УДК 004.56

Енергоефективний підхід до розподілу задач у серверному кластері

Л. С. Глоба, Н. А. Гвоздецька, В. А. Прокопець, О. В. Степурін
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», Україна

У статті описано підхід щодо підвищення енергоефективності обчислень у центрі обробки даних (ЦОД) шляхом застосування енергоефективного розподілу задач в рамках серверного кластера як складової частини інформаційної інфраструктури ЦОД. Запропонований підхід відрізняється одночасним врахуванням параметрів енергоефективності та продуктивності при розподілі задач, на його основі розроблено алгоритм підвищення енергоефективності обчислень, суть якого полягає у попередній індивідуальній атестації серверного кластера. Перевірено ефективність підходу експериментально та шляхом імітаційного моделювання. Проведено аналіз отриманих результатів, та визначено, що підхід проявляє більшу ефективність – до 49,09% за параметром продуктивності та до 9,04% за параметром енергоефективності – для великих гетерогенних кластерів.

Ключові слова: *енергоефективність, продуктивність, серверний кластер, розподіл задач, обробка даних.*

В статье описан подход к повышению энергоэффективности вычислений в центре обработки данных (ЦОД) путем применения энергоэффективного распределения задач в рамках серверного кластера как составляющей части информационной инфраструктуры ЦОД. Предложенный подход отличается одновременным учетом параметров энергоэффективности и продуктивности при распределении задач, а на его базе разработан алгоритм повышения энергоэффективности вычислений, суть которого состоит в предварительной индивидуальной аттестации серверного кластера. Эффективность подхода проверена экспериментально и путем имитационного моделирования. Проведен анализ полученных результатов и определено, что подход дает большую эффективность – до 49,09% по параметру продуктивности и до 9,04% по параметру энергоэффективности – для больших гетерогенных кластеров.

Ключевые слова: *энергоэффективность, продуктивность, серверный кластер, распределение задач, обработка данных.*

The approach to increasing energy efficiency of the data center computing by using energy efficient tasks scheduling within the server cluster as an integral part of the data center infrastructure has been described in the paper. The proposed approach is characterized by taking into account both energy efficiency and performance parameters. The approach is implemented in the task scheduling algorithm. The main idea of the algorithm is to carry out the preliminary attestation of each cluster nodes individually. The efficiency of the proposed approach has been tested by the simulation process and proved experimentally. The approach has shown a gain of up to 49,09% by the performance criteria and up to 9,04% by the energy efficiency criteria for big heterogeneous clusters.

Keywords: *energy efficiency, performance, server cluster, task distribution, data processing.*

1. Проблематика підвищення енергоефективності обчислень у ЦОД

У сучасному світі обсяги даних, що потребують обробки, стрімко зростають. Це пов’язано зокрема із розвитком сучасних технологій таких як Інтернет речей, Хмарні обчислення, технології віртуальної реальності, тощо. Разом із обсягами

© Глоба Л. С., Гвоздецька Н. А., Прокопець В. А., Степурін О. В., 2017

даних зростають і вимоги до швидкості їх обробки. Це спричиняє зростання загального енергоспоживання центрів обробки даних. Частка потужності, що споживається центрами обробки даних, становить близько 1,5% всього обсягу енергоспоживання у світі [1] і постійно зростає. Це робить актуальною проблему підвищення енергоефективності обчислень в центрах обробки даних.

Існує низка методів підвищення енергоефективності обчислень, які можна поділити на статичні та динамічні, такі, що застосовуються на рівні апаратного та програмного забезпечення, а також класифікувати за рівнем їх застосування в системі (на рівні операційної системи, на рівні віртуалізації, на рівні архітектури процесора, тощо) (Рис. 1) [2].

