УДК 621.391
© Б. Стрихалюк[2], І. Демидов[1], З. Хархаліс[1], 2014

СТРУКТУРНО-ФУНКЦІОНАЛЬНА ОПТИМІЗАЦІЯ ПРОЦЕСІВ МІГРАЦІЇ ВІРТУАЛЬНИХ МАШИН В РОЗПОДІЛЕНИХ ДАТА-ЦЕНТРАХ

Представлено структурно-функціональний підхід до оптимальної консолідації віртуальних машин на основі балансування кількома ресурсними параметрами серверів розподіленої сервісної платформи, що з одного боку дозволяє підвищити енергоефективність хмаринкових центрів оброблення даних, а з іншого боку – їх адаптованість до динамічних змін навантаження з боку користувачів, як номадичних, так і рухомих, за умов збереження належної якості сервісу.

A structural-functional approach to optimal consolidation of virtual machines based on multiple servers' resource settings balancing at distributed service platform is presented, that on the one hand can increase the energy efficiency of cloud-based data processing centers, on the other hand - their adaptability to dynamic workload changes from nomadic users, as well as moving ones, while preserving adequate quality of service.

1. ВСТУП

Всі основні тенденції розвитку інформаційних технологій, такі як хмарні обчислення базуються на великих та потужних обчислювальних системах. Постійно зростаючий попит на сервіси, що надаються хмаринковими обчислювальними системами став причиною того, що компанії та постачальники ресурсів змушені будувати масштабні центри обробки даних, які функціонують на великих апаратних потужностях, а отже, споживають багато енергії. В 2006 році енергія, що була спожита галуззю IT в США становила близько 61 мільярдів кВт·год, що становило 1.5% всієї виробленої у світі електричної енергії, що відповідає 2% світового викиду оксиду вуглецю, а це, у свою чергу, еквівалентне сукупним викидам CO_2 авіаційної галузі. Очікується, що ці цифри збільшуватимуться удвічі кожні 5 років [1].

2. ОСНОВНА ЧАСТИНА

[1] Національний університет «Львівська політехніка».

Протягом останніх кількох років відбувся значний прорив в контексті зменшення споживання енергії, в основному, за рахунок підвищення ефективності енергопостачального та охолоджувального обладнання в центрах обробки даних. Ще декілька років тому коефіцієнт ефективності використання енергії, який визначається, як відношення підведеної потужності до потужності, яка використана обчислювальними ресурсами центру обробки даних, лежав в межах між 2 та 3, а в даний час великі компанії, які спеціалізуються на хмарних обчисленнях досягають величин цього коефіцієнта менших за 1.1. Проте, все ще існує багато можливостей оптимізації самих обчислювальних ресурсів. Було підраховано, що сервери більшість часу працюють на 10-50% своєї повної потужності [2, 3]. Таке низьке використання потужностей, зокрема, зумовлене мінливістю навантаження на віртуальні машини – центри обробки даних спроектовані так, щоб витримувати піки навантажень, хоча протягом довгих періодів часу навантаження є значно меншим [4, 5]. Оскільки, ввімкнений, але бездіяльний сервер споживає близько 50-70% від тої потужності, яку він споживає при повній власній завантаженості [6], стає очевидним, що значна кількість енергії споживається навіть при низькому завантаженні.

Для вирішення цієї проблеми може бути використана парадигма віртуалізації в рамках розв’язання задач структурно функціонально-оптимізації віртуальних машин відповідної сервісної платформи. Такий підхід дає можливість консолідації навантаження, яка полягає в розміщенні максимальної кількості віртуальних машин на мінімальній кількості фізичних серверів [7]. Консолідація дозволить переводити незавантажені сервери в режим зниженого енергоспоживання чи відімкнути взагалі (що приведе до економії енергії та скорочення експлуатаційних витрат), або ж використати їх для обслуговування додаткових навантажень (що приведе до зменшення капітальних витрат). Нажаль, ефективна консолідація віртуальних машин утруднена успадкованою складністю проблеми. Оптимальне призначення віртуальних машин серверам центру обробки даних є аналогом задачі «про упаковку в контейнери» недетермінованої поліноміальної складності, суть якої полягає в розміщенні змінного набору предметів у мінімальній кількості мішків з певного їх набору. Дана проблема ускладнена двома обставинами: 1) призначення віртуальних машин має враховувати декілька параметрів сервера, наприклад параметри продуктивності процесорів (надалі CPU) і обсяг оперативної пам’яті (надалі RAM), внаслідок цього задача стає «багатовимірною задачею про упаковку в контейнери»; 2) навіть при вдалому призначенні віртуальні машини безперервно змінюють свої

