УДК 004.75:004.453.3 DOI https://doi.org/10.35546/kntu2078-4481.2023.4.28

М. В. ГОРА
аспірант
Харківський національний університет радіоелектроніки
ORCID: 0000-0003-0085-3559

М. О. ВОЛК
доктор технічних наук, професор,
професор кафедри електронних обчислювальних машин
Харківський національний університет радіоелектроніки
ORCID: 0000-0003-4229-9904

МОДЕЛІ УПРАВЛІННЯ РЕСУРАМИ ДЛЯ ЗАБЕЗПЕЧЕННЯ ФУНКЦІОНАЛЬНОЇ СТІЙКОСТІ ПРОЦЕСУ РОЗПОДІЛЕНИХ ОБЧИСЛЕНЬ

У статті розглядаються питання підвищення ефективності систем розподіленої обробки даних з підтримкою функціональної стійкості обчислювального процесу. Однією з основних задач, яка з'являється в процесі підтримки функціональної стійкості, є задача розподілу завдань за обчислювальними ресурсами. В роботі проводиться аналіз сучасних моделей, методів та планувальників, які здійснюють розподіл обчислювальних завдань за розподіленими комп'ютерними ресурсами. Існуючі методи розподілення ресурсів зазвичай використовують різні критерії при виборі вільних ресурсів для використання. Основні критерії – це вартість, час виконання, процент використання ресурсів. Сьогодні важними критеріями стають енергоспоживання, підтримка функціональної стійкості та час самовідновлення систем. Запропоновано моделі управління ресурсами для забезпечення функціональної стійкості, а саме: модифіковану теоретико-множинну модель розподіленого обчислювального процесу з підтримкою функціональної стійкості, модифіковані моделі оцінки часу виконання та енергоспоживання, модель процесу підтримки функціональної стійкості. Експерименти, описані в роботі показують, що використання запропонованих моделей в стандартних методах розподілу ресурсів, стандартних планувальниках дозволило знизити час виконання обчислювальних завдань до 43%, а енергоспоживання у середньому на 26%. Результати дослідження можуть бути використані при розробленні нових методів розподілу ресурсів та технологій розподілених обчислень з використанням отриманих моделей, які враховують засоби підтримки функціональної стійкості обчислювального процесу.

Ключові слова: *розподілені обчислення, комп'ютерні ресурсі, моделі розподілених систем, функціональна стійкість, методи розподілення завдань за ресурсами.*

M. V. HORA
Postgraduate Student
Kharkiv National University of Radio Electronics
ORCID: 0000-0003-0085-3559

M. O. VOLK
Doctor of Technical Sciences, Professor,
Professor at the Department of Electronic Computers
Kharkiv National University of Radio Electronics
ORCID: 0000-0003-4229-9904

RESOURCE MANAGEMENT MODELS FOR ENSURING THE FUNCTIONAL SUSTAINABILITY OF THE DISTRIBUTED COMPUTING PROCESS

The article examines issues of increasing the efficiency of distributed data processing systems with support for the functional stability of the computing process. One of the main tasks that appears in the process of maintaining functional stability is the task of distributing tasks according to computing resources. The paper analyses modern models, methods and planners that perform the distribution of computational tasks according to distributed computer resources. Existing resource allocation methods typically use different criteria when selecting free resources to use. The main criteria are cost, execution time, percentage of resource utilization. Today, energy consumption, maintenance of functional stability and self-recovery time of systems are becoming important criteria. Resource management models for ensuring functional stability are proposed, namely: a modified set-theoretic model of a distributed computing process with support for functional stability, modified models for estimating execution time and energy consumption, a model of the process of supporting functional stability. The experiments described in the paper show that the use of the proposed models in standard methods of resource allocation, standard schedulers made it possible to reduce the time of computing tasks by

43%, and energy consumption by an average of 26%. The results of the research can be used in the development of new methods of resource allocation and technologies of distributed computing using the obtained models, which take into account the means of supporting the functional stability of the computing process.

Key words: *distributed computing, computer resources, models of distributed systems, functional stability, methods of distributing tasks by resources*

Постановка проблеми

Сьогодні спостерігається значний розвиток інформаційних технологій, які пов'язані з розподіленою обробкою даних. Найбільш поширеними з них є технології GRID та хмарних обчислень. Наразі важко відшукати сферу діяльності або виробництва, яка б не використовувала інформаційні технології розподілених обчислень. Застосування таких технологій надає можливість підвищувати ефективність виробничих процесів, зменшує навантаження на персонал та надає нові можливості, які розширюються з кожним роком. Водночас з цим, деякі сфери економіки та суспільного життя вже неможливо уявити без використання розподілених обчислювальних ресурсів, до яких відносяться сервера, бази даних та знань, дата-центри, системи машинного навчання та великих даних.

