DOI 10.31891/2307-5732-2020-291-6-13-21
УДК 004.75

Б. В. ІВАНІЩЕВ, А. В. КАПЛУНОВ, В.В. РУСІНОВ, А. М. ВОЛОКИТА
Національний технічний університет України "Київський політехнічний інститут імені Ігоря Сікорського"

МЕТОД ПРИСКОРЕННЯ ВІДНОВЛЕННЯ КОЕФІЦІЄНТУ РЕПЛІКАЦІЇ В РОЗПОДІЛЕНОМУ СХОВИЩІ ДАНИХ НА ОСНОВІ МОНІТОРИНГУ РІВНЯ ДОВІРИ ДО ВУЗЛА

У цій роботі розглянуто проблему відновлення коефіцієнту реплікації в масштабованих розподілених сховищах при відключенні вузлів. Проведено аналіз open source механізму моніторингу. Запропоновано метод оцінки ступеня довіри до вузла, на базі оцінки метрик і виділення найбільш важливих їх типів. Запропоновано алгоритм складання плану міграції з вузлів з низьким ступенем довіри, оптимальний за критерієм часу відновлення коефіцієнту реплікації. Проведена експериментальна оцінка запропонованого методу.

Ключові слова: розподілене сховище даних, моніторинг, ступінь довіри, коефіцієнт реплікації, міграція даних.

B. IVANISHCHEV, A. KAPLUNOV, V. RUSINOV, A. VOLOKYTA
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"

METHOD OF ACCELERATING REPLICATION RATIO RENEWAL IN DISTRIBUTED DATA STORAGE BASED ON THE NODE'S TRUST LEVEL MONITORING

Considering the collective nature of the functioning of distributed computing systems the necessary mechanism for ensuring the quality of service is automatic dispatching, which is a set of control actions that optimize the operation of the entire system as a whole, carrying out operational planning and redistribution of running tasks based on the current state of the system. Further, progress of distributed computing systems will lead to a manifold increase in the number of resources and their diversity, which will lead to an aggravation of the existing problem.

The monitoring mechanism allows you to track changes in the state of system components, however, predicting the occurrence of emergency situations requires the implementation of a method for assessing the degree of trust in a node. A method of assessment based on assessing metrics and identifying their most important types is proposed. With the combination of these mechanisms, the need to reconfigure the storage becomes more transparent, which allows data balancing procedure to be performed even before the emergency shutdown of a problem node.

A method for drawing up a data migration plan with polynomial computational complexity was proposed which is based on allocating part of nodes into a separate subgraph and creating a migration plan for that subgraph firstly. The proposed method has been proven to be optimal in terms of migration time of data elements from nodes that are marked by the monitoring system as potentially unreliable. Optimization of the migration method by this criterion makes it possible to reduce the replication ratio recovery time in distributed storage by increasing total time of migration.

In this regard, the use of a storage reconfiguration mechanism based on an assessment of the degree of trust in a node is an urgent topic that is of scientific and practical interest. It has been shown experimentally that the proposed method is equally effective for distributed storages with a different number of storage devices and a different number of data items.

Keywords: distributed data storage, monitoring, level of trust, replication ratio, data migration

Постановка задачі

На сьогодні обсяг інформації, який генерується та використовується у світі, має стійку тенденцію до зростання. Це потребує постійного зростання обсягів сховищ даних, що можливо тільки коштом використання розподілених систем зберігання. Використання розподілених сховищ підіймає ряд питань, пов'язаних з надійністю зберігання даних і можливістю швидкого масштабування.

Сучасні розподілені сховища даних в більшості засновані на хмарній інфраструктурі IaaS (Infrastructure as a Service). Така інфраструктура дозволяє, не обмежуючись фіксованою кількістю пристроїв зберігання, тимчасово орендувати додаткові пристрої при необхідності. Та навпаки, звільняти пристрої, коли необхідність в них відпала. Також кількість пристроїв, що використовуються у розподіленому сховищі може зменшуватись при аварійному відключенні або деградації окремих пристроїв зберігання або частин мережі, які їх з'єднують.

Потреба в надійному зберіганні даних вирішується за допомогою зберігання в розподіленому сховищі декількох копій окремих елементів даних одночасно на різних пристроях зберігання, тобто шляхом реплікації даних. При цьому кількість копій кожного елементу даних в сховищі називається коефіцієнтом реплікації та має бути постійною у часі величиною. При плановому чи аварійному зменшенні кількості вузлів в розподіленому сховищі відбувається втрата деякої кількості копій елементів даних, тобто миттєва величина коефіцієнту реплікації зменшується та потребує найшвидшого відновлення.

Вчасне реагування на зміну стану розподіленого сховища загалом та окремих його вузлів (пристроїв зберігання) надає можливість швидко відновити коефіцієнт реплікації даних у сховищі та є запорукою надійного зберігання даних та забезпечення постійної доступності цих даних для користувачів. Тому гостро постають питання моніторингу стану розподіленого сховища, детекції аварійних ситуацій та вчасного реагування на ці ситуації для запобігання втрати даних з розподіленого сховища. Механізм моніторингу дозволяє відстежувати зміну стану компонентів системи, проте прогнозування виникнення аварійних ситуацій вимагає реалізації методу оцінки ступеня довіри до вузла, на базі оцінки метрик підконтрольної

системи і виділення найбільш важливих їх типів. За допомогою сукупності наведених механізмів необхідність реконфігурації сховища стає більш прозорою, що дозволяє підвищити контроль за середою виконання.