апаратні потреби, тобто попередня конфігурація їх призначення стає неактуальною вже за декілька годин.

Розглянемо підхід до консолідації віртуальних машин на основі вирішення багатовимірної проблеми, а саме представлений конкретним випадком, в якому віртуальні машини консолідуються стосовно двох ресурсів серверної платформи: продуктивності CPU і обсягу оперативної пам'яті RAM. Причому, віртуальні машини консолідуються за допомогою двох типів імовірнісних методик-процедур – для їх призначення та міграції відповідно. Обидві методики націлені на підвищення рівня використання серверів та динамічну консолідацію навантаження з дотриманням відповідної якості обслуговування користувачів за умови мінімізації енергетичних витрат. Всі ключові операції покладені на окремі сервери, в той час, як менеджер або гіпервізор центру обробки даних повинен лише правильно поєднувати локальні структурно-функціональні конфігурації цих серверів. Метод, який лежить в основі нашого підходу насамперед заснований на «мурашиних» алгоритмах, які спершу використовував Деньюберг [8], а потім і широке наукове співтовариство для моделювання поведінки мурашиних колоній та вирішення багатьох складних розподілених проблем. Характеристики, які надають мурашині алгоритми для даного підходу, роблять його ефективним та особливим по відношенню до інших рішень. Наведемо деякі з цих характеристик: 1) використання парадигми колективного інтелекту, яка дозволяє вирішувати складні проблеми за допомогою простих операцій, що виконуються великою кількістю автономних дійових осіб (в нашому випадку – окремих серверів); 2) використання імовірнісної методики в формі моделей поведінки реальних мурах; 3) самоорганізаційна поведінка системи, яка гарантує, що призначення конфігурації віртуальних машин серверам динамічно адаптуватиметься до змінного навантаження.

З метою оцінювання ефективності запропонованого підходу за рівнем використання ресурсів розподіленої сервісної системи ми використовуємо два взаємодоповнюючих напрямки досліджень. Перший напрямок пропонує математичну модель, яка простежує еволюцію системи протягом певного часу базуючись на тому, що залучені змінні є неперервними. Така модель дозволяє нам перевірити поведінку хмаринкової сервісної платформи в умовах великої кількості різноманітних сценаріїв, просто змінюючи значення деяких її параметрів. Другий напрямок полягає в експериментальному вивченні, яке провадиться на основі реальних центрів обробки даних. Ці два напрямки доповнюють один одного: аналітична модель вводить деякі спрощення, але дозволяє легко дослідити систему на широкому

діапазоні сценаріїв; з іншого боку, реальні експерименти не вносять ніяких спрощень, але в певній мірі обмежуються конкретною ситуативністю, що відчутно зменшує кількість сценаріїв тестування. Дослідження показали, що хмаринкова сервісна платформа розподіленого дата центру, що конфігурується на основі запропонованого підходу дозволяє досягає дуже хорошої консолідації, та плавно адаптується до можливих змін. І на сам кінець, для того щоб порівняти ефективність сконфігурованої хмаринкової системи за новим підходом з системами, які організовані згідно базового підходу, описаного в [1], а також для того щоб провести дослідження системної масштабованості ми використали спеціально підібраний симулятор.