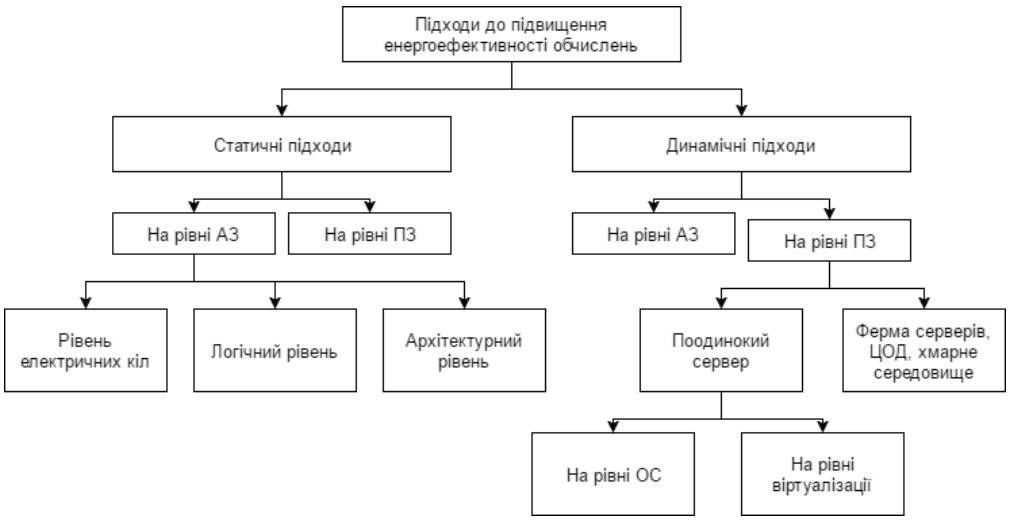


Рис. 1. Класифікація існуючих підходів щодо підвищення енергоефективності обчислень

Підходи на рівні апаратного забезпечення зазвичай мають високу ефективність, проте потребують значних затрат на розробку та побудову додаткових апаратних модулів підвищення енергоефективності та є досить складними у застосуванні. Динамічні підходи, у порівнянні із статичними, дозволяють отримати суттєвіший ефект з огляду на їх спроможність до адаптації до конкретних умов проведення обчислень [2]. Тому, запропонований у статті підхід відноситься до динамічних підходів, що застосовуються на рівні програмного забезпечення у рамках серверного кластера як інформаційної одиниці інфраструктури ЦОД.

В ході аналізу існуючих підходів до підвищення енергоефективності обчислень у ЦОД, що описані зокрема у роботах [3], [4], [5], [6], [7], було виявлено, що жоден з відомих підходів не мінімізує споживання енергії без зменшення при цьому продуктивності роботи обчислювальної системи. Запропонований у даній статті підхід має на меті одночасне підвищення енергоефективності обчислень та збереження рівня продуктивності обробки даних.

2. Запропонований підхід щодо підвищення енергоефективності обчислень у ЦОД

Постановка задачі: розглянемо серверний кластер, як одиницю інформаційної інфраструктури центру обробки даних, що являє собою групу обчислювальних машин (вузлів кластера), з'єднаних високопродуктивними каналами зв'язку. Нехай серверний кластер містить *N* вузлів та має топологію зірки, тобто містить один центральний вузол та *N-1* підлеглих вузлів. Кожен *j-й* вузол кластера характеризується параметрами: k_{kore_j} - кількість обчислювальних ядер процесора, V_j - об'єм оперативної пам'яті, $flops_j$ - продуктивність вузла у кількості елементарних операцій за секунду.

Нова задача $task_i$ приходить до системи в момент τ. Кожна задача для свого успішного виконання потребує певних значень параметрів $V_{req}, k_{core_j}, t_{max}$, тобто доступного обсягу оперативної пам'яті, доступної кількості вільних обчислювальних ядер та характеризується максимальним часом, за який вона має бути оброблена. Набір вимог щодо необхідних значень параметрів $V_{req}, k_{core_j}, t_{max}$ у поєднанні з самою задачею $task_i$ визначає роботу, яка буде виконуватись на певному обчислювальному вузлі.

$$\{V_{req}, k_{core_{req}}, t_{\max}\} \Rightarrow job_i \rightarrow task_i, \{V_{req}, k_{core_{req}}, t_{\max}\} \quad (1)$$

Необхідно:

Розробити підхід до розподілу задач, такий що $P_{\sum} \rightarrow \min$ $t_{task_i} \rightarrow \min$,

де $P_{\sum}$ - сумарна потужність спожита усім кластером,

$t_{\sum}$ - час виконання набору із *m* задач.

