Технічні науки

УДК 004.891

Хмельницький Юрій Владиславович

кандидат технічних наук,

доцент кафедри комп'ютерних систем та мереж

Хмельницький національний університет

Хмельницкий Юрий Владиславович

кандидат технических наук,

доцент кибербезопасности и компьютерных систем и сетей

Хмельницкий национальный университет

Khmelnytsky Yuri

PhD in Technical Sciences, Assistant Professor of the Department of

Cybersecurity of Computer Systems and Networks

Khmelnytsky National University

Карун Марія Євгеніївна

студент

Хмельницького національного університету

Карун Мария Евгеньевна

студент

Хмельницкого национального университета

Karun Maria

Student of the

Khmelnytsky National University

БАЛАНСУВАННЯ НАВАНТАЖЕННЯ ДОДАТКІВ У «ХМАРНОМУ» СЕРЕДОВИЩІ

БАЛАНСИРОВАНИЕ НАГРУЗКИ ПРИЛОЖЕНИЙ В «ОБЛАЧНОЙ» СРЕДЕ

LOAD BALANCING OF CLOUD APPLICATION

Анотація. *Досліджено теоретичні питання балансування навантаження у «хмарному» середовищі.*

Ключові слова: *хмарні технології, хмарні сервіси, балансування навантаження, масштабування, розподілений веб-додаток.*

Аннотация. *Исследованы теоретические вопросы балансирования нагрузки в «облачной» среде.*

Ключевые слова: *облако, облачные вычисления, IaaS, балансировка нагрузки, масштабирование, веб-приложение.*

Summary. *The theoretical aspects of load balancing of cloud application.*

Key words: *cloud, cloud computing, IaaS, load balancing, scalability, web-based application.*

Вступ. Інформаційні технології є однією з перспективних галузей сучасної науки. З'являються нові проблемно-орієнтовані галузі, виникають перспективні інноваційні рішення. Хмарні технології входять в число нових, найсучасніших інформаційних методів обробки даних та інформації. Розвиток світової IT-індустрії йде за чотирма головними напрямками:

– хмарні обчислення, мобільні рішення і мобільні технології;

– обробка та аналіз великих даних;

– забезпечення всіх видів інформаційної безпеки і конфіденційності;

– захист даних.

«Хмари» активно впроваджуються в різні сфери діяльності – медицину, освіту, органи влади, науку. Не стала винятком і сфера економіки і бізнесу.

Хмарні технології полягають в наданні кінцевим користувачам віддаленого динамічного доступу до послуг, обчислювальним ресурсам та додаткам через Інтернет» [18].

Хмарні технології почали свій розвиток ще в 1960-ті роки з виникненням комп'ютерної техніки. Але особливо активно розвиватися і застосовуватися вони стали з 2009 року, коли було запущено додаток Google Apps.

Постановка задачі. На основі проведеного деталізованого аналітичного огляду технології хмарних обчислень показано, що споживачі хмарних обчислень можуть значно зменшити витрати на інфраструктуру інформаційних технологій (в короткостроковому і середньостроковому планах) і гнучко реагувати на зміни обчислювальних потреб, використовуючи властивості обчислювальної еластичності хмарних послуг.

Наведено та проаналізовано принципи роботи хмарного середовища, проведено аналіз особливостей моделей хмарного розміщення, а саме: програмне забезпечення як послуга (SaaS), платформа як послуга (PaaS) та інфраструктура як послуга (IaaS). Виділено переваги та недоліки хмарних технологій, які показують, що завдяки об'єднанню ресурсів та непостійному характеру споживання з боку користувачів, можна забезпечити економію в масштабі системи, використовуючи менші апаратні ресурси, ніж при виділенні апаратних потужностей для кожного споживача, а завдяки автоматизації модифікації виділення ресурсів значно знижуються витрати на обслуговування.