Відновлення коефіцієнта реплікації в розподіленому сховищі відбувається шляхом проведення процедури балансування сховища. Процедура балансування даних складається з задач розміщення та міграції даних. На першому етапі процедури балансування визначається оптимальне розміщення даних на пристроях зберігання в поточний момент часу, що вирішується задачею розміщення. Вихідними даними цієї задачі є відображення, в якому кожному елементу даних зіставляються пристрої зберігання, на яких цей елемент повинен бути розміщений після балансування.

Визначивши нове розміщення даних, а також знаючи попереднє розміщення, можна побудувати мультиграф вимог G, який описує переміщення даних [1]. Цей мультиграф є направленим і ациклічним:

$$\begin{gathered} G=(V,E,P)\,E\subseteq V\times V \\ E\subseteq V\times V \\ P:E\to N \\ \forall v,\omega\in V\Rightarrow P(v,\omega)=0,якщо(v,\omega)\notin E, \end{gathered} \tag{1}$$

де V – пристрої зберігання,

E – процедури переміщення даних між пристроями зберігання,

P – вагова функція мультиграфу (вартість переміщення даних).

Оскільки всі елементи даних мають фіксований розмір та однаковий час передачі, то план міграції можна розбити на кроки однакової тривалості. Завдання міграції полягає в складанні плану переміщення даних між пристроями зберігання згідно мультиграфу вимог (1) з мінімальною кількістю кроків. У такому вигляді ця задача зводиться до задачі розфарбовування дуг мультиграфу. При цьому кількість кроків у оптимальному плані міграції повинна дорівнювати мінімальній кількості кольорів розфарбування дуг мультиграфу, тобто хроматичному індексу мультиграфа.

Задача міграції в масштабованому сховищі є окремим випадком задачі міграції та має кілька критеріїв оптимізації. Одним з критеріїв є час міграції елементів даних, коефіцієнт реплікації яких не дорівнює цільовому, тобто час міграції даних, копії яких знаходяться на пристроях зберігання, помічених підсистемою моніторингу як потенційно ненадійні. Оптимізація за цим критерієм дозволяє швидко відновити коефіцієнт реплікації сховища даних, підвищуючи надійність зберігання даних. Іншим критерієм є загальний час міграції всіх елементів даних.

У статті запропоновано метод прискорення відновлення коефіцієнту реплікації в розподіленому сховищі даних, який використовує запропонований алгоритм міграції даних, що оптимізований за часом міграції з пристроїв, які помічені підсистемою моніторингу як потенційно ненадійні на основі оцінки рівня довіри до вузлів.

Аналіз досліджень та публікацій

Рішення для моніторингу вузлів. При виборі системи моніторингу розподіленої інфраструктури потрібно врахувати ряд факторів. В першу чергу оцінити відповідність функціоналу системи моніторингу вашим технічним і бізнес-вимогам, а також розглянути особливості розгортання та супроводу, щоб підібрати інструмент, що максимально відповідає вашій інфраструктурі.

Zabbix [2] - це універсальне рішення для моніторингу з відкритим вихідним кодом, що має підтримку технології єдиного входу, розподіленого моніторингу, Zabbix Insights (набір тригерних функцій для виявлення аномалій і неправильної поведінки системи), розширені налаштування для забезпечення безпеки, відсутність обмежень на зберігання даних і т.д. Zabbix сервер - центральний процес програмного забезпечення. Сервер виконує опитування даних, обчислює тригери, відправляє оповіщення користувачам. Він є центральним компонентом, якому агенти і проксі повідомляють дані про доступність і цілісність систем. Сервер може самостійно віддалено перевіряти мережеві служби (такі як веб-сервера і поштові сервери), використовуючи прості перевірки сервісів. Сервер є головним сховищем, в якому зберігаються всі конфігураційні, статистичні та оперативні дані, також він розсилає повідомлення адміністраторам в разі виникнення проблем з будь-яким з підконтрольних систем.

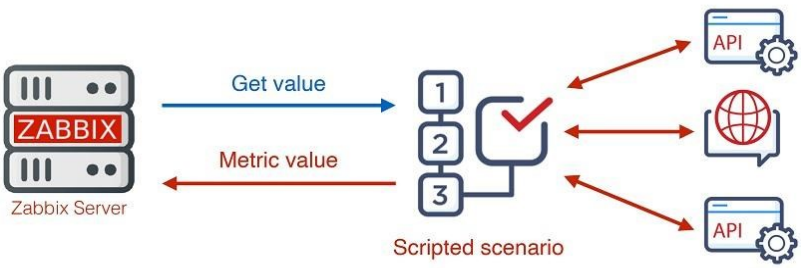

Рис. 1. Zabbix-сценарій комплексного багатоступеневого збору даних [2]

Агенти розгортаються на підконтрольних цілях для активного моніторингу локальних ресурсів і додатків (статистика жорстких дисків, пам'яті, процесорів і т.д.). Агент локально збирає оперативну інформацію і відправляє дані серверу для подальшої обробки. Ефективність агентів досягається завдяки використанню рідних системні виклики для збору інформації статистики. У разі проблем (таких як відсутність вільного місця на жорсткому диску або аварійного завершення процесу сервісу), сервер може швидко повідомити адміністраторів серверу, який повідомив про помилку.

