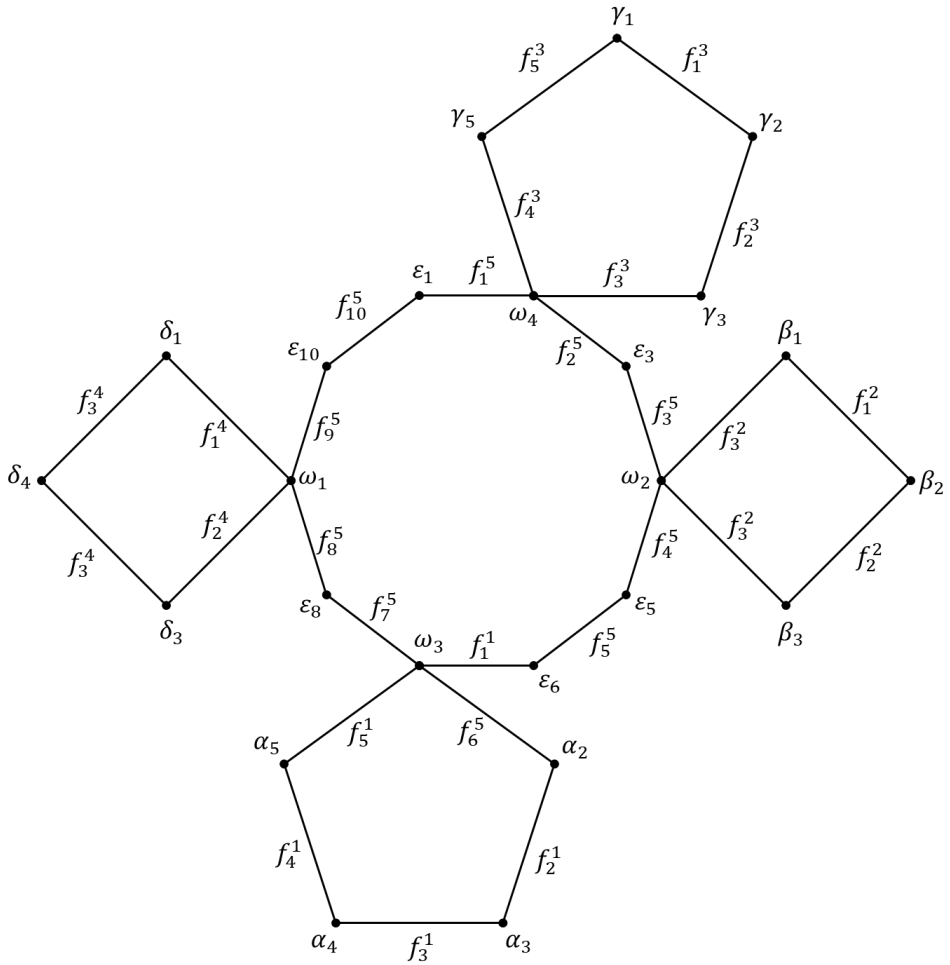

Рис. 3. **Альтернативний варіант GL-моделі розглянутої системи**

Варто також відзначити, що запропонований метод не накладає обмежень на способи побудови GL-моделей для кожної з умов. Так, наприклад, при його використанні можуть бути одночасно поєднані моделі базових систем, побудовані відповідно до [22] та [23]. Крім того, умови не обов'язково мають відповідати базовим системам (вихід з ладу не більш ніж певної кількості довільних процесорів), а графи моделей також не обов'язково повинні бути циклічними.

СПИСОК ЛІТЕРАТУРИ

1. Kotov D. O. "A generalized model of an adaptive information-control system of a car with multi-sensor channels of information interaction". *Applied Aspects of Information Technology.* 2021; 5 (1): 25–34. DOI: https://doi.org/10.15276/aait.05.2022.2.

2. Nazarova O. S., Osadchyy V. V., Rudim B. Y. "Computer simulation of the microprocessor liquid level automatic control system". *Applied Aspects of Information Technology.* 2023; 6 (2): 163-174. DOI: https://doi.org/10.15276/aait.06.2023.12.

3. Kovalev I. S., Drozd O. V., Rucinski A., Drozd M. O., Antoniuk V. V., Sulima Y. Y. "Development of Computer System Components in Critical Applications: Problems, Their Origins

and Solutions". *Herald of Advanced Information Technology. Publ. Nauka i Tekhnika.* 2020; 3 (4): 252–262. DOI: https://doi.org/10.15276/hait.04.2020.4.