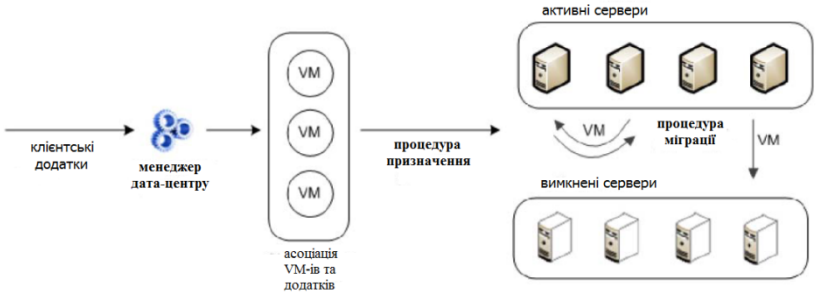

Рис. 1. Призначення та міграція віртуальних машин в розподіленому хмаринковому центрі обробки даних сервісної платформи

Основною метою запропонованого підходу є динамічне призначення віртуальних машин серверам з метою зменшення енерговитрат (за рахунок консолідації віртуальних машин, що дає змогу помістити деякі сервери в енергозберігаючий режим), за умов аналогічної або навіть кращої якості сервісу. Сценарій схематично зображено на рис.1: запит програмного додатку передається від клієнта до менеджера (або гіпервізора) центру обробки даних, який обирає віртуальну машину, що підходить даному програмному додатку на основі його системних потреб (показників CPU, RAM, дискового простору), а також параметрів клієнта, наприклад типу клієнтської операційної системи. Після цього віртуальна машина призначається одному з існуючих серверів за допомогою процедури призначення.

Основна ідея всього підходу полягає в тому, що сам сервер повинен вирішити, чи слід йому прийняти або ж відхилити віртуальну машину. Таке рішення приймається на основі локальної інформації наявної на сервері – наприклад, інформації про поточне ресурсне завантаження CPU і RAM – і ґрунтується на методиці випробувань

Бернуллі. Гіпервізор центру обробки даних виконує лише координуючу роль, яка не потребує складних централізованих алгоритмів для оптимізації процесів призначення віртуальних машин.

Навантаження, яке припадає на кожен програмний додаток є динамічним, або, іншими словами, обчислювальні ресурси, які необхідні для правильної роботи такого програмного додатку, змінюються з часом: наприклад, потреби щодо продуктивності CPU web-сервера залежать від навантаження, яке генерується його користувачами. Таким чином, призначення віртуальних машин постійно контролюється і налаштовується через процедуру міграції. Міграція віртуальної машини може бути вигідною, як у випадку, коли використання обчислювальних ресурсів є надто малим, так і у випадку, коли їх використання є надто високим (що може привести до ситуації перевантажень та порушення якості сервісу). Міграція також може виникати у розподіленій хмаринковій системі тоді, коли користувачі починають концентрувати навантаження у деякій її просторовій частині. Процедура міграції складається з двох етапів: на першому етапі сервер запитує міграцію віртуальної машини, на основі інформації про використання ресурсів його CPU/RAM. Метою наступного кроку є обрання сервера, який буде обслуговувати віртуальну машину, що мігрує за допомогою методів, подібних до тих, які використовуються при процедурі призначення.

Ефективність технології конфігурування хмаринкових сервісних систем оцінюється за допомогою таких метрик:

- *Використання обчислювальних ресурсів.* З метою сприяння консолідації та енергозбереженню, сервер або повинен використовуватись майже на повну потужність, або знаходитись в режимі сну. Аналіз використання ресурсів CPU і RAM має на меті перевірку того, чи досягнута дана мета.
- *Кількість активних серверів.* Віртуальні машини повинні бути розміщені на найменшій можливій кількості серверів.
- *Спожита потужність.* Кінцевою метою є економія енергоресурсів, тому ми обчислюємо потужність, спожиту всім центром обробки даних в різних станах його завантаження.
- *Частота міграцій та змін режимів роботи серверів.* Міграція віртуальної машини викликає невелике зниження продуктивності програмного додатку, що нею обслуговується. Час, який необхідний для того, щоб передати пам'ять віртуальної машини з одного сервера на інший може варіюватися в межах від декількох секунд до двох хвилин [9, 10]. Під час такого періоду віртуальна машина є активною на вихідному сервері. Протягом фактичного передавання віртуальної