Агенти можуть виконувати пасивні і активні перевірки. У разі пасивної перевірки агент відповідає на запит даних. сервер (або проксі) запитує дані, наприклад, завантаження CPU, і агент повертає результат. Активні перевірки вимагають більш складної обробки. Агент спочатку отримує список елементів даних для незалежної обробки від сервера. Далі він буде періодично відправляти нові значення сервера.

За рахунок використання рідних системних викликів, агенти дозволяють ефективно відстежувати зміни метрик підконтрольної системи, що є необхідною умовою використання методу оцінки ступеня довіри до вузла.

Алгоритми міграції даних. Задача розфарбовування дуг графа належить до класу NP-складних [3]. Алгоритми точного її рішення вимагають повного перебору і мають експоненціальну обчислювальну складність, тобто не є поліноміальними. Однак, також існують апроксимаційні поліноміальні алгоритми розв'язання цієї задачі.

У статті [4] описаний поліноміальний апроксимаційний алгоритм розфарбовування ребер графу з обчислювальною складністю O(A*(V+δ)), де A – множина дуг графу; V - множина вершин; δ - деяка константа. Цей алгоритм заснований на принципі “розділяй і володарюй”. Загальне рішення для всього графа вибудовується з об'єднання рішень для множини розфарбованих підграфів. На початку роботи алгоритму максимальний розмір підграфів обмежений невеликим числом. Дуги обираються послідовно в довільному порядку і формують нові підграфи або входять до складу вже відомих. При певних умовах нова дуга може об'єднати два підграфи в один. У процесі роботи алгоритму максимальний розмір підграфів поступово збільшується. Після проходу по всіх дугах розфарбовані підграфи об'єднуються в єдиний граф.

Задача розфарбування дуг дводольного графа є окремим випадком задачі розфарбування дуг, але вона не належить до класу NP-повних. У статті [5] можна знайти опис оптимального алгоритму розфарбовування дуг з обчислювальною складністю O(A*logD), де A – множина дуг; D – максимальний ступінь вершин графу. В разі простого, дводольного графа можна визначити правила додавання дуг до тих чи інших підграфів і правила об'єднання підграфів, які дозволяють формувати гарантовано оптимальне розфарбування дуг на кожному кроці алгоритму.

У статті [6] приведено спосіб, який дозволяє зменшити час міграції даних з пристроїв зберігання, які додаються або від'єднуються від сховища під час масштабування, шляхом збільшення загального часу міграції даних. В цьому способі серед множини пристроїв зберігання, що входять до складу масштабованого розподіленого сховища, виділяється підмножина масштабуючих пристроїв зберігання S. До цієї підмножини відносяться пристрої, що на момент поточної процедури балансування додаються або звільняються зі сховища. Переміщення даних між двома масштабуючими пристроями зберігання під час процедури балансування даних не проводиться.

Під час масштабування необхідно виконати процедуру міграції на всіх масштабуючих пристроях S. Після цього їх можна звільнити та виконати міграцію на пристроях, що залишилися. Отже, процедуру міграції можна розділити на дві частини: масштабування та залишкову міграцію. Для вирішення подібних задач використовують розділення графу на підграфи. З огляду на це мультиграф вимог G (1) можна розділити на два підграфа: масштабуючий G_S і залишковий G_R.

Масштабуючим підграфом G_S є підграф мультиграфу вимог G, що складається з множини вершин S (масштабуючих пристроїв зберігання) і множини вершин S_{adj}, суміжних з S, а також всіх дуг, що з'єднують вершини S с S_{adj}:

$$\begin{gathered} Gs = (Vs, Es, P) \\ Vs \subseteq V, Vs = S \bigcup Sadj \\ Es \subseteq E \\ \upsilon \in Sadj \subseteq V \Leftrightarrow \exists \omega \in S, P(\upsilon, \omega) > 0 \\ (\upsilon, \omega) \in Es \Leftrightarrow \upsilon \in S, \omega \in Sadj, \\ P(\upsilon, \omega) > 0 \end{gathered} \tag{2}$$

Залишковим підграфом G_R є підграф графу вимог G, що складається з множини вершин і дуг графу G за винятком вершин S і дуг, що з'єднують вершини S і S_{adj}:

$$V_R \subseteq V, V_R = V, \quad S$$
$$E_R \subseteq E \tag{3}$$
$$(\upsilon, \omega) \in E_R \Leftrightarrow \upsilon, \omega \in V_R, P(\upsilon, \omega) > 0$$

Приклад поділу мультиграфу вимог на підграфи наведено на рисунку 2. Очевидно, що множина вершин S_{adj} належить як підграфу G_S так і підграфу G_R, згідно з їх визначеннями. Дуги, що з'єднують вершини S_{adj} між собою належіть тільки підграфу G_R.