Широке використання технологій розподілених обчислень пред'являє значні вимоги до функціональної стійкості програмного забезпечення, що лежить в основі будь-якої інформаційної системи. Функціональна стійкість забезпечується різними шляхами, у тому числі за допомогою підтримки надійності, живучості та самовідновлення програмних систем [1]. З боку реалізації, функціональна стійкість може бути реалізованою засобами системи управління розподіленим обчислювальному процесом (РОП). До задач, які вирішує система управління, відносяться моніторинг, виявлення відмов або збоїв та відновлення програмних систем до стану, який є найбільш близьким до того, в якому вони перебували в момент відмови або збою. Тіж дії можуть забезпечуватися засобами, вбудованими в програмне забезпечення (як системного рівня, так і прикладного).

Однією з основних задач, яка з'являється в процесі підтримки функціональної стійкості, є задача розподілу (перерозподілу) завдань за обчислювальними ресурсами [2]. Існуючі методи розподілення ресурсів в розподілених системах зазвичай використовують різні критерії при виборі вільних ресурсів для використання. Основні критерії – це вартість, час виконання, процент використання ресурсів. Сьогодні важними критеріями стають енергоспоживання, підтримка функціональної стійкості та час самовідновлення систем. Зазвичай не достатньо одного критерія. Також користувач може виставляти додаткові умови, які получили назву угоди про рівень послуг (англ. Service-level agreement, SLA), що може об'єднувати якісні та кількісні характеристики наданих послуг, такі як їх доступність, технічне супроводження, час виправлення відмови або збою та інше.

Існує декілька наукових праць, де досліджується ефективне розподілення ресурсів на основі класифікації завдань і порогу прийняття рішень. При цьому покроково виконуються декілька етапів, на кожному з яких працюють різні методи. На першому етапі виконується обробка програмних завдань з метою уникнути вузьких місць в залежності від довгого часу виконання. На другому етапі використовується відомий метод PSO [1] для вибору найкращого порядку виконання завдань. Запропонована комбінація методів була порівняна із методами PSO, енергоефективним алгоритмом балансування навантаження з плануванням робочого процесу, оптимізацією Firefly. Результати експериментів виявили, що запропонований метод набагато кращий у енергозбереженні, розширенні та балансуванні навантаження. У якості майбутніх цілей автори методу вирішили проаналізувати міграції віртуальних машин на базі адаптивних порогових вимог для отримання більшої ефективності в споживанні енергії [4].

В роботі [5] запропоновано метод планування на основі робочого процесу з обмеженим бюджетом, який отримав назву методу планування, що керується завданнями (TDSA). Основна мета методу – оптимізація часу виконання обчислювального процесу. Автори методу запропонували два нових механізми. Перший з них – динамічний механізм розподілу суббюджету для завдання. Другий – механізм планування на основі дублювання завдань. В основі методу лежить розподіл підбюджетних коштів, направлений на повертання бюджету, який не було використано, для подальшого планування. У другому механізмі планування завдань використовуються запити. Після чого знаходяться неактивні пули в обчислювальних ресурсах для вибіркового дублювання завдань. Результати їх експериментів показують, що вони збільшують процент використання ресурсів до 30%. Розроблений метод було порівняно з жадібним алгоритмом резервування ресурсів та модифікованим методом часу найранішнього завершення виконання завдання (GRP-HEFT) [6]. В останній роботіавтори зосередилися на плануванні робочого процесу з обмеженим бюджетом, для чого цілевою функцією боло зменьшення часу виконання завдань. Оскільки модель виставлення рахунків у більшості хмарних систем базується на часі використання ресурсів, а це повсякденна задача будь-якого провайдера, це може призводити до збільшення часу виконання з боку постачальника послуг, а також до непрозорих рішень.

Ще один метод, в основі якого є зниження споживання енергії в хмарі використовує перевідображення критичних завдань (RMREC). Критеріями оптимізації виступають час виконання або вартість в умовах бюджетного обмеження. Метод складається з двох етапів. На першому етапі зорганізується зменшення споживання енергії

з виконанням перерозподілу критичних завдань. Для реалізації цієї мети, виконується розподіл завдань за наявними віртуальними машинами, який базується на найменшому енергоспоживанні і є попереднім. Другий етап методу перенаправляє критичні завдання на віртуальні машини, що мають менші витрати споживання енергії. В експериментальній частині метод був порівняний з методом мінімізації споживання енергії з бюджетними обмеженнями, мінімізацією енергоспоживання та балансуванням бюджетного навантаження [7].