При виконанні умов розподілу: $V_{N_j} \geq V_{req}, k_{core_{n_j}} \geq k_{core_{req}}, t_{task_i} < t_{\max}$

Запропонований підхід:

Запропонований підхід до енергоефективного розподілу задач складається з двох основних етапів:

1. Етап попередньої атестації вузлів серверного кластера;
2. Етап динамічного розподілу задач.

Етап попередньої атестації проводиться у серверному кластері періодично, при його налаштуванні. Цей етап передбачає попереднє визначення залежностей енергоспоживання кожного вузла кластера від його завантаженості у вигляді функцій:

$$P = f(CPU),$$

де *P* – потужність, що споживається вузлом кластера,
CPU – завантаженість центрального процесора вузла у відсотках.

Функції *P = f(CPU)* визначаються індивідуально для кожного вузла кластера у табличному вигляді (як результат вимірювання потужності у ряді дискретних точок, що відповідають певним значенням завантаженості), після чого отримані функції представляються у вигляді поліномів ступеню *n* в результаті

інтерполяції їх поліномами канонічного вигляду. Коефіцієнти отриманих поліномів та інші параметри вузлів кластера, такі як кількість ядер процесора, об'єм оперативної пам'яті та продуктивність вузла у кількості елементарних операцій за секунду (величина *FLOPS*), формують описи вузлів, які міститимуться на центральному вузлі кластера із планувальником задач (брокером).

Індивідуальне визначення функцій $P = f(CPU)$ для кожного вузла обробки та їх подальше використання у процесі розподілу задач є ключовою особливістю запропонованого підходу.

Другий етап передбачає вирішення оптимізаційної задачі за критеріями енергоефективності та продуктивності обробки задач з використанням індивідуально визначених енергетичних моделей вузлів (тобто функцій $P = f(CPU)$). Результатом цього процесу є розміщення кожної поточної задачі для обробки на сервер із оптимальними параметрами. Процес розподілу задач є динамічним, оскільки для вибору вузла обробки враховуються поточні характеристики системи обробки (завантаженість вузлів, доступні ресурси, енергоефективність).

В загальному випадку запропонований підхід можна представити у вигляді алгоритму, у якому Крок 0 відповідає за виконання 1 етапу підходу, Кроки 1-7 – за виконання другого етапу.

Алгоритм:

Крок 0. Попередня атестація кластера

В процесі початкового налаштування кластера та його підготовки до роботи необхідно виміряти залежності споживаної кожним сервером електроенергії від завантаженості його центрального вузла обробки. Ця залежність може бути виміряна із використанням навантажувального тесту та представлена у вигляді функції:

$$P = f(CPU)$$

де *CPU – Central Processor Utilization* – завантаженість центрального процесора (вузла обробки).

Функції початково представлені у табличному вигляді. У даній статті запропоновано проводити подальшу інтерполяцію функцій $P = f(CPU)$ поліномами ступеня *n* для отримання аналітичного вигляду функцій.

Після визначення залежностей $P = f(CPU)$ згідно запропонованого підходу відбувається формування опису кожного вузла. До опису входять такі дані:

Функція енергоспоживання сервера від *CPU*;

Загальна продуктивність сервера;

Кількість ядер центрального процесора сервера;

Об'єм оперативної пам'яті сервера.

Крок 1. Оцінка стану кластера в момент τ_{k-1}

У момент часу τ нова задача (робота) job_i надходить до кластера. У момент часу τ_{k-1} (попередня оцінка) необхідно виміряти такі параметри кожного вузла N_i

$\Delta V_{j_{avail}} = V_j - V_{j_used}$ - доступний об'єм оперативної пам'яті *j-го* вузла, де V_{j_used} – оперативна пам'ять, зайнята на момент часу τ_{k-1} ;

$\Delta k_{core_{j_{avail}}} = K_{core\,j} - \Delta k_{core_{j_{used}}}$ - кількість ядер процесора, доступних на вузлі N_j ,

де $k_{core_{j_{used}}}$ - кількість ядер вузла N_j , що зайняті іншими задачами на момент часу τ_{k-1};

$P_{\sum} = \sum_j P_j \mid \tau_{k-1}$ - сумарна потужність, що споживається кластером в момент часу τ_{k-1}.

