Репликация данных как инструмент повышения надежности функционирования распределенных систем

С.К. Сомов

Институт проблем управления им. В.А. Трапезникова РАН, г. Москва, Россия

Аннотация. Рассмотрено решение задачи повышения уровня сохранности данных в распределенных системах методами репликации массивов данных. Приведены примеры методов статической и динамической репликации. Дана постановка и алгоритм решения задачи оптимального распределения реплик по узлам компьютерной сети. Представлен пример работы предложенного алгоритма и выполнен анализ результатов его работы.

Ключевые слова: распределенные системы данных, репликация данных.

DOI 10.14357/20718632180307

Введение

Распределенные системы обработки данных (РСОД) в течение нескольких последних десятилетий получили чрезвычайно широкое распространение в разнообразных отраслях человеческой деятельности. РСОД это сложные технические системы, в рамках которых между собой взаимодействует огромное количество различных компонент, каждая из которых может подвергнуться воздействию различных негативных факторов. В результате некоторые элементы системы могут выйти из строя или снизить свои эксплуатационные характеристики. Потеря работоспособности отдельными компонентами системы может привести к сбоям и простоям в работе всей системы разной степени серьезности.

Негативные факторы, которые могут спровоцировать возникновение инцидентов в работе РСОД, имеют разнообразную природу. Например, факторы, приводящие к возникновению инцидентов с данными, можно классифицировать следующим образом [1,2]:

- Некорректная работа программного обеспечения, содержащего ошибки, приводящая к искажению входных и выходных данных, к потере информации.
- Отказы аппаратных средств компьютерной сети, например, отказы дисковых контроллеров, отказы кэш-памяти, неисправности электроники в дисковых устройствах.
- Сбои в работе оборудования распределенной системы: перебои в системе электропитания, сбои в каналах связи, сбои маршрутизаторов.
- Сбой оборудования в системах хранения или архивации, приводящие к искажению или потере информации.
- Заражение компьютерными вирусами.
- Несанкционированный доступ в систему.
- Ошибки в работе обслуживающего персонала или пользователей системы.

В результате компании несут убытки и потери, которые могут иметь разный характер и последствия для компаний. Например, это могут быть репутационные, имиджевые и финансовые потери, а также прямые и косвенные убытки компаний [3]. Можно сделать вывод о том, что убытки и потери компаний, в том числе и связанные с серьезными инцидентами с сохранностью данных, могут быть критичны для существования самих компаний. Поэтому обеспечение высокого уровня сохранности используемых в РСОД данных является одной из важнейших задач проектирования РСОД.

1. Метод репликации данных

Одним из способов повышения надежности работы распределенных систем является обеспечение высокого уровня сохранности используемой ими информации [4]. Эффективным методом решения этой задачи является создание и размещение в узлах сети реплик (идентичных копий) используемых в РСОД массивов данных (таблицы и фрагменты таблиц баз данных, файлы данных и их фрагменты) [5, 6]. Распределение реплик по узлам сети выполняется с целью максимального их приближения к потребителям данных (пользователи и/или прикладные приложения РСОД). С помощью такого размещения реплик решается сразу несколько важных задач проектирования РСОД: уменьшается время доступа к репликам; повышается надежность работы РСОД, поскольку при отказе узла с репликой запросы пользователей перенаправляются в другой ближайший работоспособный узел; упрощается решение задачи масштабирования системы.

Идентичность данных в репликах одного массива обеспечивается механизмом репликации. Данный механизм заключается в том, что изменения, сделанные в одном экземпляре реплики, распространяются на все другие узлы с аналогичными репликами. Репликация выполняется в синхронном или в асинхронном режиме. Выбор режима репликации зависит от расстояния между узлами с репликами и от требований ко времени распространения изменений.

Для реализации механизма репликации используются три основных метода [6]:

1. распространение по узлам с репликами сообщений об изменении данных;
2. передача самих измененных данных;
3. распространение операции обновления по узлам с репликами.