машини відбувається її простій, тривалість якого декілька мілісекунд. Аналогічно, для ввімкнення/вимкнення сервера потрібен час та додаткова потужність. Тому, хоча міграції та зміни режимів роботи серверів і мають важливе значення для консолідації віртуальних машин та зменшення споживаної потужності, важливим також є зменшення частоти їх виникнення. Ще важливішим є недопускання великої кількості міграцій віртуальних машин: асинхронна та поступова їх міграція є значно менш згубною, ніж паралельна міграція такої ж кількості віртуальних машин; наприклад, паралельні міграції можуть згенерувати трафік, який перевищить пропускну здатність каналів сервісної платформи, а отже збільшиться час простою.

- *Порушення угоди про рівень обслуговування (Service Level Agreement, SLA).* Порушення угоди про рівень обслуговування може відбутися тоді, коли навантаження на деякі віртуальних машини зростає і сервери, якими вони обслуговуються, стають перевантаженими. Появі такої ситуації можна запобігти шляхом своєчасного перенесення деяких віртуальних машин на інші, менш завантажені сервери. Ця метрика, відповідно до нещодавніх досліджень [1], використовується для оцінки рівня QoS послуг, які пропонуються користувачам.

Декілька досліджень та експериментів, наприклад [6, 11], показали, що активний сервер з дуже низьким рівнем використання CPU споживає близько 50-70% тої потужності, яку він споживає при повній своїй завантаженості. Більше того, коли використання CPU зростає, споживану потужність можна описати з похибкою меншою ніж 10%, за допомогою лінійної функції починаючи від потужності, яка відповідає стану бездіяльності, до потужності, яка відповідає стану повного завантаження сервера [12, 13]. Хоча деякі дослідження вивели більш точні нелінійні залежності [14], такі уточнення вносять мало практичної користі для наших цілей. Таким чином, в аналітичних та імітаційних експериментах, потужність, яка споживається одним сервером виражається як:

$$P(u) = P_{прост} + (P_{макс} - P_{прост})u \qquad (1)$$

де $P_{макс}$ – потужність, яка споживається про повному завантаженні CPU, тобто коефіцієнт утилізації $(u = 1)$, $P_{прост}$ – потужність, яка споживається тоді, коли сервер є активним, але в стані простою $(u = 0)$. В експериментах, які проводяться в реальних центрах обробки даних, споживана потужність безпосередньо відслідковується та вимірюється.

Як вже було зазначено, існує дві основні ймовірнісні процедури, на яких базується запропоновуй технологічний підхід, а саме процедури *призначення* та *міграції*. Основним критерієм розміщення віртуальних машин на серверах є наявні в них вільні ресурси CPU та RAM.

Процедура призначення виконується тоді, коли клієнт запускає в центрі обробки даних новий програмний додаток. Отже, коли програмний додаток вже пов'язаний з сумісною віртуальною машиною, менеджер-гіпервізор центру обробки даних повинен призначити дану віртуальну машину одному з серверів. Менеджер, замість того, щоб самостійно прийняти рішення (що потребувало б застосування складних оптимізаційних алгоритмів), передає основну частину процедури самим серверам. Зокрема, він відправляє запрошення всім активним серверам, або тільки їх частині (в залежності від розміру хмари та архітектури розподіленого центру обробки даних). Воно полягає в команді перевірити, чи вони мають можливість прийняти нову віртуальну машину. Кожен сервер приймає рішення щодо того, відхилити, чи прийняти запрошення і тим самим зробити свій вклад до процесу консолідації навантаження на мінімально можливій кількості серверів. Запрошення повинно бути відхиленим, якщо сервер перевантажений або недостатньо завантажений з точки зору будь-якого з двох розглянутих ресурсів – CPU чи RAM. У випадку перевикористання ресурсів, основною причиною відмови від запрошення є уникнення ситуацій перевантаження, яке може зменшити якість сервісу, який отримують користувачі, а у випадку недовикористання ресурсів – сервер повинен відхилити запрошення, та, *водночас*, спробувати відмовитись від тих віртуальних машин, які на ньому працюють, оскільки йому варто перейти в режим сну для економії енергії. В іншому випадку, коли рівень використання обчислювальних ресурсів сервера є проміжним, він повинен прийняти запрошення на обслуговування нової віртуальної машини для покращення консолідації.