Крок 2. Виключення із розгляду всіх вузлів, які не підходять для обробки job_i (за відсутністю ресурсів для обробки)

Якщо доступний об'єм оперативної пам'яті менший за той, що вимагається для виконання задачі $task_i (\Delta V_{j_{avail}} \le V_{req_i})$ вузол N_j кластера виключається із розгляду доступних для обробки даної задачі вузлів. Аналогічно, якщо кількість незадіяних у роботі ядер вузла N_j менша, ніж цього вимагає задача $task_i$ вузол N_j виключається із розгляду доступних для обробки даної задачі вузлів.

У результаті таких операцій отримуємо масив $\{N_{avail_j}\}$ теоретично доступних для розміщення задачі $task_i$ вузлів, такий що $\{N_{avail_j}\} \subset \{N_j\}$.

Крок 3. Знаходження масиву значень сумарного енергоспоживання кластера $P_{\sum} = P_{\sum_j}$ при розміщенні задачі $task_i$ на кожен з вузлів N_j

Для кожного *j-го* вузла обчислювального кластера залежність потужності споживання від завантаженості процесора є відомою функцією: $P_j = f_j(CPU_j)$. Цю залежність було виміряно і збережено при проведенні попередньої атестації кластера.

Припустивши, що робота job_i була призначена для виконання вузлу N_j , та знаючи залежність $P_i = f_j(CPU_j)$ для цього вузла, обчислимо теоретичне значення сумарної спожитої кластером потужності у разі розміщення задачі на обробку на *j-й* вузол:

$$P_{\sum_j} \mid \tau_k \;\&\; N_j = P_{\sum} \mid \tau_{k-1} + \Delta P_{j_i} \mid \tau_k$$

Проведемо такі обчислення для кожного вузла кластера.

Таким чином, маємо масив значень $P_{\sum} = \sum P_{\sum_j}$ отриманий шляхом теоретичного розміщення роботи job_i на кожний з вузлів N_j

Крок 4. Сортування масиву вузлів N_j за теоретичним вкладом у загальну споживану потужність: $N_{P_{avail}} = \{N_{flops_{i_{\max}}}, ... N_{flops_{i_{\min}}}\}$ $N_P = \{N_{P_j}\}$

Проведемо сортування масиву доступних для розміщення задачі $task_i$ вузлів N_{avail_i} за їх теоретично обрахованим на кроці 3 вкладом у загальне сумарне енергоспоживання всього обчислювального кластера $\Delta P_j \mid \tau_k$. Нехай сортування буде також проведене в порядку спадання. В результаті отримаємо масив $N_{P_{avail}}$:

$$N_{P_{avail}} = \{N_{p_{j_{\max}}}, ...N_{p_{j_{\min}}}\}$$

Крок 5. Сортування масиву вузлів, що лишилися, за продуктивністю (*FLOPS*)

Проведемо сортування масиву доступних для розміщення задачі $task_i$ вузлів N_{avail_i} за їх продуктивністю $flops_i$ у порядку спадання.

В результаті отримаємо масив $N_{flops_{avail}}$:

$$N_{flops_{avail}} = \{N_{flops_{i_{\max}}}, ...N_{flops_{i_{\min}}}\}$$

Крок 6. Призначення вагових коефіцієнтів (балів) за продуктивність та енергоефективність кожному вузлу

В залежності від позиції вузла із номером *j* у сортованому масиві $N_{flops_{avail}}$ та $N_{P_{avail}}$, він отримує два вагових коефіцієнта (дві оцінки): $mark_P$ і $mark_{flops}$ відповідно, при чому $mark_P$, $mark_{flops} = \overline{1,n}$ де *n* – кількість вузлів у кластері. $mark_P$ рівна позиції *j-го* вузла у сортованому масиві $N_{P_{avail}}$, $mark_{flops}$ – позиції цього вузла у сортованому масиві $N_{flops_{avail}}$ відповідно.