Але при цьому використання хмарного середовища має ряд невирішених питань. Гострим на сьогодні залишається питання безпеки, конфіденційності та балансування навантаження у хмарному середовищі. Отже метою даного дослідження є оцінка можливостей використання алгоритмів балансування навантаження на додатки у хмарному середовищі.

Основна частина. При побудові розподілених систем обчислення у зв'язку з розвитком засобів передачі даних у все більшій мірі

використовуються підходи розподіленого програмування. Виконання великих задач розподіляється на декілька менших підзадач, які виконуються на різних комп'ютерах що мають спільну мережу, а після всіх обчислень результати їх роботи можуть бути використані при обчисленні початкових задач. При вирішенні деяких завдань використання розподіленої системи застосовується для підвищення наступних показників ефективності: зниження вартості, збільшення надійності, досягнення певного рівня продуктивності системи, простоти масштабування.

У веб-сервісах завжди є клієнт і сервер. Сервер – це веб-сервіс (endpoint: кінцева точка, куди доходять SOAP повідомлення від клієнта).

Основні кроки реалізації:

– опис інтерфейсу веб-сервісу;

– реалізація інтерфейсу;

– запуск веб-сервісу;

– створення клієнта і віддалений виклик потрібного методу веб-сервісу.

Балансування навантаження – метод розподілу завдань між декількома мережевими пристроями з метою оптимізації використання ресурсів, скорочення часу обслуговування запитів, горизонтального масштабування кластера (динамічне додавання / видалення пристроїв), а також забезпечення відмовостійкості (резервування).

При балансуванні навантаження «хмарних» додатків слід враховувати особливості роботи додатків, особливості інфраструктури обчислювальних комплексів і додаткові вимоги при організації сервісів.

Для порівняння можливостей систем балансування навантаження додатків в "хмарних" середовищах визначимо основні критерії такого порівняння.

Ефективність розподілу навантаження – характеризує можливості системи забезпечити рівномірність розподілу навантаження між обслуговуючими вузлами при різних вимогах до роботи додатків і до структур обчислювальних комплексів, які можуть бути враховані через наявні в системі алгоритми балансування навантаження і тип балансування (статичне і динамічне).

Основною перевагою статичного розподілу, який не змінюється у процесі роботи сервісу, є простота реалізації. Динамічне балансування є більш гнучким і передбачає перерозподіл навантаження на обчислювальні вузли під час роботи сервісу на підставі визначення прямих показників (завантаження центрального процесора, обсяг доступної оперативної пам'яті, доступного дискового простору тощо), або непрямих (кількість активних підключень до сервера чи час його відгуку).

Врахування геоположення вузлів – характеризує можливість системи забезпечити зниження часу її відгуку через врахування географічної близькості певних груп клієнтів до певних груп обслуговуючих вузлів. Потрібно зауважити, що ефективність методу дуже залежить від актуальності даних про довжину маршрутів, пропускну спроможність і завантаження ліній зв'язку між потенційними клієнтами і серверами.

Незалежність від типу мережі – незалежність системи балансування від того, у локальній чи у глобальній мережі знаходяться обслуговуючі сервери, а також незалежність від типу серверної платформи.

Можливості автоматичного масштабування – характеризують наявність в системі механізмів горизонтального масштабування обслуговуючих вузлів при балансуванні навантаження. Автомасштабування в «хмарних» платформах забезпечується спеціальними модулями, зв'язаними з системами балансування навантаження, які безпосередньо цю задачу не вирішують [11].

Автоматичне визначення і видалення непрацюючих вузлів – забезпечення системою балансування відмовостійкості сервісу на основі моніторингу доступності і працездатності його компонентів та автоматичного видалення непрацюючих вузлів з пулу розподілу навантаження.

Забезпечення довгострокових сесій – характеризує можливість системи балансування забезпечити додатку зберігання стану сесії користувача з декількох повторних запитів. Може реалізуватися в системі або через з'єднання користувача з одним і тим самим сервером, або через зберігання сесії в загальному для всіх серверів сховищі. Прив'язка клієнта до конкретного сервера зазвичай реалізується або за його IP-адресою (яка, однак, може змінюватись через певний час), або по cookies (в який записується ідентифікатор сервера).