4. Antoniuk V. V., Drozd M. O., Drozd O. B. "Power-oriented checkability and monitoring of the current consumption in FPGA projects of the critical applications". *Applied Aspects of Information Technology.* 2019; 2 (2): 105–114. DOI: https://doi.org/10.15276/aait.02.2019.2.

5. Drozd O., Ivanova O., Zashcholkin K., Romankevich V., Drozd Yu. "Checkability Important for Fail-Safety of FPGA-based Components in Critical Systems". *CEUR Workshop Proceedings.* 2021; 2853: 471–480.

6. Abbaspour A., Mokhtari S., Sargolzaei A., Yen, K. K. "A survey on active fault-tolerant control systems". *Electronics.* 2020; 9 (9): 1–23. DOI: https://doi.org/10.3390/electronics9091513.

7. Joshi H., Sinha N. K. "Adaptive fault tolerant control design for stratospheric airship with actuator faults". *IFAC-PapersOnLine.* 2022; 55 (1): 819–825. DOI: https://doi.org/10.1016/j.ifacol.2022.04.134.

8. Nedeljkovic J. N., Dosic S. M., Nikolic G. S. "A Survey of Hardware Fault Tolerance Techniques". *2023 58th International Scientific Conference on Information, Communication and Energy Systems and Technologies, ICEST 2023 – Proceedings.* 2023. p. 223–226. DOI: https://doi.org/10.1109/ICEST58410.2023.10187275.

9. Billinton R., Allan, R. N. "Reliability Evaluation of Engineering Systems. Concepts and Techniques". *Springer New York.* 1992. DOI: https://doi.org/10.1007/978-1-4899-0685-4.

10. Kuo W., Zuo M. "Optimal Reliability Modeling". *John Willey & Sons*. 2002.

11. Hu B., Seiler P. "Pivotal decomposition for reliability analysis of fault tolerant control systems on unmanned aerial vehicles". *Reliability Engineering & System Safety.* 2015; 140: 130–141. DOI: https://doi.org/10.1016/j.ress.2015.04.005.

12. Lu J., Yi H., Li X., Balakrishnan N. "Joint Reliability of Two Consecutive-(1, l) or (2, k)-out-of-(2, n): F Type Systems and Its Application in Smart Street Light Deployment". *Methodology and Computing in Applied Probability.* 2023; 25 (1): 33. DOI: https://doi.org/10.1007/s11009-023-09984-3.

13. Jianu M., Daus L., Dragoi V. F., Beiu V. "Reliability polynomials of consecutive-k-out-of-n:F systems have unbounded roots". *Networks.* 2023; 82 (3): 222–228. DOI: https://doi.org/10.1002/net.22168.

14. Yin J., Balakrishnan N., Cui L. "Efficient reliability computation of consecutive-k-out-of-n:F systems with shared components". *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability.* 2022; 224: 108549. DOI: https://doi.org/10.1177/1748006X221130540.

15. Yin J., Cui L., Balakrishnan N. "Reliability of consecutive-(k,l)-out-of-n: F systems with shared components under non-homogeneous Markov dependence". *Reliability Engineering & System Safety.* 2022; 224: 108549. DOI: https://doi.org/10.1016/j.ress.2022.108549.

16. Yi H., Cui L., Gao H. "Reliabilities of Some Multistate Consecutive-k Systems". *IEEE Transactions on Reliability.* 2020; 69 (2): 414–429. DOI: https://doi.org/10.1109/TR.2019.2897726.

17. Lee W. S., Grosh D. L., Tillman F. A. & Lie C. H. "Fault tree analysis, methods and applications:a review". *IEEE Transactions on Reliability.* 1985; 34 (3): 194–203. DOI: https://doi.org/10.1109/TR.1985.5222114.

18. Wu J., Chen C., Ma Y., Xiu Z., Cheng Z., Pan Y., Song S. "Integrated Fault Tree and Case Analysis for Equipment Conventional Fault IETM Diagnosis". *Sensors.* 2025; 25 (17): 5231. DOI: https://doi.org/10.3390/s25175231.

19. Dong H., Xie H., Xu H. "Reliability evaluation of motor controllers based on FMEA and DFTA". Hangkong Dongli Xuebao/Journal of Aerospace Power. 2025; 40 (9): 20230656. DOI: https://doi.org/10.13224/j.cnki.jasp.20230656.

20. Романкевич А. М., Карачун Л. Ф., Романкевич В. А. «Графо-логічні моделі для аналізу складних стійких до відмови обчислювальних систем». *Електронне моделювання.* 2001; 23 (1): 102–111.