Крок 7. Розміщення задачі $task_i$ на вузол із найбільшим сумарним ваговим коефіцієнтом

Сумарна оцінка кожного вузла:

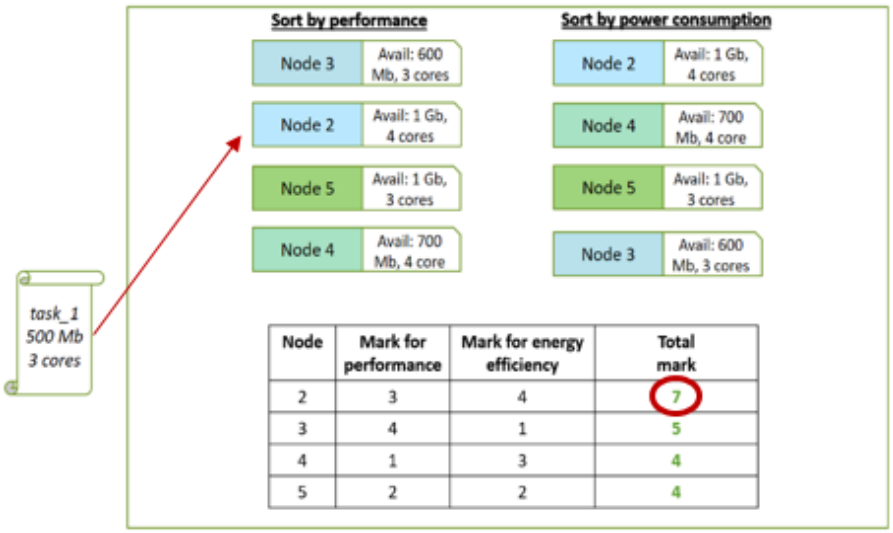
$$mark_{\sum i} = mark_{flops_i} + mark_{P_i}$$

Вузол, для якого $mark_{\sum i} = \max_j mark_{\sum k}$ вибирається для здійснення обробки завдання $task_i$

Приклад роботи алгоритму при виборі із 4 підходящих вузлів кластера за їх сумарними оцінками зображено на Рис. 2.

З рис.2 видно, що найбільший рейтинг за продуктивністю має *Node 3*, а найефективнішим вузлом за енергоефективністю є *Node 2*.

При підрахуванні найвищого рейтингу за комплексом показників визначено, що найкращим є вузол *Node 2*, тому що має найбільший загальний рейтинг. Виходячи з цього цей вузол було визначено як цільовий вузол, куди брокером буде направлено задачу (роботу) job_i для її виконання.



Node	Mark for performance	Mark for energy efficiency	Total mark
2	3	4	7
3	4	1	5
4	1	3	4
5	2	2	4

Рис. 2. Приклад вибору вузла за сумою балів за продуктивність та енергоефективність обчислень

3. Експериментальна перевірка ефективності запропонованого підходу

Для перевірки енергоефективності запропонованого підходу було проведено натурний експеримент, в ході якого було апробовано роботу підходу у невеликому кластері Дрезденського технічного університету.

Для проведення експерименту було використано 5 машин (серверів). Серед них 3 належали до одного типу (мали подібні фізичні характеристики), а 2 інші – до другого типу. Сервери було об'єднано у кластер із топологією зірки (Рис. 3).