Сервер приймає рішення на основі випробування Бернуллі. Імовірність успіху такого випробування рівна величині загальної функції ймовірності призначення, що, в свою чергу, визначається оцінюванням функцій ймовірностей призначень для кожного розглянутого ресурсу. Якщо χ (величина між 0 та 1), яка є відносним рівнем використання ресурсу (CPU чи RAM), а T є максимальним дозволеним рівнем використання (наприклад $T = 0,8$ означає, що рівень використання ресурсу не повинен перевищувати 80% обсягу

ресурсу цього сервера), функція призначення рівна нулю коли $\chi > T$, інакше вона обчислюється як:

$$f(\chi, p, T) = \frac{1}{M_p} \chi^p (T - \chi), \qquad 0 \le \chi \le T \qquad (2)$$

де p є параметром розподілу, а дільник M_p використовується для нормалізації максимального значення до 1 і визначається як:

$$M_p = \frac{p^p}{(p+1)^{p+1}} T^{(p+1)}. \qquad (3)$$

На рис. 2 показано графік для функції призначення з єдиним ресурсом для деяких значень p та $T = 0,9$. Параметр p можна використовувати для зміни форми залежності функції в деяких межах. Насправді, величина χ, при якій функція досягає свого максимуму (тобто значення, при якому спроба призначення буде успішною з найбільшою ймовірністю) рівна $p/(p+1)T$, тобто збільшується і наближається до T, коли значення p збільшується.

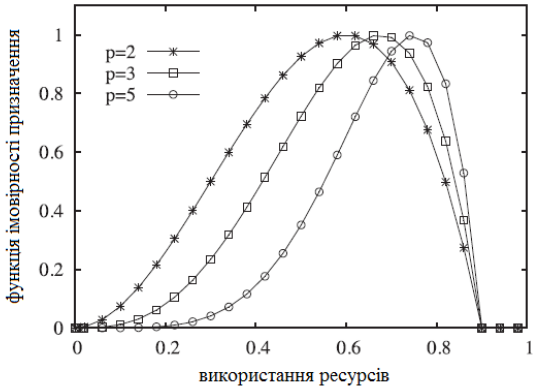

Рис. 2. Функція ймовірності призначення $f(\chi, p, T)$ *для трьох різних значень параметру* p *та* $T = 0,9$.

Значення даної функції рівне нулю, або є дуже низьким, коли обчислювальний ресурс або перевикористовується, або недовикористовується.

Нехай u_s та m_s є поточними рівнями використання ресурсів CPU та RAM сервера s відповідно, тоді загальна функція призначення буде рівна добутку двох функцій призначення (2), в яких $x = u_s$ та $x = m_s$.

Також, припустимо, що p_u та p_m є параметрами розподілу обчислювальних ресурсів, які розглядаються, а T_u та T_m – їх максимальними рівнями використання, відповідно. Тоді загальна функція призначення для сервера s, яку позначимо як f_s, буде виглядати як:

$$f_s(u_s, m_s, p_u, p_m, T_u, T_m) = f(u_s, p_u, T_u) \cdot f(m_s, p_m, T_m). \quad (4)$$

Форма функцій призначення, об'єднана з визначенням функцій (4), гарантує, що сервери повинні позитивно відповідати на запрошення прийняти віртуальну машину, коли рівень завантаження їх ресурсів RAM та CPU є проміжним, а якщо один з ресурсів недо- або перевикористовується, то ймовірнісне значення, яке повертає випробування Бернуллі буде низьким.