Балансування навантаження в хмарному середовищі має відміності від класичної моделі архітектури балансування навантаження та її впровадження за допомогою серверу для виконання балансування навантаження, оскільки неможливо передбачити кількість запитів, які передаються на сервери. Це надає нові можливості, а також представляє свій унікальний комплекс завдань. Балансування навантаження є однією з центральних проблем в області хмарних обчислень. Це механізм, який рівномірно розподіляє динамічне локальне навантаження на всіх вузлах по всій хмарі, щоб уникнути ситуації, коли деякі вузли сильно завантажуються, тоді як інші не працюють.

Спосіб динамічного балансування навантаження в хмарному середовищі на основі методу міграції задач та модернізованого алгоритму Weighted Least Connections, а саме алгоритм динамічного навантаження з урахуванням часу обробки завдання та середнього коефіцієнта використання ресурсів як параметра є оптимальним рішенням для поставленого завдання.

Загальна схема запропонованого способу динамічного балансування навантаження представлена на рис. 1

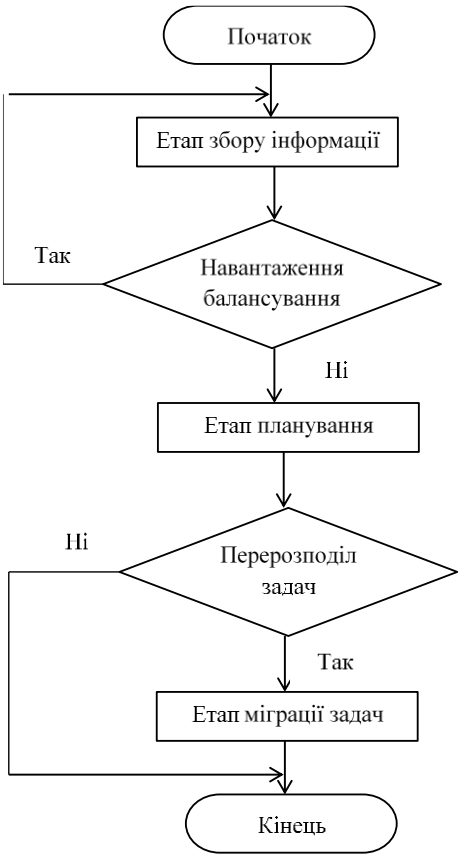

Рис. 1. Загальна схема запропонованого способу динамічного балансування навантаження

Планування виконання n завдань в m вузлах (віртуальних машинах) необхідно виконати так, щоб користувачі хмарного середовища могли виконувати свої завдання у мінімальні часи роботи з максимальними використаннями ресурсів.

Планувальник задач отримує N запитів на завдання T_1,T_2 T_3……T_N. Всі завдання не являються пріоритетними та незалежними, кожне завдання має довжину T L_1 , p швидкість обробки, кількість процесорів q, обсяг основної пам'яті r та обов'язково смугу пропускання B. Планувальник завдань (рис.2) в хмарному середовищі має інформацію про віртуальну машину M, а саме швидкість обробки процесора, кількість

процесорів, пам'ять, пропускну спроможність VM_1,VM_2 VM_3 VM_4……VM_N.