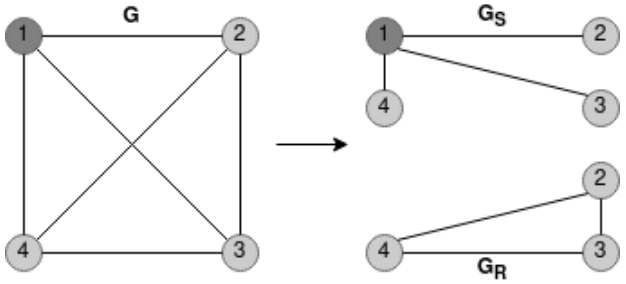

Рис. 2. Поділ графу вимог розподіленого масштабованого сховища даних G на два підграфа: G_S і G_R [6].

Виходячи з визначення масштабуючого підграфа (3) він складається з множини масштабуючих вершин S і множини суміжних з ними вершин S_{adj}. Вершини S не мають спільних дуг, так як це суперечить визначенню масштабуючих вершин, а вершини S_{adj} не мають спільних дуг, так як це суперечить визначенню масштабуючого підграфа (3). Отже, масштабуючий підграф G_S є дводольним графом з долями, що складаються з множини вершин S і S_{adj}.

Враховуючи, що підграф G_S дводольний, у статті [6] запропоновано наступний спосіб розв'язання задачі міграції та показано, що він має поліноміальну складність max{O(A*logD), O(A*(V+δ))}, де A – множина дуг, V – множина вершин, D – максимальна ступінь вершини, δ – деяка константа:

1. виділити підграф G_S з графу G на основі множини масштабуючих вершин S;
2. виділити підграф G_R з графу G на основі підграфу G_S;
3. використовувати алгоритм зі статті [5] для знаходження плану міграції для дводольного графу вимог G_S;
4. використовувати алгоритм зі статті [4] для знаходження плану міграції для графу вимог G_R;
5. отримати загальний план міграції для графу вимог G шляхом послідовного об'єднання планів міграції для підграфів G_S і G_R.

Модифікований метод оцінки ступеня довіри до вузлів

Базова роль системи моніторингу - це інформування про наближення стану компоненту до критичних значень. Візьмемо за основу наступний принцип взаємодії системи моніторингу з підконтрольною системою [7]:

$$S = \square\ O, P, L, M, f_p, f_u \square \tag{4}$$

де O – множина метрик підконтрольної системи; P – множина властивостей метрик підконтрольної системи; L – множина пов'язаних між собою метрик, що відображає залежність стабільного стану їх самих або їх властивостей один від одного; M – множина датчиків або способів аналізу системи моніторингу; f_p – функція, що відображає наявність певної метрики у компонента системи:

$$f_p = O \otimes P \rightarrow \{0,1\} \tag{5}$$

– функція, яка відображає можливість використання конкретного датчика або методу аналізу для контролю стану даної метрики:

$$f_u = M \otimes P \rightarrow \{0,1\} \tag{6}$$

Ступінь однозначності стану підконтрольної системи може бути представлена як

$$E = \frac{\sum_{i=1}^{c} w_{p_i} n_{p_i}}{\sum_{j=1}^{d} w_{p_j} n_{p_j}} \tag{7}$$

де p_i – метрики, підконтрольні системі моніторингу (тобто для яких існує $f_u(p_i, s) = 1$); p_j –

всі метрики підконтрольної системи (тобто для яких існує $f_p(o, p_i) = 1$); w – внесок метрики в функціонування підконтрольної системи; n_p – число компонентів, що мають метрику p в підконтрольній системі.

Додамо додатковий критерій при оцінці якості моніторингу - середню інформативність повідомлення про стан підконтрольної системи за певний період часу. Для вирішення цього завдання використаємо стандартні засоби системи моніторингу Zabbix.

Нижче представлені виділені класи по ступеню інформативності повідомлень (наведені приклади реальних повідомлень для кожного класу):

0. Повідомлення про відсутність компонента системи або не містить інформативної діагностичної інформації:

Switch 1 - WS-C2960S-48LPD-L - Power Supply 0: Device has been replaced (new serial number received)

1. Повідомлення пов'язані з неправильним налаштуванням компонентів системи:

Interface Fa0/4(ucsfi1-A.cr8.p16a.kiev.mgmt (mgmt_if)): In half-duplex mode

2. Повідомлення про загальну недоступність компонента:

Unavailable by ICMP ping

3. Повідомлення про загальну непрацездатність певної локалізованої складової частини компонента:

Interface Gi0/2(): Link down

4. Повідомлення містить інформацію тільки про факт невідповідності значення критерію:

Sw1, PS1 Normal, RPS NotExist: Power supply is in critical state

5. Повідомлення містить поточне значення параметра:

#1: High CPU utilization (over 90% for 5m)

Кожному повідомленню присвоюється певна інформаційна вага від 0 (для повідомлень категорії 0) до 1 (для повідомлень категорії 5) відповідно до її внеску в систему оцінювання стану системи.