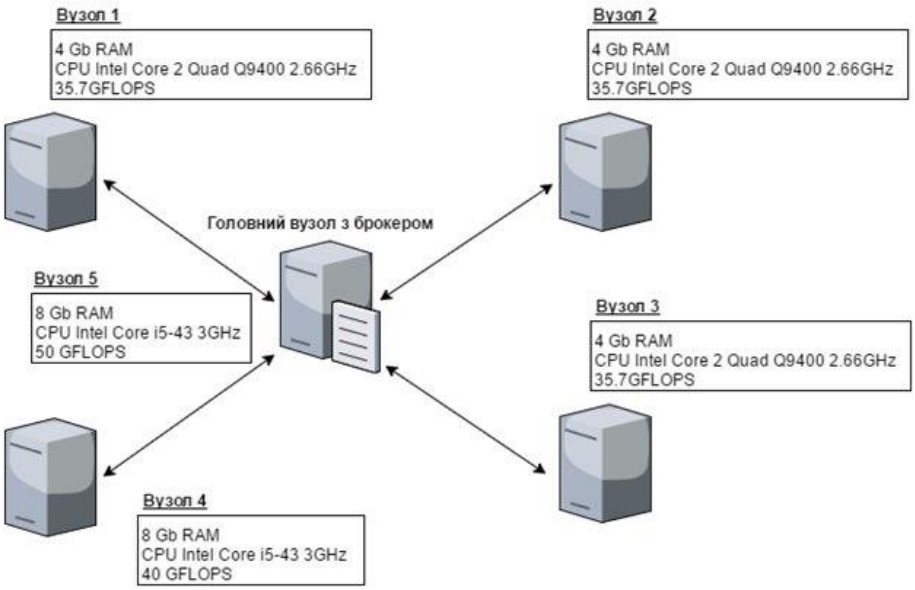


Рис. 3. Топологія серверного кластера, використаного при експериментальній перевірці запропонованого підходу

На центральний вузол кластера було завантажено код планувальника задач, що працює за запропонованим алгоритмом і реалізований на мові програмування С.

Для моделювання реальної роботи серверного кластера було обрано різні за характеристиками задачі, при цьому задачі були об'єднані у пакет задач.

Для зняття характеристик $P_j = f_j(CPU_j)$ кожного вузла, було використано цифровий мультиметр Yokogawa WT210, що здатний вимірювати струм, напругу та миттєву потужність безпосередньо між джерелом живлення та навантаженням. Також, для зняття характеристик $P_j = f_j(CPU_j)$ було використано навантажувальний тест (stress-test) [8].

Зняття характеристик проводилось шляхом послідовного підключення мультиметра у коло живлення кожного сервера. Вимірювання потужності споживання проводилось для 0, 25, 50, 75 та 100% завантаженості *CPU*. Визначені експериментально характеристики зображені на Рис. 4а у графічному та на Рис. 4б у аналітичному представлені внаслідок інтерполяції їх поліномами 3 ступеня.

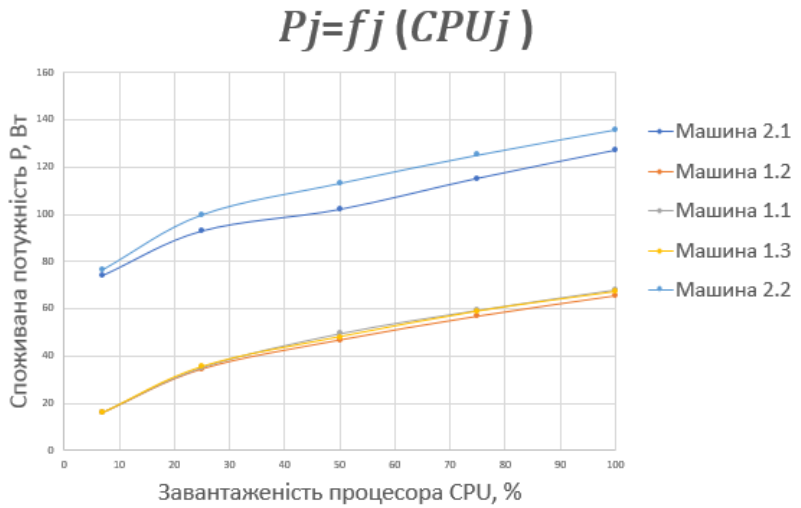


Рис. 4а. Експериментально визначені залежності енергоспоживання вузлів кластера від їх завантаженості (у графічному вигляді)

$$P(CPU) = 0.0001 * x^3 - 0.0139 * x^2 + 1.0443 * x^1 + 15.86 * x^0$$

$$P(CPU) = -0.0018 * x^2 + 0.8275 * x^1 + 16.13 * x^0$$

$$P(CPU) = 0.0001 * x^3 - 0.0107 * x^2 + 0.9536 * x^1 + 15.86 * x^0$$

$$P(CPU) = 0.0005 * x^3 - 0.0328 * x^2 + 1.3271 * x^1 + 73.91 * x^0$$

$$P(CPU) = 0.0002 * x^3 - 0.0212 * x^2 + 1.3389 * x^1 + 76.33 * x^0$$

Рис. 4б. Експериментально визначені залежності енергоспоживання вузлів кластера від їх завантаженості (у аналітичному вигляді)