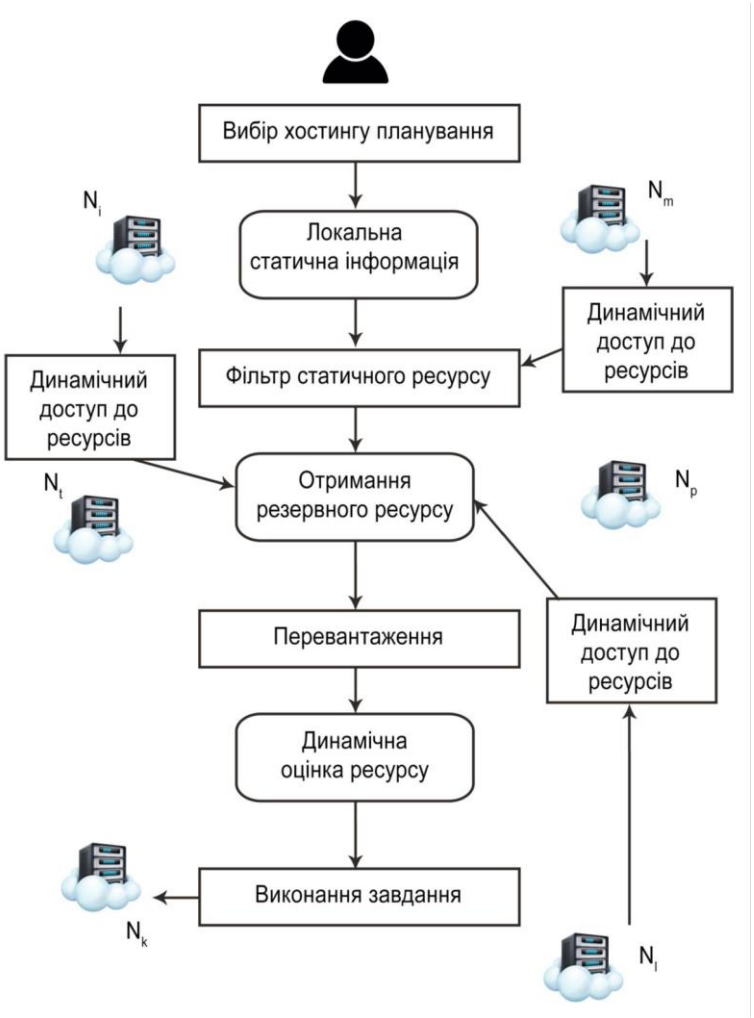

Рис. 2. Опис процесу планування завдань

Для обчислення потенціалу окремої віртуальної машини та ємності використовуємо формулу:

$$C_{VM} = p*q \quad (1)$$

де:

p – швидкість обробки процесора в мільйонах інструкцій за секунду;

q– кількість процесорів зайнятих для виконання завдання.

Завантаження інформації на віртуальну машину: планувальник завдань із хмарного середовища розподіляє завдання на віртуальну машину, кожна віртуальна машина має чергу для зберігання навантаження.

Загальна довжина черги у віртуальній машині визначається як навантаження на цю віртуальну машину [12].

Завантаження віртуальної машини можна розрахувати як

$$LVM_{i,t} = \frac{K*TL_{i(t)}}{S(VM_{i,t})} \quad (2)$$

де K=1,2,3…….N завдання.

$S(VM_{i,t})$ визначається як швидкість обслуговування віртуальної машини в момент часу t, який можна виразити у формі потужності p та кількості процесорів q як p*x(t), де x=1,2,3…q.

Навантаження віртуальної машини за часом t обчислюємо як кількість задач на конкретній віртуальній машині, поділену на швидкість обслуговування віртуальної машини.

Отже, загальне навантаження на всю віртуальну машину дорівнює

$$L = \sum_{j=1}^{M} LVM_j \quad (3)$$

Якщо заплановане робоче навантаження всієї системи є більшим, ніж її потенціал, то центр обробки даних не має можливості впоратися з усіма майбутніми запитами, тому або відмовиться від майбутнього запиту на завдання, або збільшить віртуальну машину за допомогою концепції еластичності. Якщо майбутнє робоче навантаження менше, ніж потенціал всієї віртуальної системи, то знаходиться навантаження на кожну окрему віртуальну машину та визначається, чи є віртуальні машини, які перевантажені або завантажені. Якщо всі віртуальні машини перевантажені, то завдання переноситься на завантажену віртуальну машину так, щоб всі завдання могли бути виконані за мінімальний час.