Після розподілу ваги I_{aver} розраховується за формулою:

$$I_{Aver} = \frac{\sum_{i=1}^{k} I_{m_i}}{k} \tag{8}$$

де I_m – інформаційна вага повідомлення m; k – число опитувань стану властивості m, здійснених системою за заданий період часу.

Для відображення різниці в інформаційній вазі різних повідомлень і сенсорів (методів) їх оцінки, слід розширити область значення функції застосовності f_u:

$$f_u = M \otimes P \to \{0...1\} \tag{9}$$

де

$$f_u(s, p) = I_{Aver_{s,p}} \tag{10}$$

тобто значення функції для деякої пари властивості p і сенсора s визначається середньою інформативністю зміни метрики, зафіксованої способом аналізу або сенсором s в процесі моніторингу властивості p. Таким чином, в разі неможливості застосування способу аналізу або сенсора для контролю певної метрики, зібрані ним дані будуть належати до класу 0, і значення функції застосовності як і в початковому вигляді дорівнюватиме 0.

Таким чином:

$$E = \frac{\sum_{i=1}^{c} \Delta_{ep_i} w_{p_i} n_{p_i}}{\sum_{j=1}^{d} w_{p_j} n_{p_j}} \tag{11}$$

де f_{ep_i} - коефіцієнт інформативності, який визначається як значення функції застосовності методу аналізу або сенсору, що використовується системою для моніторингу властивості p_i.

Спосіб паралельного розрахунку плану міграції

Запропонований способі заснований на способі міграції даних, який представлений у статті [6]. В цьому способі використано заміну послідовного об'єднання двох планів міграції на їх паралельне об'єднання.

Під паралельним об'єднанням мається на увазі наступне: роздільне розфарбування підграфів G_S і G_R, як це робиться в первісному способі, та складання окремих планів міграції для цих підграфів, але при складанні загального плану міграції G для всіх пересилок на кожному кроці в плані міграції підграфа G_S

проводиться пошук таких пересилок даних в плані міграції підграфа G_R, щоб ці пересилки можна було об'єднати в один крок в загальному плані міграції мультиграфа G. Вважається, що пересилки можна об'єднати на одному кроці плану міграції, якщо на цьому кроці жоден з пристроїв зберігання не використовується більше одного разу як джерело або приймач пересилки даних.

Враховуючи це запропонований спосіб розв'язання задачі міграції даних виглядає так:

1. виділяється підграф G_S з мультиграфа G на основі множини пристроїв зберігання S, які позначені підсистемою моніторингу як потенційно ненадійні;

2. виділяється підграф G_R з мультиграфа G на основі підграфа G_S;

3. традиційним способом розраховується план міграції M_S для підграфа G_S, який містить m кроків, M_{Si} – крок плану міграції, i = 1..m;

4. традиційним способом розраховується план міграції M_R для підграфа G_R, який містить n кроків, M_{Rj} – крок плану міграції, j = 1..n;

5. шляхом покрокового виконання наступних операцій знаходиться загальний план міграції M для мультиграфа G:

5.1. з плану міграції M_S для підграфа G_S вибирається крок k з найменшою кількістю операцій переміщення даних;

5.2. з обраного кроку отримується множина пристроїв зберігання X_S, яки беруть участь у переміщенні даних на цьому кроці;

5.3. для кожного кроку M_{Rj}, j = 1..n з плану міграції M_R отримується множина пересилок даних T_{Rj} яки беруть участь у переміщенні даних на цьому кроці;

5.4. для кожної пересилки даних з T_{Rj} отримується множина пристроїв зберігання X_R, які беруть участь у цій пересилці;

5.5. якщо множини X_S та X_R не перетинаються, то пересилка даних додається до кроку k та видаляється з плану міграції M_R;

5.6. після того, як проглянуто усі кроки з плану міграції M_R, крок k додається до загального плану міграції M та видаляється з плану міграції M_S;

5.7. після того, як з плану міграції M_S видалені усі кроки та додані до плану міграції M, до M додаються також усі кроки плану міграції M_R, які залишились не використаними.

Тепер визначимо обчислювальну складність запропонованого алгоритму. Задача виділення підграфів з графа має лінійну обчислювальну складність O(V). Завдання паралельного об'єднання планів міграції має поліноміальну обчислювальну складність O(m*n), де m – кількість кроків у плані міграції для підграфа G_S, n – кількість кроків у плані міграції для підграфа G_R. Алгоритми зі статей [4] та [5] мають поліноміальну обчислювальну складність: O(A*(V+δ)) і O(A*logD), відповідно. Оскільки представлений алгоритм являє собою послідовне об'єднання кроків з першого по п'ятий, його обчислювальна складність дорівнює складності найбільш витратного кроку: max{O(V), O(A*logD), O(A*(V+δ)), O(m*n)} = max{O(A*logD), O(A*(V+δ))}. З цього випливає, що алгоритм має поліноміальну складність.