У наступної статті [8] розглянуто підхід кластеризації (англ. clustering approach, CA) для мінімізації порушень SLA. Дослідники зосередилися на прогнозуванні порушень рівнів SLA та передбачанні робочого навантаження в схмарной системі. Для класифікації використовувався підхід Naive Bayer. В роботі було порівняно результати запропонованого методу з результатами роботи активних віртуальних машин без прогнозування та методом, який використовував авторегресійне інтегроване ковзне середне для оцінки продуктивності віртуальних машин.

Новий метод [9] базується на погодинному циклі формування рахунків. Оскільки обмеження даних застосовуються до програмних завдань в хмарі, існує ймовірність неактивних пулів ресурсів у хмарі. Деякі завдання можна використовувати в межах цих неактивних пулів з метою повторного виконання завдань, які вже було виконано на цих ресурсах. Такий підхід дозволяє скоротити час виконання. У цілому, це дозволяє мінімізувати тривалість робочого обчислювального процесу. Водночас забезпечуються бюджетні обмеження на виконання завдання. Запропонована в роботі модель обслуговування TDSA використовує два важливих метода: метод динамічного суббюджетування, який відповідає за повернення невикористаного бюджету та його передачу далі. Другий метод базується на механізмі планування дубльованих завдань, який використовує незадіяні системні пули ресурсів для вибіркового запуску дубльованих завдань. Результат експериментів з впровадження методу показує, що використання методу TDSA зменшує вартість на 14,7%, а використання ресурсів на 30,8% [10]. Оскільки управління обчислювальним процесом і планування в хмарних системах є дуже актуальними задачами, дослідники зосередили увагу на розумному управлінні хмарними ресурсами з метою підвищення продуктивності.

Зменшити витрати на планування обчислювального процесу вдалось в [11] та [12], де було запропоновано підходи до планування завдань, які зменшує час виконання використовуючи коефіцієнти довжини планування (SLR) і прискорення обчислень. Ці методи враховують різнорідність ресурсів та механізм пріоритетів завдань. В роботі порівняно різні методи, і запропонований підхід домінував над існуючими алгоритмами. Показано, що гібридні методи, наприклад, генетичні та імунні, які використовуються для ефективного планування та виконання завдань в хмарному середовищі є перспективними напрямками досліджень [13].

Порівняння описаних методів наведено у таблиці 1.

Таким чином, було проаналізовано різні методи планування завдань і обчислювальних процесів, які враховують одноцільові та багатоцільові підходи. Але планування в хмарних обчисленнях все ще є завданням, оскільки це NP-складна проблема. Крім того, окрім пріоритетного планування розподілу за обраним критерієм, зазвичай стоїть інша задача, наприклад мінімізації енергоспоживання або оцінка порушень SLA. Для подолання цієї прогалини і ефективного планування робочих процесів на обчислювальних ресурсах, в даній роботі запропонована модель планування робочого процесу.

Таблиця 1

Показники ефективності для методів розподілу ресурсів

Назва метода	Енергоспо-живання	SLA	Використання ресурсів	Час виконання	Балансування навантаження
PSO	+			+	+
TDSA	+	+			+
GRP-HEFT				+	
RMREC	+				
CA		+	+		
SLAAEERM	+	+			
MOACO, CCA			+	+	
HDPSO		+	+	+	
TDA			+	+	

Формулювання мети дослідження

Мета даної роботи полягає у підвищенні ефективності управління розподіленим обчислювальним процесом шляхом аналізу і розроблення моделей процесу розподілу програмних завдань за обчислювальними ресурсами з урахуванням забезпечення функціональної стійкості програмного забезпечення інформаційних систем.

Викладення основного матеріалу дослідження

Елементи, якими оперує система управління обчислювальним процесом, зазвичай є різнорідними. Так, наприклад, завдання та обчислювальні ресурси являють собою безліч програм та комп'ютерних компонентів, на яких ці програми можуть бути виконані. Методи розподілу, синхронізації, підтримки функціональної стійкості – це

окремі підпрограми та розподілене програмне забезпечення (ПЗ), які викликаються системою управління для реалізації тих чи інших цілей. Модель системи управління РОП містить наступні елементи:

$$\Omega = < Z, R, Mg, Sh, Q, Sy, Sc, Ms >, \quad (1)$$

де Z – множина завдань, які необхідно виконати в пакетному режимі або режимі реального часу; R – множина обчислювальних ресурсів, які доступні системі управління; Mg – множина сервісів моніторингу параметрів гетерогенної комп'ютерної системи; Sh – множина схем призначення, що визначають порядок розподілу завдань за обчислювальними ресурсами; Q – множина методів розподілу ресурсів (мета застосування методу розподілу ресурсів – створити схему призначення); Sc – множина засобів масштабування; Sy – множина методів синхронізації програм у завданнях; Ms – множина методів, які забезпечують життєздатність обчислювальних процесів у відмові ресурсів.