Для перевірки ефективності підходу його було порівняно із широко застосовуваним у реальних кластерах алгоритм планування Round Robin [9]. Порівняння було здійснено за двома критеріями – середній час обробки задачі (як критерій продуктивності обчислень) та критерій енергоефективності обчислень у вигляді сумарної спожитої кластером потужності за час обчислень. Результати експерименту наведено у Табл.1.

Табл. 1. Результати експериментального порівняння запропонованого підходу із широко використовуваним підходом Round Robin

Алгоритм	Середній час обробки задачі, *с*	Сумарна спожита потужність, *Вт*	Виграш у порівнянні із Round Robin, %	
			За часом	За спожитою енергією
Round Robin	14.835	376.88	-	-
Запропонований підхід	13.324	365.8	10,2%	3%

Отриманий виграш є незначним. Виграш за параметром енергоефективності майже підпадає під класифікацію статистичної помилки (при визначеній похибці у 2%). Такий незначний виграш пояснюється тим, що вузли серверного кластера, використаного в експерименті, за своїми характеристиками належали до 2 груп, при чому в рамках груп характеристики були однаковими.

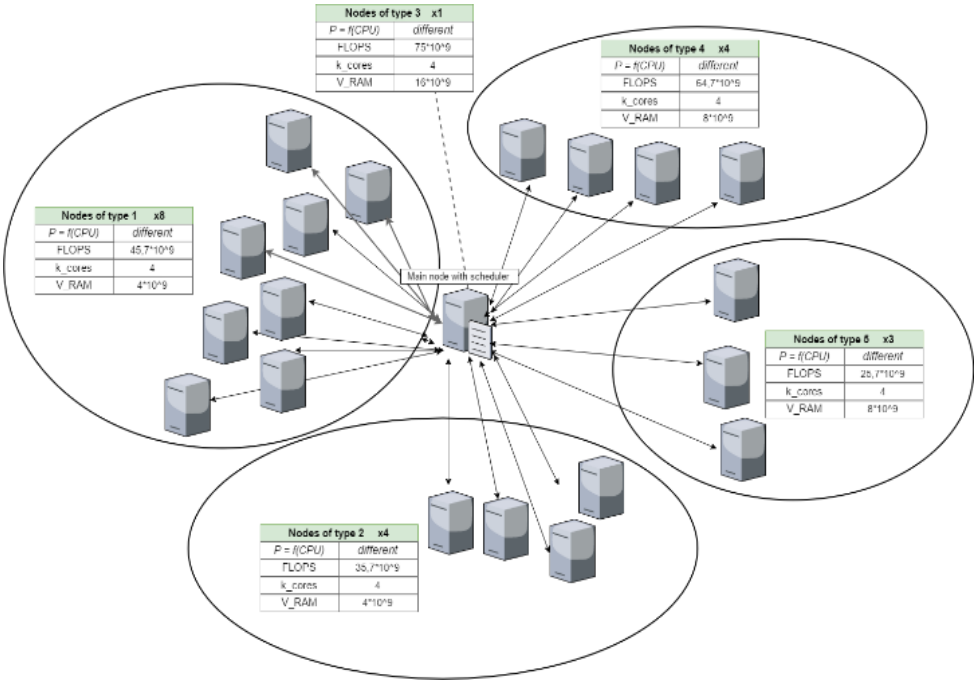


Рис. 5. Топологія кластера з 20 вузлів, використаного у ході імітаційного моделювання

Крім того, залежності $P = f(CPU)$ для досліджуваного кластера також були близькими.

Для подальшого дослідження ефективності підходу (для масштабніших та гетерогенних кластерів), було використано засоби імітаційного моделювання у середовищі Matlab. Перш за все було доведено, що у випадку використання імітаційної моделі результати є статистично еквівалентними та їх похибка не перевищує допустимого рівня.