Якщо випробування Бернуллі пройдено успішно, то сервер повідомляє менеджеру (гіпервізору) центру обробки даних про свою готовність обслуговувати віртуальну машину. Після цього, менеджер просто вибирає один з таких наявних серверів і призначає йому дану нову віртуальну машину. Для цього він може використати оцінки взаємопов'язаних параметрів серверів, обрані, зокрема, за принципами використання теорії нечітких множин [15]. Якщо ж немає жодного наявного сервера, тобто всі випробування Бернуллі пройшли невдало, то досить імовірно, що у всіх серверах один з двох ресурсів (CPU або RAM) близький до порогу використання. Таке зазвичай відбувається тоді, коли загальне навантаження збільшується настільки, що поточна кількість активних серверів не є достатньою для підтримки такого навантаження [15]. В такому випадку, менеджер виводить з режиму сну один з неактивних серверів та доручає йому обслуговування нової віртуальної машини. Якщо має місце випадок, коли немає сервера, який можна вивести з режиму сну, оскільки всі наявні сервери вже є активними, то це означає, що всі вони разом не в змозі обслуговувати таке навантаження навіть при його консолідації, і компанії слід розглянути можливість встановлення нових серверів.

У випадку коли віртуальні машини збільшують свої вимоги, сервер може стати перевантаженим, що ймовірно призведе до подій пов'язаних із порушенням угоди про рівень сервісу, які негативно вплинуть на репутацію центру обробки даних. В обох випадках, а саме недовикористання та перевикористання сервера, деякі віртуальні машини можуть вигідно мігрувати до інших серверів, для того щоб дати змогу виключити його, або зменшити його завантаженість.

Процедура міграції відбувається наступним чином: кожен сервер відслідковує рівень використання своїх ресурсів CPU та RAM за

допомогою бібліотек, які постачаються разом із інфраструктурою віртуалізації (наприклад VMWare чи Hyper-V) і перевіряє, чи ці рівні лежать в межах попередньо визначених порогів – нижній поріг $T_н$ та верхній поріг $T_в$. Коли такий стан порушується, сервер оцінює відповідну функцію ймовірності $f^{н}_{мігр}$ чи $f^{в}_{мігр}$ та проводить випробування Бернуллі, імовірність успіху якого присвоюється значенню вибраної функції. Якщо випробування пройшло вдало, то сервер запитує міграцію однієї з його локальних віртуальних машин. Позначивши використання одного з обчислювальних ресурсів (CPU чи RAM) як χ, функції ймовірності міграцій будуть виглядати наступним чином:

$$f^{н}_{мігр} = \left(1 - \frac{\chi}{T_н}\right)^{\alpha} \tag{5}$$

$$f^{в}_{мігр} = \left(1 - \frac{\chi - 1}{1 - T_в}\right)^{\beta} \tag{6}$$

Функції, графіки яких зображені на рис. 3 підібрані таким чином, щоб викликати міграцію віртуальних машин тоді, коли використання обчислювальних ресурсів стає нижче порогового рівня $T_н$ або вище порогового рівня $T_в$ відповідно. Обидва види міграції в подальшому будуть згадуватись як «низькі міграції» і «високі міграції». Форма залежностей даних функцій може бути змінена за допомогою підбирання параметрів α та β, внаслідок чого можна сприяти або перешкоджати міграціям відповідно. Дані функції є однаковими по відношенню до ресурсів CPU чи RAM, хоча параметри $T_н$, $T_в$, α та β можуть бути різними для цих двох обчислювальних ресурсів.

Після кожної ітерації, коли випробування Бернуллі завершилось успіхом, сервер повинен обрати віртуальну машину для її міграції. У випадку «високої міграції», сервер фокусується на перевикористаному ресурсі (CPU чи RAM) та розглядає віртуальні машини для яких рівень використання даного ресурсу є вищим, аніж різниця між поточним рівнем використання сервером даного ресурсу та його пороговим значенням $T_в$. Після цього, одна з таких віртуальних машин вибирається випадковим чином для її міграції, оскільки це надає можливість зробити меншим рівень використання даного

обчислювального ресурсу. У випадку «низької міграції» вибір віртуальної машини для її міграції може бути цілком випадковим.