До елементів моделі системи управління РОП також відноситься множина параметрів, що оцінюють ефективність виконання завдань:

$$O = \{\overline{T_j}, \overline{TR_j}, \overline{TP_j}, \overline{Vz_j}, \overline{Yz_j}, \overline{C_j}\}, j = \overline{1, N} \quad (2)$$

де T – визначає час, витрачений обчислювальним ресурсом виконання завдання, TR и TP – час роботи та простою ресурсу, Vz – вимоги до оперативної пам'яті з боку завдання, Yz – характеристики зв'язності та трафіку всередині завдання; C – вартість виконання завдання, N – кількість завдань.

Як бачимо, наведені параметри мають різну физичну природу та одиницю виміру (час, обсяг памяті, вартість). Крім того, час виконання, простою та вартість – це кількісні характеристики, що характеризують обчислювальний процес. З цього випливає, що управління обчислювальним процесом у даному контексті є складним і погано формалізованим завданням. У процесі експериментів було виявлено ряд параметрів обчислювального середовища, що впливають на ефективність РОП, такі як обсяги оперативної пам'яті, час простою, енергоспоживання, рівень забезпечення SLA та інші. Саме ці параметри враховуються під час реалізації розподілених обчислень.

На основі цього розробимо модифіковану теоретико-множинну модель розподіленого обчислювального процесу з підтримкою функціональної стійкості. Кожен елемент з множини Ω (1) містить програму або сукупність програм (програмну систему), які забезпечують виконання функціональності даного елемента в системі РОП. Таким чином, кожному елементу цієї множини можна поставити у відповідність програмну систему PS, $PS = \bigcup \Pr_i$, яка може містить одну або більше програм: $\forall \omega \in \Omega, \exists ps \in PS$. Кожна програма Pr, що входить до ps складається з коду та даних.

Введемо атрибут t, який визначає часову мітку для кожного елемента множин (1) та (2), яка відображає, що в процесі плину часу, вказані елементи змінюють свій стан.

Зміна стану системи у часі $\Omega^t \xrightarrow{\bigcup \omega^t} \Omega^{t+1}$ або $PS^t \xrightarrow{\bigcup ps^t} PS^{t+1}$ відбувається під впливом однієї або декількох програмних компонент самої системи.

Функціональна стійкість системі в нашому контексті означає, що при відмові будь-якого програмного компонента P, $P \in PS$, система відновить його роботу в найкоротший термін. При відмові однієї з програм (ресурсу), для відновлення функціонування, необхідно відновити код та дані програми на іншому ресурсі. Для забезпечення даного процесу необхідні наступні дії.

1. Постійно зберігати дані програми одним або декількома методами. Дану функцію виконують менеджер, який отримає стан програмних компонент, та модуль відновлення, який зберігає стан програмних компонент в момент часу t_q:

$$Mg^t \xrightarrow{\Pr_d(t_q) \mid \Pr_c(Cd^{ad,sz}, t_q)} Mg^{t+1} \Rightarrow Ms^t \xrightarrow{Mg^t} Ms^{t+1}, \quad (3)$$

де $\Rightarrow$ позначає наслідок дії менеджера, який ініціює збереження даних та передачу їх до модуля відновлення, $Pr_d(t_q)$ та $Pr_c(Cd^{ad,sz}, t_q)$ – процедури збереження стану.

2. Виявити відмову програми (ресурсу) або множини програм:

$$Mg^t \xrightarrow{P^t \not\longrightarrow P^{t+1}} Mg^{t+1} \Rightarrow P^t_{Err}, P_{Err} \in P, \quad (4)$$

де P_{Err} – множина програм, що відмовили.

3. Проводити моніторинг вільних ресурсів R^e, на які можливо перерозподілити програмні компоненти:

$$Mg(R) = R^e = \bigcup_{m=1}^{M} R_m \setminus \bigcup_{i=1}^{N} R_r \,\Big| r = j, \forall \{P_i \to R_j\}, \quad (5)$$

що, в залежності від методу самовідновлення, можна виконувати після отримання множини P_{Err}, або робити постійно у фоновому режимі.

Якщо $P_{Err} = \varnothing$, обчислювальний процес продовжується у штатному режимі. В противному випадку, необхідно ініціювати процес відновлення програми (програм). Для цього треба:

1. Менеджер передає множини програм для відновлення P_{Err} та вільних ресурсів R^e до модулю відновлення: $Mg(R^e, P_{Err}) \Rightarrow Ms(R^e, P_{Err})$.