Час передачі завдання може бути розрахований TT = довжина завдання (TL) / пропускна спроможність (B).

Знаходимо час виконання завдання T_i на віртуальній машині VM_j

$$T_{exej} = \frac{\sum_i^N E_{ij*TL_i}}{p*q} \quad (4)$$

де E_{ij} = 1, якщо завдання T_i призначене віртуальній машині VM_j, інакше значення = 0.

Необхідно визначити час, який складається з часу виконання та передачі завдання (за умови, що будь-яке завдання може переміщуватися з перевантаженої віртуальної машини до завантаженої).

$$MST = \max\{\sum_{j=1}^{M} T_{exej}\} \qquad (5)$$

Міграція для вузла, який визначає, чи є вигідною міграція одного процесу або суперпроцесу, вона потребує лише знання про:

– обсяг даних, обмінюваних для зовнішнього зв'язку процесів, на яких він розміщений;

– обсяг даних, що обмінюються для внутрішньої (локальної) комунікації процесів, на яких він розміщується;

– найближчих сусідів, залучених до вищезгаданого зовнішнього зв'язку.

Оскільки вся ця інформація вже відома вузлам всередині системи, виконання Weighted Least Connections (WLC) базується лише на локальній інформації.

Алгоритм WLC не вимагає ніякої локальної інформації, яка ще не доступна або яка не може бути зібрана та мати незначну вартість. Прикладом такої інформації є характеристики навантаження вузлів / процесів, від яких алгоритм не залежить. Для правильного функціонування алгоритму кожен процес повинен знати розташування суміжних процесів. Розумно припустити, що або ця інформація є легкодоступною для того, щоб відбувався процес комунікації, або це може бути поширено повільно, коли процеси спілкуються один з одним.

Щоб відобразити динамічні зміни в обсязі трафіку між двома процесами, приймаємо метод усереднення.

Розглянемо стандартизацію атрибуту ресурсу. Серед рішень проблем багатоцільового атрибуту спосіб квантості атрибутів безпосередньо впливає на об'єктивну оцінку. У процесі оцінки ресурсів використовується кілька показників; кожен індикатор має іншу одиницю і розмірність. Щоб краще відображати фактичний стан ресурсів, кожен індикатор потрібно порівнювати за уніфікованими показниками; тому, перш ніж оцінювати кожен ресурс обчислювального вузла всебічно, нам потрібні вимірювання та стандартизація відповідних показників. У даній роботі використовується метод середнього геометричного методу в методі стандартизації векторів для стандартизації атрибутів ресурсу, метод простий для розрахунку та вплив екстремального значення менше. Спосіб полягає в наступному:

$$Z_{ij} = \frac{X_{ij}}{\sqrt{\Sigma(X_{ij})^2}} \quad (6)$$

Оскільки різний атрибут ресурсу має інший вимір, використовуємо формулу (3.6) для розміру та стандартизуємо атрибути статичного ресурсу, як показано нижче:

$$p_i = \frac{p_i * k_i}{\sqrt{\sum_{j=1}^{u}(d_j * k_j)^2}} \; m_i = \frac{m_i}{\sqrt{\sum_{j=1}^{u} m_j{}^2}}$$

$$d_i = \frac{d_i}{\sqrt{\sum_{j=1}^{u} d_j{}^2}} \; c_i = \frac{c_i}{\sqrt{\sum_{j=1}^{u} c_j{}^2}} \qquad (7)$$

У формулі 3.7. p означає частоту процесора, k означає число ядер, m – об'єм пам'яті, d – пропускна спроможність периферійного зберігання, а c – пропускна спроможність; i={1,2,…,u} означає всі доступні обчислюванні ресурси.

Розглянемо кількісне розміщення динамічних атрибутів. Оцінка перевантаження усуває обчислювальні ресурси з великим завантаженням і

визначає число хостів, які мають кращу продуктивність в системі та є обчислювальними вузлами з відносно легким навантаженням. Алгоритм обирає оптимальні обчислювальні вузли для подання завдань з них за допомогою динамічної оцінки. У даній роботі було використовуємо алгоритм WLC. Завдяки наявності зваженого коефіцієнта застосування модифікованого алгоритму Least Connections визначає ключову роль у представленому способі.