Использование механизма репликации в РСОД требует дополнительных затрат: затраты на хранение реплик в узлах сети, на поддержку идентичности данных в репликах, а также требуются ресурсы на обработку дополнительного трафика сообщений, возникающего при репликации данных. По этой причине проектировщики и администраторы РСОД вынуждены искать компромисс между производительностью и надежностью работы системы и увеличением затрат на ее функционирование с другой стороны.

2. Поиск оптимальных вариантов размещения реплик

Поиск оптимальных вариантов размещения реплик ведется различными методами [7-12]. Но в итоге все они сводятся к оптимизационной задаче, в которой необходимо из множества всех N узлов сети выбрать такое подмножество узлов M $(M<N)$, которое обеспечит наилучшее значение используемого критерия оптимальности. Такие задачи обладают большой вычислительной сложностью, поэтому для их решения используются различные специальные методы, например эвристики, которые позволяют уменьшить вычислительную сложность таких задач [7]. Выделяют две основные группы методов/алгоритмов решения задач подобного типа: статические методы и методы динамического размещения реплик.

Первую группу составляют алгоритмы поиска размещения реплик [7, 8], которые основываются на предположении о статичной работе РСОД, когда характеристики системы и ее топология не изменяются. Трафик запросов пользователей и его распределение по узлам системы также считаются неизменными. Недостатком таких алгоритмов является то, что на практике трафик и место генерации запросов пользователей и их интенсивность со временем изменяются. Параметры самой системы со временем также могут меняться. Вследствие этих причин эксплуатационные характеристики РСОД могут значительно ухудшаться.

Вторую группу методов составляют алгоритмы динамического размещения реплик

[7, 9], которые позволяют подстраивать параметры РСОД к изменившимся условиям функционирования. Эти методы используют различные способы управления размещением реплик по узлам системы и их количеством.

Рассмотрим в качестве примера статического размещения реплик алгоритм, определяющий сегменты сети (кластеры) для размещения в них реплик [10]. Под кластером в рассматриваемом алгоритме понимается множество узлов сети, которые генерируют запросы к одинаковым данным, и расстояние между которыми мало. Алгоритм решает задачу размещения реплик в два этапа. На первом этапе определяются кластеры с наибольшим количеством входящих в них узлов сети. На втором этапе для каждого из этих кластеров выбирается узел – сервер реплик, который будет хранить реплики и обрабатывать поступающие запросы к данным.

Предполагается, что узлы сети находятся в m–мерном геометрическом пространстве, которое разбивается на ячейки одинакового размера. Из этих ячеек выбирается M ячеек, которые содержат наибольшее количество размещенных в них узлов.

Заданный в алгоритме размер ячеек, на которые разбивается m–мерное пространство, оказывает сильное влияние на результат выбора ячеек, предназначенных для размещения реплик. При очень малом размере ячеек узлы большого кластера попадут сразу в несколько ячеек (Рис. 1, А). При этом в сети будет размещено слишком много реплик. Если размер ячеек очень большой, то в одну ячейку может поместиться одновременно несколько небольших кластеров. Так как в одной ячейке размещается одна реплика, то в этом случае реплик в сети будет недостаточно для обработки запросов с требуемой производительностью. При идеально подобранном размере ячеек в каждый кластер будет размещен в своей ячейке (Рис. 1, В).

Затем для каждой из M ячеек определяется один узел – сервер реплик. Описанный алгоритм имеет вычислительную сложность: $O(N \times max\{log(N), M\})$.

Рассмотрим механизм использования репликации службами web-хостинга в качестве примера метода динамической репликации [6,11]. Службы предоставляют пользователям доступ

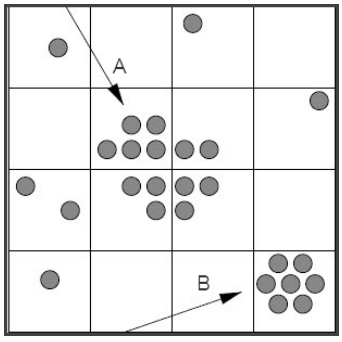

Рис. 1. Примеры распределения кластеров по ячейкам

к файлам данных, размещенным на web-серверах сети Интернет.