Оцінка ефективності запропонованого способу

Використовуючи традиційні способи міграції, відновлення коефіцієнту реплікації даних можна здійснити тільки після повного виконання процедури міграції даних. Це пов'язано з тим, що пристрої зберігання, які система моніторингу вважає потенційно ненадійними, можуть бути залучені до процедури міграції даних до самих останніх кроків. Запропонований у статті [6] спосіб виділяє в мультиграфі G масштабуючий підграф G_S, що містить всі масштабуючі пристрої, а в нашому випадку усі підозрілі пристрої. Це дозволяє швидше виконати процедуру міграції на цих пристроях. Зворотною стороною способу є необхідність виконання залишкової міграції в підграфі G_R після виконання міграції в G_S і послідовне об'єднання кроків міграції, яке збільшує загальний час міграції. У запропонованому способі послідовне об'єднання планів міграції у підграфах G_S та G_R замінюється на паралельне об'єднання отриманих планів міграції.

Для оцінки ефективності запропонованого способу було проведено його порівняння з традиційним способом міграції, якій оптимізований за загальним часом міграції на всіх пристроях зберігання, та зі способом, запропонованим у статті [6], якій також оптимізований за часом міграції на окремо виділених пристроях зберігання.

Під час проведення експериментів на вхід всіх способів подавалися однакові мультиграфи вимог G, які були отримані після розв'язання задачі розміщення з наступними вхідними параметрами:

- загальна кількість пристроїв зберігання - від 10 до 100 з кроком 10;
- кількість підозрілих пристроїв зберігання - 1 у всіх чергах експериментів.

Вихідними параметрами способів є:

- T(G) – загальний час міграції (у кроках) на всіх пристроях за традиційним способом;
- $T(G_S)$ – час міграції (у кроках) на потенційно ненадійних пристроях зберігання, розраховується як час міграції у підграфі G_S;
- $T(G_R)$ – час міграції (у кроках) на залишкових пристроях зберігання, розраховується як час міграції у підграфі G_R;
- $T(G_S) + T(G_R)$ — загальний час міграції (у кроках) на всіх пристроях за способом зі статті [7];
- $T_{PARALLEL}(G)$ – загальний час міграції (у кроках) на всіх пристроях за запропонованим способом.

Очевидно, що час міграції на масштабуючих пристроях для традиційного способу дорівнює загальному часу міграції T(G).

Виходячи з критеріїв задачі міграції даних в розподіленому сховищі, будемо використовувати наступні параметри оцінки:

1. P — час (у кроках), на який зменшується процес міграції з потенційно ненадійних пристроїв зберігання, де:

$$P = T\left(G\right) - T\left(G_S\right) \tag{12}$$

2. ΔP - це відносний виграш у часі міграції з потенційно ненадійних пристроїв зберігання, а отже й у часі відновлення коефіцієнту реплікації) при застосуванні запропонованого способу в порівнянні з традиційним способом, де:

$$\Delta P = \frac{P}{T\left(G\right)} = \frac{T\left(G\right) - T\left(G_S\right)}{T\left(G\right)} \tag{13}$$

3. $\Delta(T(G_S) + T(G_R))$ - це відносний виграш у загальному часі міграції при застосуванні запропонованого у статті [7] способу в порівнянні з традиційним способом, де:

$$\Delta L = \frac{T\left(G_S\right) + T\left(G_R\right)}{T\left(G\right)} - 1 \tag{14}$$

4. $\Delta T_{PARALLEL}(G)$ - це відносний виграш у загальному часі міграції при застосуванні запропонованого способу в порівнянні з традиційним способом, де:

$$\Delta L = \frac{T_{PARALLEL}\left(G\right)}{T\left(G\right)} - 1 \tag{15}$$

Було проведено 3 черги експериментів для різної відносної кількості елементів даних:

1. кількість елементів даних у 4 рази більша за кількість пристроїв зберігання;
2. кількість елементів даних у 8 разів більша за кількість пристроїв зберігання;
3. кількість елементів даних у 16 разів більша за кількість пристроїв зберігання.

В кожній черзі експериментів було проведено експерименти для різної кількості пристроїв зберігання в розподіленому сховищі: від 10 до 100 з кроком 10. Для кожного значення кількості пристроїв було проведено 100 експериментів. Остаточні значення вихідних параметрів отримувались усередненням. Результати усіх черг експериментів наведені у таблиці 1.

Таблиця 1

Результати експериментів

V	T(G)	$T(G_S)$	P	ΔP	$T(G_S)+T(G_R)$	$\Delta(T(G_S) + T(G_R))$	$T_{PARALLEL}(G)$	$\Delta T_{PARALLEL}(G)$
1 черга – кількість елементів даних – 4 * V								
10	14.00	12.26	1.74	12.4%	24.77	76.9%	22.74	62.4%
20	17.67	13.33	4.34	24.6%	30.13	70.5%	24.63	39.4%
30	19.22	14.18	5.04	26.2%	33.07	72.1%	24.42	27.1%
40	20.07	14.47	5.60	27.9%	34.11	70.0%	23.53	17.2%
50	20.67	14.79	5.88	28.4%	35.05	69.6%	23.39	13.2%
60	21.57	15.18	6.39	29.6%	36.26	68.1%	23.18	7.5%
70	21.62	15.47	6.15	28.4%	36.67	69.6%	23.01	6.4%
80	21.72	15.47	6.25	28.8%	36.94	70.1%	22.57	3.9%
90	22.11	15.47	6.64	30.0%	37.27	68.6%	22.51	1.8%
100	22.08	15.28	6.80	30.8%	37.03	67.7%	22.62	2.4%
2 черга – кількість елементів даних – 8 * V								
10	19.74	19.30	0.44	2.2%	37.12	88.0%	33.74	70.9%
20	28.86	24.45	4.41	15.3%	51.82	79.6%	45.72	58.4%
30	33.72	25.03	8.69	25.8%	57.53	70.6%	48.84	44.8%
40	35.98	26.54	9.44	26.2%	61.64	71.3%	48.62	35.1%