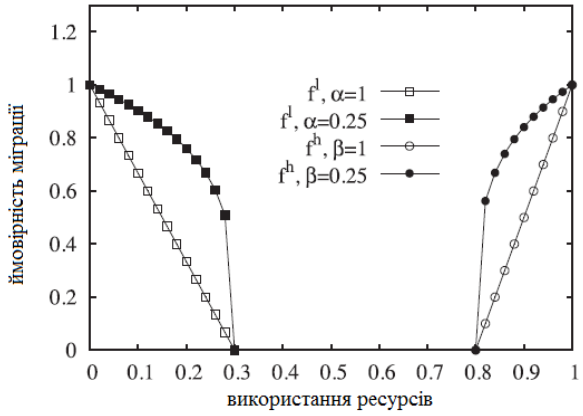

Рис. 3. Функції ймовірності міграції $f^{н}_{мігр}$ *чи* $f^{в}_{мігр}$ *(позначені як* f^{l} *та* f^{h} *) для двох різних значень параметрів* α *та* β *. На цьому прикладі нижній поріг* $T_{н} = 0,3$ *, а верхній поріг* $T_{в} = 0,8$ *.*

Процес вибору нового сервера, на якому розміщуватиметься віртуальна машина, що мігрує є подібним до процесу вибору віртуальної машини в процедурі призначення, проте з двома суттєвими відмінностями. Перша відмінність стосується міграції у зв'язку з перевантаженням, і полягає в тому, що пороговий рівень T функції призначення встановлюється в значення рівне 0.9 від рівня використання обчислювального ресурсу сервером, який ініціює процедуру міграції і розсилається всім іншим серверам разом із запрошенням прийняти на обслуговування віртуальну машину. Це гарантує те, що віртуальна машина мігрує до менш завантаженого сервера, і дозволить уникнути багаторазових міграцій однієї і тої ж віртуальної машини. Друга відмінність стосується процедури «низької міграції». Коли немає активного сервера для обслуговування віртуальної машини, що мігрує, буде неприйнятним ввімкнути новий сервер для розміщення її на ньому – оскільки недоречно вмикати якийсь сервер для того, щоб інший сервер перевести в режим глибокого сну. У зв'язку з цим, при «низькій міграції» – коли немає наявного сервера – міграція віртуальної машини не відбувається взагалі.

Варто відзначити, що запропонований підхід забезпечує поступовий та безперервний процес міграції, в той час, як більшість інших нещодавно запропонованих методів міграції вимагають одночасної міграції великої кількості віртуальних машин.

Порогові значення, як правило, можуть задаватись адміністратором центру обробки даних, на основі попереднього аналізу дисперсії завантаженості віртуальних машин. Параметри форми залежностей рис. 2-3 дозволяють адміністратору центру обробки даних вибирати між різними стратегіями консолідації (наприклад, консервативною, проміжною, агресивною). Більш агресивна стратегія дозволить перевести в режим сну більшу кількість серверів, водночас спричиняючи більшій кількості міграцій. Вибір потрібної стратегії проводиться шляхом підбору значень параметрів залежностей рис. 2-3. Оскільки аналіз стратегій консолідації виходить за рамки даної роботи, значення параметрів наведених залежностей відповідають проміжній стратегії консолідації. Очевидно, що для умов більш високої динаміки змін навантаження користувачів слід обирати більш агресивну стратегію консолідації віртуальних машин, а для випадку більшої просторової міграції користувачів відносно хмаринкової сервісної системи більше підійде стратегія міграції, що наближається до консервативної.

3. Висновки

У роботі запропоновано структурно-функціональний підхід оптимальної консолідації віртуальних машин на основі балансування кількома ресурсними параметрами серверів розподіленої сервісної платформи, що з одного боку дозволяє підвищити енергоефективність хмаринкових центрів оброблення даних, а з іншого боку – їх адаптованість до динамічних змін навантаження з боку користувачів, як номадичних, так і рухомих.

Для оцінки ефективності використання ресурсів розподіленої сервісної платформи ми пропонуємо використання двох взаємодоповнюючих напрямів досліджень. Перший напрямок пропонує аналітичну модель, яка простежує еволюцію системи протягом певного часу, базуючись на тому, що залучені змінні є неперервними. Другий напрямок орієнтується на експерименти, які виконуються на основі реальних центрів обробки даних. Отримані залежності дозволяють оцінити структурно-функціональні конфігурації віртуальних машин в процесі динамічної міграції в межах хмаринкових сервісних платформ з метою досягнення критеріїв

енергоефективності та дотримання належного рівня якості комплексних сервісів, що надаються, в умовах динамічних змін їх конфігурацій та навантаження користувачів.