Наступним кроком було проведено імітаційне моделювання роботи підходу у серверному кластері з 20 вузлів, топологія та характеристика вузлів якого наведена на Рис. 5.

При цьому, запропонований підхід було порівняно із рядом широко відомих алгоритмів планування, серед яких і два енергоефективні підходи, описані у джерелах [6] та [7]. Було змодельовано роботу кластера впродовж доби із навантаженням, подібним до реального [10].

Результати моделювання показали, що запропонований підхід дозволяє отримати виграш до 49,09% за параметром продуктивності обчислень та виграш до 9,04% за параметром енергоефективності.

Такі результати дозволяють зробити висновок, що запропонований підхід може суттєво підвищити енергоефективність обчислень, одночасно приносячи виграш за продуктивністю. Розглянутий у ході моделювання серверний кластер порівняно не є великим. У випадку використання підходу у більш масштабних кластерах очікується зростання його ефективності, що буде перевірено у рамках майбутніх досліджень.

4. Висновки

У статті запропоновано підхід до підвищення енергоефективності обчислень для серверного кластера як інформаційної одиниці інфраструктури ЦОД, який відрізняється одночасним врахуванням параметрів енергоефективності та продуктивності при розподілі задач.

Розроблено алгоритм підвищення енергоефективності обчислень на основі підходу, суть якого полягає у попередній індивідуальній атестації серверного кластера та застосуванні алгоритму енергоефективного розподілу задач.

Врахування індивідуальних залежностей енергоспоживання серверів від їх завантаженості є основною відмінною рисою запропонованого підходу.

Перевірено ефективність підходу експериментально та шляхом імітаційного моделювання.

Проведено аналіз отриманих результатів, та визначено, що підхід проявляє більшу ефективність – до 49,09% за параметром продуктивності та до 9,04% за параметром енергоефективності – для великих гетерогенних кластерів.

ЛІТЕРАТУРА

1. Koomey J. G., Worldwide electricity used in data centers / J. Koomey // Environmental Research Letters, vol. 3, no. 3, p., 2008.
2. Beloglazov A. Energy-Efficient Management of Virtual Machines in Data Centers for Cloud Computing / Beloglazov Anton – Melbourne, 2013.

3. Hosseinimotlagh S. A Cooperative Two-Tier Energy-Aware Scheduling for Real-Time Tasks in Computing Clouds / S. Hosseinimotlagh, F. Khunjush, S. Hosseinimotlagh. // Euromicro International Conference on Parallel, Distributed, and Network-Based Processing. – 2014. – №22. – С. 178–182.
4. An Energy and Deadline Aware Resource Provisioning, Scheduling and Optimization Framework for Cloud Systems / Y.Gao, Y. Wang, S. K. Gupta, M. Pedram. // IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis. – 2013. – №9. – С. 31:1–31:10
5. Min_ c: стратегия неоднородной концентрации задач для энергосберегающих компьютерных расписаний / [Ф. Армента-Кано, А. Черных, Х. М. Кортес-Мендоза та ін.]. // Труды ИСП РАН. – 2015. – №6. – С. 355.
6. Liu N. Task Scheduling and Server Provisioning for Energy-Efficient Cloud-Computing Data Centers / N. Liu, Z. Dong, R. Rojas-Cessa. // IEEE 33rd International Conference on Distributed Computing Systems Workshops. – 2013. – №33. – С. 226–231.
7. Грушин Д. А. Энергоэффективные вычисления для группы кластеров / Д. А. Грушин, Н. Н. Кузюрин. – Москва, 2013. – 433 с.
8. Stress-test, POSIX [Електронний ресурс] – Режим доступу до ресурсу: https://people.seas.harvard.edu/~apw/stress/.
9. Jyoti V. Comparative Study of Load Balancing Algorithms / V. Jyoti, K. J. Anant. // IOSR Journal of Engineering (IOSRJEN). – 2013. – С. 45–50.
10. Проектирование ЦОД и строительство дата-центра [Електронний ресурс] – Режим доступу до ресурсу: http://www.datacenter-ts.ru/.

Надійшла – 09.10.2017.