21. Romankevich A., Feseniuk A., Maidaniuk I., Romankevich V. “Fault-tolerant multiprocessor systems reliability estimation using statistical experiments with GL-models”. *Advances in Intelligent Systems and Computing*. 2019; 754: 186–193. DOI: https://doi.org/10.1007/978-3-319-91008-6_19.

22. Романкевич В. А., Потапова Е. Р., Бахтари Хедаятоллах, Назаренко В. В. «GL-модель поведінки відмовостійких багатопроцесорних систем з мінімальним числом ребер, що втрачаються». *Вісник НТУУ “КПІ” нформатика, управління та ОТ*. 2006; 45: 93–100.

23. Романкевич А. М, Романкевич В. А., Кононова А. А., Рабах Ал Шбул. «Про деякі особливості GL-моделей K(2,n)». *Вісник НТУУ “КПІ”. Інформатика, управління та ОТ.* 2004; №41: 85-92.

24. Romankevich V. A., Morozov K. V., Romankevich A. M., Morozova A. V., Zacharioudakis L. “On the method of building of non-basic GL-models which are formed on combination of edge functions of basic models”. *Applied Aspects of Information Technology.* 2024; 7 (2): 175–188. DOI: https://doi.org/10.15276/aait.07.2024.13.

25. Romankevich A. M., Morozov K. V., Romankevich V. A. “Graph-Logic Models of Hierarchical Fault-Tolerant Multiprocessor Systems”. *IJCSNS International Journal of Computer Science and Network Security.* 2019; 19 (7): 151–156.

DOI: https://doi.org/10.15276/ict.02.2025.29
UDC 04.005

Method for constructing GL-models of complex non-basic fault-tolerant multiprocessor systems

Vitaliy A. Romankevich[1)]
Doctor of Engineering Sciences, Professor,
Head of the System Programming and Specialized Computer System Department
ORCID: https://orcid.org/0000-0003-4696-5935; zavkaf@scs.kpi.ua. Scopus Author ID: 57193263058
Kostiantyn V. Morozov[1)]
PhD, Associate Professor of the System Programming and Specialized Computer System Department
ORCID: https://orcid.org/0000-0003-0978-6292; mcng@ukr.net. Scopus Author ID: 57222509251
Daniil V. Halytsky[1)]
Postgraduate Student of the System Programming and Specialized Computer System Department
ORCID: https://orcid.org/0009-0004-4421-3443; zipper135401@gmail.com. Scopus Author ID: 58553487600
Ihor A. Yermolenko[1)]
Postgraduate Student of the System Programming and Specialized Computer System Department
ORCID: https://orcid.org/0009-0008-5298-4888; yermolenkomail@gmail.com.
[1)] National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”, 37, Peremogy Ave., Kyiv, 03056, Ukraine

ABSTRACT

The paper is devoted to a method for constructing GL-models of the behavior of complex non-basic fault-tolerant multiprocessor systems under a failure flow. Such models can, in particular, be applied to the reliability analysis of these systems using methods based on statistical experiments. The study considers systems whose behavior can be represented by a set of relatively simple conditions, for each of which methods for constructing the corresponding GL-models are available (for example, the occurrence of no more than a specified number of failures within a certain subset of the system’s processors). It is assumed that the operability of the system is maintained only when all these conditions are simultaneously satisfied. In particular, such systems may include so-called hierarchical systems composed of several subsystems, each having a certain level of fault tolerance, as well as systems containing specialized processors of various types. According to the proposed method, models are first constructed for each individual condition, after which they are combined into a unified GL-model by sequentially merging their graphs through specific vertices (these vertices are merged into one, while the remaining vertices and edges are replicated). The sequence of models and the vertices used for merging can be chosen arbitrarily, which allows, if necessary, obtaining a more convenient graph structure for the resulting GL-model. An example illustrating the application of the proposed method is presented. Two variants of model merging – differing in both the sequence of model integration and the vertices used for merging – are examined. Experimental results confirm that, despite differences in the structure of the resulting model graphs, their behavior for identical input vectors coincides and corresponds to the behavior under a failure flow of the fault-tolerant multiprocessor system for which they were constructed. It is also noted that the proposed method imposes no restrictions on the techniques used to build GL-models for individual conditions (models of different types can be combined simultaneously). The conditions do not necessarily correspond to basic systems (i.e., the failure of no more than a specified number of arbitrary processors), and the graphs of the corresponding models may be non-cyclic.

Keywords: fault-tolerant multiprocessor systems; GL-models; non-basic systems; behavior under a failure flow; control systems; hierarchical systems; reliability assessment; statistical experiments