V	T(G)	$T(G_S)$	P	ΔP	$T(G_S)+T(G_R)$	$\Delta(T(G_S) + T(G_R))$	$T_{PARALLEL}(G)$	$\Delta T_{PARALLEL}(G)$
50	37.89	27.17	10.72	28.3%	64.33	69.8%	48.18	27.2%
60	38.89	27.76	11.13	28.6%	66.13	70.0%	47.29	21.6%
70	40.04	28.30	11.74	29.3%	68.08	70.0%	46.67	16.6%
80	40.75	28.97	11.78	28.9%	69.34	70.2%	46.61	14.4%
90	41.54	29.41	12.13	29.2%	70.40	69.5%	46.14	11.1%
100	41.99	29.56	12.43	29.6%	71.11	69.3%	45.60	8.6%
3 черга – кількість елементів даних – 16 * V								
10	23.87	23.75	0.12	0.5%	46.20	93.5%	40.59	70.0%
20	42.04	40.47	1.57	3.7%	80.18	90.7%	70.11	66.8%
30	52.99	47.29	5.70	10.8%	98.54	86.0%	84.52	59.5%
40	59.92	48.30	11.62	19.4%	106.93	78.5%	90.89	51.7%
50	64.95	48.49	16.46	25.3%	112.36	73.0%	94.37	45.3%
60	68.37	49.81	18.56	27.1%	117.42	71.7%	95.34	39.4%
70	71.52	51.11	20.41	28.5%	121.28	69.6%	95.52	33.6%
80	73.61	52.55	21.06	28.6%	125.27	70.2%	95.60	29.9%
90	75.16	53.82	21.34	28.4%	128.24	70.6%	94.20	25.3%
100	76.51	55.07	21.44	28.0%	130.90	71.1%	93.94	22.8%

На рисунках 3 та 4 зображені графіки відношення значень ΔP та $\Delta T_{PARALLEL}(G)$ від кількості пристроїв зберігання даних V для усіх черг експериментів.

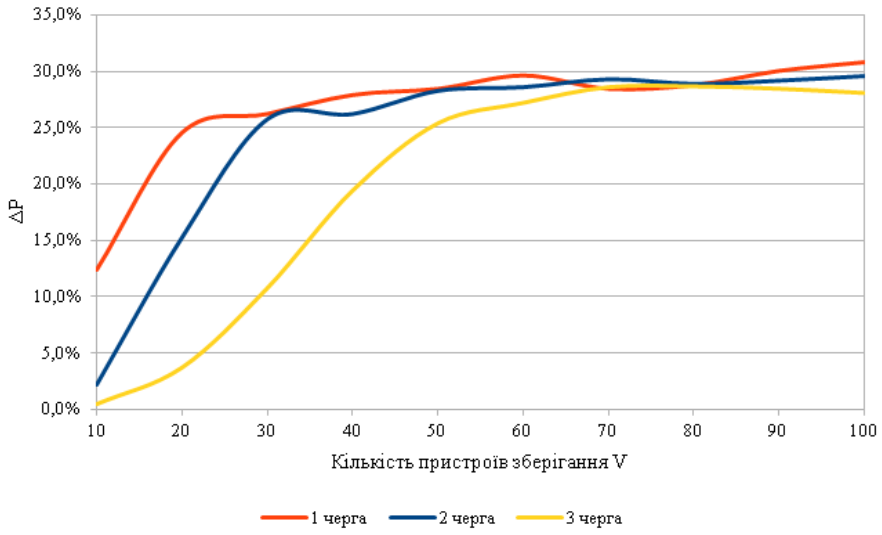

Рис. 3. Графік відношення значення ΔP від кількості пристроїв зберігання даних V для усіх черг експериментів.

Як видно з таблиці та графіків, запропонований спосіб міграції даних дозволяє зменшити час міграції з пристроїв, які помічені підсистемою моніторингу як потенційно ненадійні, при будь-якій кількості пристроїв зберігання та при будь-якій кількості елементів даних. Це своєю чергою зменшує час відновлення коефіцієнту реплікації у розподілених сховищах будь-якого розміру. Зменшення часу відновлення коефіцієнту реплікації досягає 30%.