2. Модуль відновлення обрає новий ресурс (ресурси) для виконання програми (нову схему призначення): $Sh^{Err} = \{P^{Err} \to R^e\}$.

3. Модуль відновлення завантажує відповідний програмний компонент на обраний ресурс та відновляє роботу програми, для чого повертає її до стану, найближчому к часу відмови ресурсу:

$$Ms^t \xrightarrow{P_{Err}, R^e, \Pr'_d(t_q) \mid \Pr'_c(Cd^{ad,sz}, t_q)} Ms^{t+1} \Rightarrow Mg^{t+1}\left(Sh^{Err} = \{P^{Err} \to R^e\}\right), \quad (6)$$

де процедури відновлення $Pr'_d(t_q)$ і $Pr'_c(Cd^{ad,sz}\, t_q)$ стану.

Перед розгортанням, програма профілюється для виявлення потенційних точок відновлення. Після завершення профілювання, програма розгортається у своєму робочому середовищі. На цьому етапі деякі методи самовідновлення можуть вплинити на початковий стан розподіленої системи. Наприклад, зарезервувати ресурси для подальшого відновлення. Це стосується баз даних, в яких планується зберігати стан програмних компонент, множини ресурсів, які залишилися вільними після первинного розподілу, або «гарячих резервів» ресурсів, на яких система може запустити копії деяких критично важливих програмних компонент.

Під час виконання, система аналізує стан програми за допомогою засобів менеджера, які виявляють та звітують про збій програми або системи.

Для оцінки обраної стратегії та моделі управління системою розподілених обчислень пропонуємо моделі оцінки часу виконання та енергоефективності перерозподілу завдань за обчислювальними ресурсами при забезпеченні функціональної стійкості.

Одними з основних критеріїв оцінювання якості обраної стратегії керування є час виконання завдання, вартість виконання та енергоефективність.

У зв'язку з розподіленою природою завдання та обчислювального середовища, час виконання усього завдання (2.4) дорівнює максимальному часу виконання одної з програм в завданні:

$$T_Z = Max(t_i^{finish} - t_i^{start}), i = \overline{1, N} \quad (7)$$

де N – кількість програм в завданні Z, t^{finish} – час завершення виконання програми, t^{start} – час початку виконання програми.

Оцінка споживання енергії на виконання завдань та підтримку функціонування середовища виконання розраховується на основі загального енергоспоживання серверів, які є доступними у центрах обробки даних.

У системах хмарних обчислень споживання енергії має дві складові: фіксоване E_f та динамічне E_d енергоспоживання. Фіксоване споживання енергії обумовлене постійною роботою сервера між виконанням завдань та фоновою роботою інших програм (у тому числі операційної системи та системи управління розподіленими обчисленнями). Динамічне споживання енергії пов'язане з виконанням конкретного завдання та використання інших сервісів у хмарі, які підтримують виконання завдання [14]:

$$E_{full} = (\sigma) \cdot E_f + (1 - \sigma) \cdot E_d, \quad (8)$$

де σ – частина простою ресурса, яка для конкретного обчислювального ресурсу візначається як

$$\sigma_i = \frac{T_{task} - (t_i^{finish} - t_i^{start})}{T_{task}}. \quad (9)$$

Динамічне споживання енергії складається з декількох компонентів: роботи системи охолодження (E_{col}), мережного ресурсу з обміну даними ($E_{network}$), ресурсу зберігання проміжних, вхідних та вихідних даних (E_{storg}) та енергії, яка витрачається на проведення обчислень (E_{CPU}):

$$E_d = E_{CPU} + E_{storg} + E_{network} + E_{col}. \quad (10)$$

В окремих випадках вираз (2.31) може бути розширений додатковими затратами енергії, наприклад при використанні графічних та інших спеціальних процесорів або зовнішніх пристроїв.

Фіксоване енергоспоживання може бути оцінено таким же чином або на основі статистичних амортизаційних характеристик дата центру, які формуються провайдером хмарних послуг.

Цільова функція енергоефективності з урахуванням часу простою та активним часом обслуговування завдань може бути сформульована наступним чином:

$$\min(E_{full}) = \min\left(\sum_{i=1}^{N}(\sigma_i \cdot E_f + (1-\sigma_i)\cdot E_d)\right), \quad (11)$$

за умови однорідності параметрів хмарних обчислювальних ресурсів. Інакше, E_f і E_d доведеться обчислювати для кожного ресурсу окремо.