1. A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-Aware Resource Allocation Heuristics for Efficient Management of Data Centers for Cloud Computing," Future Generation Computer Systems, vol. 28, no. 5, pp. 755-768, 2012. 2. L.A. Barroso and U. Hölzle, "The Case for Energy-Proportional Computing," IEEE Computer, vol. 40, no. 12, pp. 33-37, Dec. 2007. 3. G. Dasgupta, A. Sharma, A. Verma, A. Neogi, and R. Kothari, "Workload Management for Power Efficiency in Virtualized Data Centers," Comm. ACM, vol. 54, pp. 131-141, July 2011. 4. L. Hosman and B. Baikie, "Solar-Powered Cloud Computing Datacenters," IT Professional, vol. 15, no. 2, pp. 15-21, 2013. 5. M. Aggar, "Developers, Developers, Developers: Engaging the Missing Link in It Resource Efficiency," technical report, The Green Grid, Mar. 2013. 6. A. Greenberg, J. Hamilton, D.A. Maltz, and P. Patel, "The Cost of a Cloud: Research Problems in Data Center Networks," Proc. ACM SIGCOMM Computer Comm. Rev., vol. 39, no. 1, pp. 68-73, 2009. 7. M. Cardosa, M.R. Korupolu, and A. Singh, "Shares and Utilities Based Power Consolidation in Virtualized Server Environments," Proc. 11th IFIP/IEEE Integrated Network Management (IM '09), June 2009. 8. J.L. Deneubourg, S. Goss, N. Franks, A. Sendova-Franks, C. Detrain, and L. Chre´tien, "The Dynamics of Collective Sorting: Robot-Like Ants and Ant-Like Robots," Proc. First Int'l Conf. Simulation of Adaptive Behavior on from Animals to Animats, pp. 356-363, 1990. 9. T. Hirofuchi, H. Ogawa, H. Nakada, S. Itoh, and S. Sekiguchi, "A Live Storage Migration Mechanism over Wan for Relocatable Virtual Machine Services on Clouds," Proc. Ninth IEEE/ACM Int'l Symp. Cluster Computing and the Grid (CCGrid '09), pp. 460-465, May 2009. 10. H. Liu, C.-Z. Xu, H. Jin, J. Gong, and X. Liao, "Performance and Energy Modeling for Live Migration of Virtual Machines," Proc. 20th Int'l Symp. High Performance Distributed Computing (HPDC '11), pp. 171-182, June 2011. 11. A. Khosravi, S. Garg, and R. Buyya, "Energy and Carbon-Efficient Placement of Virtual Machines in Distributed Cloud Data Centers," Proc. 19th Int'l Conf. Parallel Processing (Euro-Par '13), 2013. 12. M. Mazzucco, D. Dyachuk, and R. Deters, "Maximizing Cloud Providers" Revenues via Energy Aware Allocation Policies," Proc. 10th IEEE/ACM Int'l Symp. Cluster Computing and the Grid (CCGrid '10), pp. 131-138, May 2010. 13. S. Rivoire, P. Ranganathan, and C. Kozyrakis, "A Comparison of High-Level Full-System Power Models," Proc. Conf. Power Aware Computing and Systems (HotPower '08), Dec. 2008. 14. X. Fan, W.-D. Weber, and L.A. Barroso, "Power Provisioning for a Warehouse-Sized Computer," Proc. 34th Ann. Int'l Symp. Computer Architecture (ISCA '07), pp. 13-23, June 2007. 15. Klymash M. A Novel Approach of Optimum Multi-criteria Vertical Handoff Algorithm for Heterogeneous Wireless Networks / M. Klymash, B. Stryhaluk, I. Demydov, M. Beshley, M. Seliuchenko // International Journal of Engineering and Innovative Technology (IJEIT). – November 2014. – Volume 4, Issue 5. – P. 42-52.