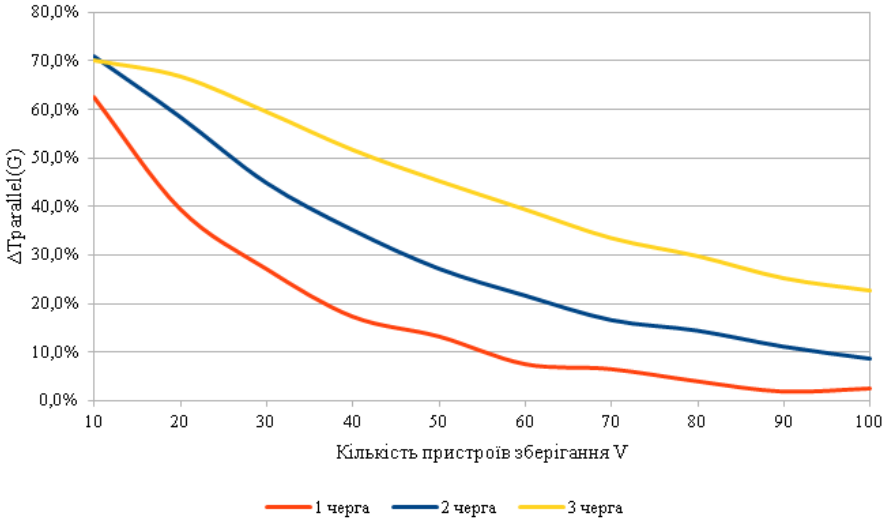

Рис. 4. Графік відношення значення $\Delta T_{PARALLEL}(G)$ від кількості пристроїв зберігання даних V для усіх черг експериментів.

Зменшення часу відновлення коефіцієнту реплікації відбувається коштом збільшення загального часу міграції даних, що досягає 70% при малій кількості пристроїв зберігання та поступово зменшується з ростом кількості пристроїв.

Висновки

В роботі розглянуто проблеми розв'язання задач міграції даних та відновлення коефіцієнту реплікації в розподілених масштабованих сховищах. Було запропоновано спосіб складання плану міграції даних з поліноміальною обчислювальною складністю. Показано, що запропонований алгоритм є оптимальним за часом міграції на пристроях, які позначені підсистемою моніторингу як потенційно ненадійні, а отже й за часом відновлення коефіцієнту реплікації у розподіленому сховищі. Експериментально показано, що запропонований спосіб однаково ефективний для розподілених сховищ з різною кількістю пристроїв зберігання та різною кількістю елементів даних, та здатний зменшувати час відновлення коефіцієнту реплікації у розподіленому сховищі до 30% коштом збільшення загального часу міграції даних на 20%.

Література

1. Hall J., Hartline J., Karlin A., Saia J., Wilkes J. On Algorithms for Efficient Data Migration // ACM Symposium on Discrete Algorithms. – 2001. – P. 620–629.

2. Vladishev A. What's new in Zabbix 5.2 [Електронний ресурс] / A. Vladishev // Zabbix blog. – 2020. – Режим доступу до ресурсу: https://blog.zabbix.com/whats-new-in-zabbix-5-2/12550/.

3. Holyer I. The NP-completeness of Edge-Coloring // SIAM J. Comp. – 1982. No11. – P. 117–129.

4. Hochbau D. S., Nishizeki T., Shmoys D. B. A better than "Best Possible" algorithm to edge color multigraphs // J. off Algorithms. – 1986. – No7. – P. 79–104.

5. Cole R., Ost K., Schirra S. Edge-coloring bipartite multigraphs in O(E log D) time // Combinatorica. – 2001. – №21. – P. 5–12.

6. Петров Д. Л. Оптимальний алгоритм міграції даних у масштабованих хмарних сховищах // Управління великими системами. – 2010. – No30. – С. 180–197.

7. Dmitrienko P. Methods of evaluating the effectiveness of systems for computing resources monitoring // Computer Research and Modeling – No. 3(4). – 2012 –pp. 661-668.

References

1. Hall J., Hartline J., Karlin A., Saia J., Wilkes J. On Algorithms for Efficient Data Migration // ACM Symposium on Discrete Algorithms. – 2001. – P. 620–629.

2. Vladishev A. What's new in Zabbix 5.2 [Електронний ресурс] / A. Vladishev // Zabbix blog. – 2020. – Режим доступу до ресурсу: https://blog.zabbix.com/whats-new-in-zabbix-5-2/12550/.

3. Holyer I. The NP-completeness of Edge-Coloring // SIAM J. Comp. – 1982. No11. – P. 117–129.

4. Hochbau D. S., Nishizeki T., Shmoys D. B. A better than "Best Possible" algorithm to edge color multigraphs // J. off Algorithms. – 1986. – No7. – P. 79–104.

5. Cole R., Ost K., Schirra S. Edge-coloring bipartite multigraphs in O(E log D) time // Combinatorica. – 2001. – №21. – P. 5–12.

6. Petrov D. Optymaljnyj alghorytm mighraciji danykh u masshtabovanykh khmarnykh skhovyshhakh / D. Petrov // Upravlinnja velykymy systemamy. – 2010. – No 30. – S. 180–197.

7. Dmitrienko P. Methods of evaluating the effectiveness of systems for computing resources monitoring // Computer Research and Modeling – No. 3(4). – 2012 –pp. 661-668.

Надійшла / Paper received : 18.11.2020 р. Надрукована/Printed :04.01.2021 р.