Ідеальним випадком для виконання завдання є ситуація, коли час простою дорівнює 0. У цьому випадку споживання енергії буде мінімальним. Корекція споживання енергії в режимі простою може відбуватися шляхом динамічного масштабування напруги, частоти конкретного основного процесора та периферійних пристроїв, але в цьому випадку можливе значне збільшення часу «гарячого старту» ресурсу при появі завдання.

Аналіз літератури показує, що для моделювання робочих процесів в розподілених системах, як правило використовується відкритий проект WorkflowSim [15], який дозволяє запропоновані підході порівняти з існуючими підходами та оцінити їх ефективність. В системі реалізовані декілька сучасних методів розподілу ресурсів. Для проведення експериментів, які проводилися з метою перевірки корисності розроблених моделей, було вибрано наступні методи: first-come-first-served (FCFS), Min–Min (мінімальний час виконання, мінімальне енергоспоживання), minimal completion time (MCT) і Round Robin (RR).

Програмні модулі, які реалізують ці методи було модифіковано наступним чином: змінені моделі оцінки часу виконання та енергоспоживання відповідно до виразів (7–11); введена ймовірність відмови одного з обчислювальних ресурсів на протязі певного інтервалу часу; модифіковані засоби моделювання відновлення функціонування системи при відмові ресурсу згідно виразів (3–6); стратегія управління розподіленим обчислювальним процесом направлена на мінімізацію енергоспоживання.

Було використано різну кількість віртуальних машин і різну кількість завдань, щоб перевірити валідність запропонованого нами методу. Кількість завдань визначалась в діапазоні від 50 до 150, кількість віртуальних машин – 10–20, об'єм фізичної пам'яті – 16 Gb, об'єм зовнішньої пам'яті – 4Tb, гіпервізор Xen. Потік завдань, наявна конфігурація та відмови генерувались випадково, після чого застосовувались стандартні та модифіковані з використанням розроблених моделей методи. Отримані в результаті параметри усереднювалися на основі нормування у часі.

На рис. 1 наведено результати моделювання РОП у випадку, коли відмови не відбувалися. З рисунку видно, що час середній час виконання завдань, отриманий за допомогою стандартних та модифікованих моделей приблизно однаковий (похибка не перевищує 1–3%), що говорить про правильність моделей оцінки часу виконання завдань.

На рис. 2 зображено результати декількох експериментів з оцінкою часу виконання завдань з різною ймовірністю η виникнення збою (відмови) одного з обчислювальних ресурсів.

Аналіз результатів показує ефективність запропонованих моделей у випадку відмов ресурсів, а саме зменшення часу виконання завдань. За рахунок відновлення стану програмних компонент, обчислювальних процес відновлює виконання своїх функцій швидше, ніж з застосуванням стандартних методів. Так для ймовірності відмови $\eta = 0.01$, зменшення часу виконання завдань склало 22%, для $\eta = 0.02$ – 43%. І з ростом ймовірності відмов, процент економії часу виконання зростає.

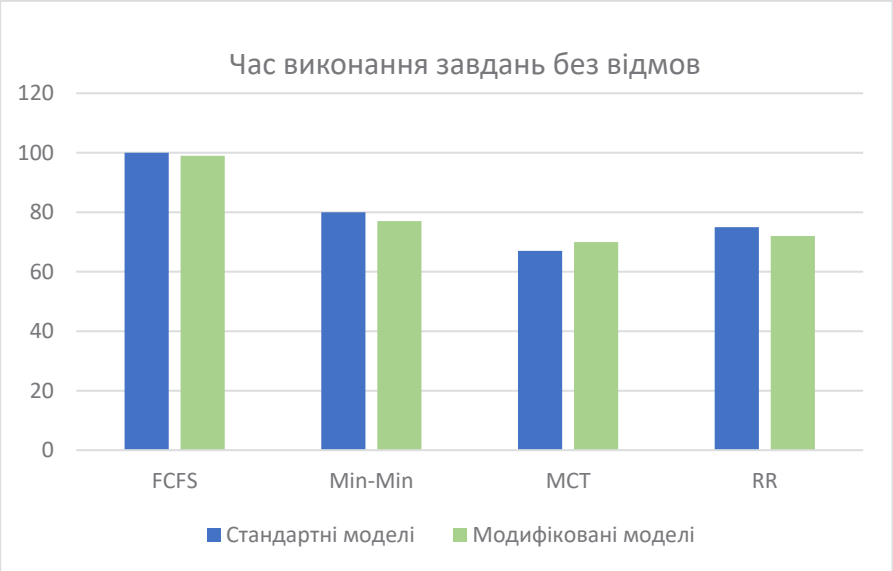

Рис. 1. Оцінка часу виконання завдань з застосуванням стандартних та модифікованих моделей