Перед динамічною оцінкою необхідно стандартизувати динамічну інформацію обчислювальних вузлів. Оскільки кожен обчислювальний процесор може мати кілька ядер, процес індикаторів виглядає наступним чином:

$$p_i'(t) = \sum_{i=1}^{k_i}[(1 - p_i(t)) * p_i] \quad (8)$$

У формулі k_i означає основний номер хоста N_i.

Для розрахунку стандартизованого значення індикатора динамічного атрибута одиночного обчислюваного ресурсу, використовується формула (3.7), (3.8) та (3.9), виглядає наступним чином:

$$p_i(t) = \frac{p_i * k_i}{\sqrt{\sum_{j=1}^{g}(p_i'(t))^2}} \quad m_i(t) = \frac{\left(1 - m_i(t)\right) * m_i}{\sum_{j=1}^{g}[\left(1 - m_j(t)\right) * m_j]^2}$$

$$d_i(t) = \frac{(1-d_i(t))*d_i}{\sqrt{\sum_{j=1}^{g}[\left(1-d_j(t)\right)*d_j]^2}} \quad c_i(t) = \frac{(1-c_i(t))*c_i}{\sqrt{\sum_{j=1}^{g}[\left(1-c_j(t)\right)*c_j]^2}} \quad (9)$$

Завантаження інформаційної системи зворотного зв'язку. Після фільтрування ресурсів за вимогою та квантованості статичних атрибутів використовуємо вікно перехоплення продукту *v*, щоб виключити обчислювальні вузли з низькою продуктивністю та виділити *l* резервні обчислювальні ресурси з *v* ресурсів з кращою продуктивністю, використовуючи стохастичну вагу.

У процесі вибору резервних ресурсів коефіцієнт пропорції *y* безпосередньо впливає на розподіл ймовірності видобутку ресурсів. Коли *y* дорівнює 1, то ймовірність вибору сортованих обчислювальних ресурсів для резервування близька; коли λ відстає від 1, то ймовірність вибору відмінного обчислювального вузла у фронті буде збільшуватися. Наприклад, коли *v* беруться 4, λприйнято 0,5 , можна отримати вірогідність вибору чотирьох сортованих обчислювальних ресурсів відповідно:

$$a_1 = \frac{8}{15} a_2 = \frac{4}{15} a_3 = \frac{2}{15} a_4 = \frac{1}{15} \qquad (10)$$

У наведеному вище прикладі ймовірність ресурсу з найбільш відмінними статичними атрибутами становить більше 0,5.

Алгоритм Weighted Least Connections є динамічним, тому що його використовує кожна перевантажена станція і контролюється він з головної станції. Алгоритм зважених найменших зв'язків використовує «ваговий» компонент оснований на відповідних можливостях кожного сервера. Для цього необхідно заздалегідь зазначити «вагу» кожного сервера [3].

Балансувальник навантаження, який використовує алгоритм зважених найменших зв'язків враховує наступні параметри: вагу (ємність) кожного сервера та поточну кількість клієнтів, які в даний час мають підключення до кожного із серверів. Вибір вузла системи цим способом балансування навантаження виконується на основі кількості активних з'єднань, тобто потужності сервера. Алгоритм WLC є динамічним та здатним швидко назначити «вагу» серверам.

Недоліки алгоритму WLC наступні: немає ніякого сенсу застосовувати цей алгоритм для задач з короткими сесіями, які характерні, наприклад, для HTTP протоколів. Для таких видів завдань краще застосовувати Round Robin.

Переваги алгоритму наступні: алгоритм WL використовується для завдань, які пов'язані з тривалими з'єднаннями. Наприклад, для розподілу навантаження між серверами бази даних. Якщо на деяких вузлах буде занадто багато активних підключень, нових запитів вони вже не отримають, і навпаки.