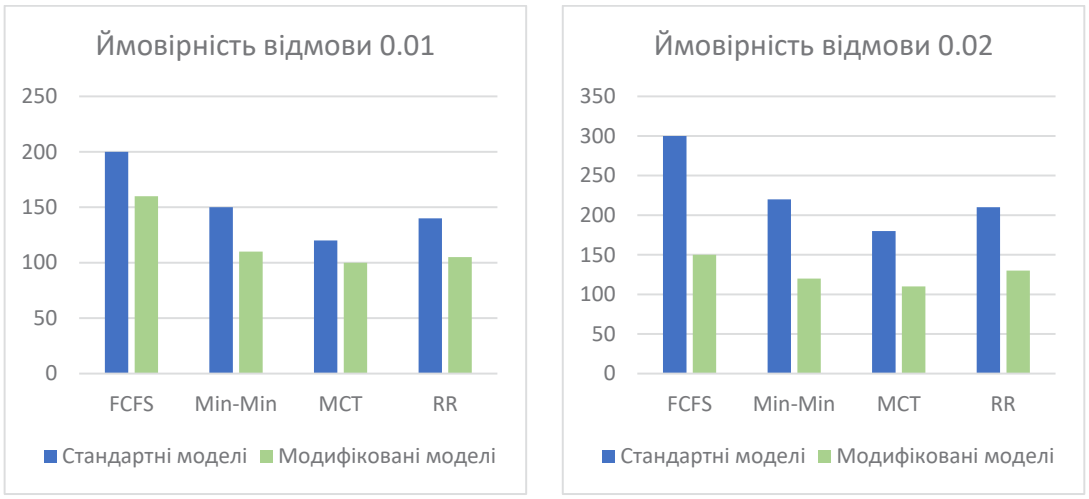

Рис. 2. Оцінка часу виконання завдань з застосуванням стандартних та модифікованих моделей при різної ймовірності відмов ресурсів

На рис. 3 представлені усередненні оцінки енергоспоживання при виконанні завдань стандартними методами планування з використанням стандартних та модифікованих моделей. Умови експериментів були такі ж самі, які описано раніше. Ймовірність відмови обиралась випадковим чином з діапазону $\eta \in [0,0.02]$. Результати свідчать, що запропоновані модифіковані моделі оцінки енергоспоживання РОП з підтримкою функціональної стійкості, дозволяють знизити споживання енергії. У середньому, в процесі проведення експериментів, використання модифікованих моделей в стандартних методах планування, дозволило зменшити споживання енергії на 26 відсотків. При збільшенні розмірності завдань та ймовірності відмов, процент зниження енергоспоживання збільшується.

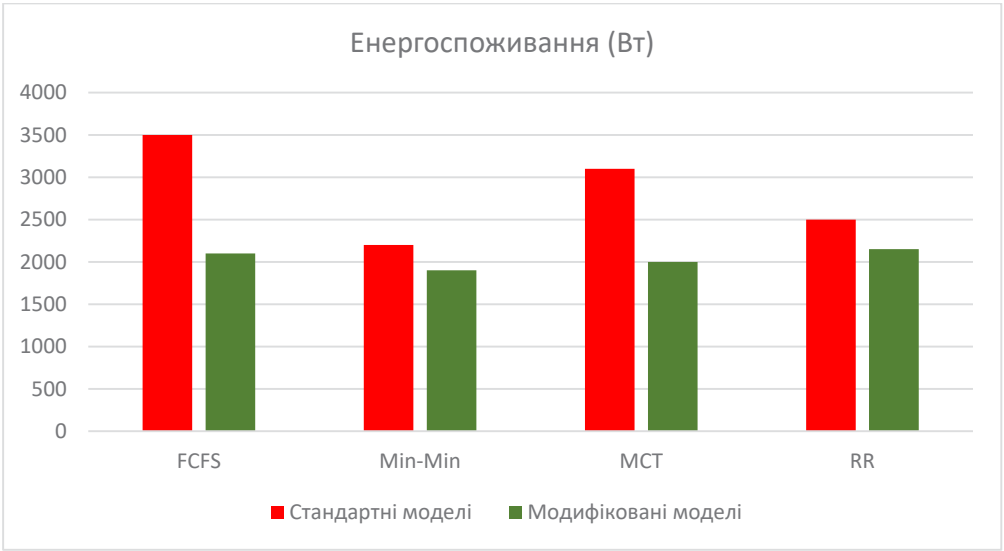

Рис. 3. Оцінка споживання енергії при виконанні завдань з застосуванням стандартних та модифікованих моделей

Висновки

Запропоновані моделі вирішують задачі підвищення ефективності обчислювального процесу в розподілених комп'ютерних системах. Вони включають модифіковану теоретико-множинну модель РОП з підтримкою функціональної стійкості, модифіковані моделі оцінки часу виконання та енергоспоживання.

Використання запропонованих моделей в стандартних методах розподілу ресурсів, стандартних планувальниках дозволило в описаних експериментах знизити час виконання обчислювальних завдань до 43%, а енергоспоживання у середньому на 26%.

Перспективним напрямом бачиться розробка методів розподілу ресурсів та технологій РОП з використанням отриманих моделей, які враховують засоби підтримки функціональної стійкості обчислювального процесу.

Список використаної літератури

1. I. Ruban, M. Volk, T. Filimonchuk, I. Ivanisenko, M. Risukhin, Y. Romanenkov. The Method for Ensuring the Survivability of Distributed Computing in Heterogeneous Computer Systems. 5th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T), Kharkiv, Ukraine, October 9–12, 2018. pp. 1–7.

2. T. Filimonchuk, M. Volk, I. Ruban, V. Tkachov Development of information technology of tasks distribution for grid-systems using the GRASS simulation environment. Eastern-European Journal of Enterprise Technologies. Information and controlling system, 2016. Vol. 3/9 (81). pp. 45–53.

3. P. Wang, Y. Lei, P. R. Agbedanu, and Z. Zhang, "Makespandriven workflow scheduling in clouds using immune-based PSO algorithm," IEEE Access, vol. 8, pp. 29281–29290, 2020.

4. N. Malik, M. Sardaraz, M. Tahir, B. Shah, G. Ali, and F. Moreira, "Energy-efficient load balancing algorithm for workflow scheduling in cloud data centers using queuing and thresholds," Applied Sciences, vol. 11, no. 13, Article ID 5849, 2021.

5. P. Wang, Y. Lei, P. R. Agbedanu, and Z. Zhang, "Makespandriven workflow scheduling in clouds using immune-based PSO algorithm," IEEE Access, vol. 8, pp. 29281–29290, 2020.

6. H. R. Faragardi,M. R. S. Sedghpour, S. Fazliahmadi, T. Fahringer, and N. Rasouli, "GRP-HEFT: a budget-constrained resource provisioning scheme for workflow scheduling in IaaS clouds," IEEE Transactions on Parallel and Distributed Systems, vol. 31, no. 6, pp. 1239–1254, 2020.

7. L. Zhang, L. Wang, Z. Wen, M. Xiao, and J. Man, "Minimizing energy consumption scheduling algorithm of workflows with cost budget constraint on heterogeneous cloud computing systems," IEEE Access, vol. 8, pp. 205099–205110, 2020.

8. R. Anitha and C. Vidyaraj, "Workload and SLA violationprediction in cloud computing," in 2019 Third International Conference on Inventive Systems and Control (ICISC), pp. 582–587, IEEE, Coimbatore, India, 2019.

9. Y. Hu, H. Wang, and W. Ma, "Intelligent cloud workflow management and scheduling method for big data applications," Journal of Cloud Computing, vol. 9, Article ID 39, 2020.

10. Y. Cui and Z. Xiaoqing, "Workflow tasks scheduling optimization based on genetic algorithm in clouds," in IEEE 3rd International Conference on Cloud Computing and Big Data Analysis (ICCCBDA), pp. 6–10, IEEE, Chengdu, China, 2018.

11. M. N. Aktan and H. Bulut, "Metaheuristic task scheduling algorithms for cloud computing environments," Concurrency and Computation: Practice and Experience, vol. 34, no. 9, Article ID e6513, 2022.

12. R. N. Talouki, M. H. Shirvani, and H. Motameni, "A heuristicbased task scheduling algorithm for scientific workflows in heterogeneous cloud computing platforms," Journal of King Saud University-Computer and Information Sciences, vol. 34, no. 8, Part A, pp. 4902–4913, 2022.

13. R. N. Talouki, M. H. Shirvani, and H. Motameni, "A hybrid meta-heuristic scheduler algorithm for optimization of workflow scheduling in cloud heterogeneous computing environment," Journal of Engineering, Design and Technology, vol. 20, no. 6, pp. 1581–1605, 2022.

14. S. Mustafa, K. Bilal, S. U. R. Malik, and S. A. Madani, "SLA-aware energy efficient resource management for cloud environments," IEEE Access, vol. 6, pp. 15004–15020, 2018.

15. WorkflowSim https://github.com/WorkflowSim/WorkflowSim-1.0