Розглянемо розрахунок часу обробки при навантажувальному тестуванні: виділяється завдання віртуальній машині за алгоритмом планування, після чого проводиться моніторинг системи віртуальних машин. В результаті маємо можливість визначити стан кожної віртуальної машини, а також з'ясувати необхідність балансування навантаження – перенесення завдань з перевантаженої віртуальної машини до менш завантаженої. Для досягнення максимальної оптимізації та виявлення достовірних результатів необхідно збільшити кількість повідомлень і їх довжину. Граничним значенням стає значення перевантаження всієї системи.

Середній коефіцієнт використання ресурсів (Average Resource Utilization Ratio, ARUR): головна мета розподілу навантаження полягає у тому, щоб була можливість максимально використовувати ресурси. Середній коефіцієнт використання ресурсів розраховуємо за формулою:

$$\text{ARUR} = \left(\frac{\text{середній час}}{\text{час обробки}}\right) * 100 \qquad (11)$$

де середній час = Σ часу, витраченого ресурсом (VM_j), щоб закінчити всю роботу ресурсу, де j={1,2,3,…M}.

Діапазон середнього коефіцієнта використання ресурсів становить від 0 до 1, максимальне значення для ARUR становить 1 (використання ресурсу становить 100%), найгірше – 0 (ресурс знаходиться в ідеальному стані).

Висновки. У даній статті визначено алгоритм, який зосереджується на збільшенні навантаження, щоб задовольнити вимоги користувача для

продовження сервісу шляхом розподілу робочого навантаження в балансовому порядку, щоб отримати максимальне використання ресурсів та зменшити час простою системи. Також представлений метод динамічного балансування навантаження у хмарному середовищі, який має за основу метод міграції задач та модернізований алгоритм Weighted Least Connections, тобто алгоритм динамічного навантаження, який враховує час проходження та середній коефіцієнт використання ресурсів як параметра. Головна мета даного методу є ефективне використання ресурсів у хмарному середовищі.

Проаналізовано головні переваги даного способу, зосереджуючись на ефективному використанні ресурсів у хмарному середовищі. Також розглянуто міграційні процеси при балансуванні навантаження у хмарному середовищі як особові об'єкти. Тому запропоновано виокремити показник, який визначає вплив міграції на мінімізацію загальної вартості мережі.

Література

1. Кононюк А.Е. Фундаментальная теория облачных технологий / «Освіта України». – Київ 2018. - С. 11–51.
2. Андрей Емельянов «Балансировка нагрузки: основные алгоритмы и методы» [Електронний ресурс]. – Режим доступу: https://blog.selcctcl.in/balansirovka-nagruzki-osnovnve-algoritmv-i-metody/
3. Батура Т.В. Software & Systems Программные продукты и системы / Мурзин Ф.А., Семич Д.Ф. / Облачные технологии: основные модели, приложения, концепции и тенденции развития Сборник. – 2014. – №3 (107).
4. Бершадский А.М. Исследование стратегий балансировки нагрузки в системах распределенной обработки данных / А.М. Бершадский, Л.С.

Курилов, А.Г. Финогеев // Известия высших учебных заведений. – 2018. – № 4 (12).

5. Гасюк А.А. Инновации в облачных технологиях – одна из основ экономического роста компаний / Электронный научный журнал. - 2016. - № 9 (12). - С. 88-91.
6. Джордж Риз: Облачные вычисления. – BHV-СПб, 2011. – 288 с., ISBN: 978-5-9775-0630-4
7. Дорожкин С.К. Методы оценки загрузки вычислительных узлов распределенной вычислительной системы / С.К. Дорожкин // Научно-технический вестник информационных технологий, механики и оптики. – 2016. – Вып. 14. – С. 140–144.
8. Журнал о системах электронного документооборота (СЭД) ECM-Journal 2018 г. [Електронний ресурс]. – Режим доступу: https://ecm-journal.ru/docs/TOP-10-oblachnykh-tendencii-v-2018-godu.aspx
9. Иванисенко И.Н. Анализ методов балансировки нагрузки в распределённых системах / И.Н. Иванисенко, Т.А. Радивилова // Сучасні напрямки розвитку інформаційно-комунікаційних технологій та засобів управління: П'ята міжнар. наук.-техн. конф., 23 – 25 квіт. 2015: матер. конф. – Полтава – Баку – Кіровоград – Харків, Україна. – С. 20-21.
10. Иванисенко И.Н. Балансировка нагрузки в облачных сервисах с учетом самоподобных свойств входящих потоков / И.Н. Иванисенко // Радиоэлектроника и молодежь в XXI веке: Междунар. молодеж. форум, 14 –16 апр. 2014: матер. форума. – Харків, Україна. – Том 5. – С. 74-76.
11. Иванисенко И.Н. Методы балансировки с учетом мультифрактальных свойств загрузки / И.Н. Иванисенко, Л.О. Кириченко, Т.А. Радивилова // Information content and processing. – 2015. – Vol. 2 (4). – P. 345-368.

12. Иванисенко И.Н. Методы решения задачи балансировки вычислительной нагрузки в сети распределенных вычислений / И.Н. Иванисенко // Ин-формационные технологии в навигации и управлении: состояние и перспективы развития: Первая научно-техн. конф., 5 – 6 июля 2010: матер. конф. – Киев, Украина, 2010. – С. 26.
13. Игнатенко Е.И. Адаптивный алгоритм мониторинга загруженности сети кластера в системе балансировки нагрузки / Е.И.Игнатенко, В.И.Бессараб, И.В.Дегтяренко // Наукові праці ДонНТУ. – 2011. – Вип. 21(183). – С. 95-102.
14. Іванісенко І.М. OPEN SOURCE продукти балансування навантаження / І.М. Іванісенко // Free and Open Source Software: VII Всеукр. наук.-практ. конф., 24 – 27 лист. 2015: матер. конф. – Харків, Україна. – С. 85.
15. К. А. Баркалов, В. В. Рябов, С. В. Сидоров, О некоторых способах балансировки локального и глобального поиска в параллельных алгоритмах глобальной оптимизации, Выч. мет. программирование, 2015, том 11, выпуск 4, 382 – 387
16. Калягин И.Н., Воронцов А.А. Использование облачных технологий для бизнеса / Международный студенческий научный вестник. 2016. - № 3-1. - С. 79–80.
17. Кашицин И.М., Сальников И.И. Применение облачных технологий / Международный студенческий научный вестник. 2016. № 3-1. - С. 81-82.
18. Клементьев И.П., Устинов В. А.: Введение в Облачные вычисления. – УГУ, 2015. – 233 с.
19. Крікет Лі. DNS BIND Керівництво для системних адміністраторів / Крікет Лі, Пол Альбітц. – Символ-Плюс, 2017. – 709 с.

20. Медведев А. Облачные технологии: тенденции развития, примеры исполнения / Современные технологии автоматизации. - 2013. - № 2. - С. 6-9.
21. Питер Фингар: «DOT. CLOUD. Облачные вычисления – бизнес-платформа XXI века», Аквамариновая Книга, 2011. – 256 с., ISBN:978-5-904136-21-5
22. Радивилова Т.А. Динамический метод оценки загрузки узлов распределенной системы / И.Н. Иванисенко, Т.А. Радивилова // Проблеми телекомунікацій. – 2016. – № 1(18). – С. 42-51.
23. Сергей Зубов «Сравнительный анализ методов балансировки трафика» [Електронний ресурс]. – Режим доступу: https://habr.eom/companv/oleg-bunin/blog/319262/
24. Спирин Г.М. Анализ эффективности работы облачных компьютерных технологий в работе компании / Электронный научный журнал. - 2016. - №4. - С. 160-163.