Множество web-серверов как правило постоянно. Но в некоторые периоды времени для ускорения доступа пользователей к файлам службы хостинга могут динамически копировать один или несколько таких файлов на серверы, которые находятся наиболее близко к пользователям, которым нужен быстрый доступ к этим файлам. В результате такого динамического перераспределения реплик увеличивается производительность самой системы и сокращается время ее отклика на запросы пользователей.

Использование динамической репликации требует ответа на три ключевых вопроса:

1. какие узлы будут хранить реплики и обрабатывать поступающие к ним запросы,
2. когда необходимо дополнительные реплики,
3. в какие моменты времени реплики можно удалять за ненадобностью.

Ответ на первый вопрос можно получить, используя один из методов статического размещения реплик [7, 12].

Ответы на второй и третий вопрос можно получить с помощью алгоритма динамической репликации файлов web-хостинга, предложенный в [6]. Алгоритм решает две основных задачи: уменьшение нагрузки на сервера методом динамической репликации, и репликация файлов данных на сервера, расположенные наиболее близко к пользователям этих файлов.

Каждый сервер с репликами файлов и обрабатывающий запросы пользователей контролирует два параметра. Сервер контролирует узлы, из которых приходят запросы, и он подсчитывает интенсивность этих запросов. Допустим,

на сервере S расположен файл F. Запросы к файлу F, поступающие от сервера P, будут подсчитываться в переменной - счетчике обращений counters(P,F). Кроме того, для каждого сервера S и каждого файла F задаются два лимита: порог репликации rep(S,F) и порог удаления реплики del(S,F).

Предположим, что значение счетчика counters(P,F) обращений к файлу F на сервере S, превысило порог репликации rep(S,F). Это означает, что интенсивность запросов к файлу F стало слишком большой. Следовательно, файл F необходимо реплицировать на другой сервер. В силу разных причин данная репликация (перенос файла на другой сервер) может быть неуспешной по разным причинам. Например, на другом сервере недостаточно памяти для размещения файла или сервер уже сильно нагружен. Тогда используется другой сервер для переноса на него файла F.

Если число запросов к реплике файла F на сервере S стало меньше порога удаления, то эта реплика удаляется с сервера S. Реплика не удаляется с сервера, если это последняя реплика файла F в системе.

3. Пример формулировки задачи оптимального размещения реплик

Задачу поиска оптимального размещения реплик по узлам распределенной системы сформулируем в виде обобщенной минисуммной задачи о p-медиане [14-16]. Здесь p – это количество узлов системы, в которых необходимо разместить реплики. В качестве критерия оптимизации используем минимум затрат на функционирование распределенной системы.

Допустим, что РСОД функционирует на основе компьютерной сети, состоящей из N узлов. Топология сети задана и представлена в виде неориентированного графа $G = (X, \Gamma)$. Часть узлов играют роль пользовательских серверов. Остальные узлы исполняют роль серверов реплик. В них размещены реплики массивов данных, и они обрабатывают запросы к репликам, поступающие от пользовательских серверов. Каждый массив данных может иметь более одной реплики, которые размещены в разных узлах сети. Требуется найти подмножество $\tilde{X}_p$ из p узлов множества X всех узлов сети такое, что размещение в нем p реплик массива данных обеспечит минимальное значение стоимости функционирования РСОД.

Пользовательские сервера генерируют информационные запросы и запросы на модификацию данных. Информационные запросы пересылаются в ближайший сервер реплик с необходимыми данными. Запросы на модификацию данных посылается во все узлы с репликами, данные которых должны быть изменены.

Пусть $\Lambda_n^e = \|\lambda_n^e\|$ это вектор размерности N, а λ_n^e – интенсивность информационных запросов, создаваемых в узле x_n. Аналогично $\Lambda_n^u = \|\lambda_n^u\|$ - вектор размерности N, где λ_n^u интенсивность запросов на модификацию, генерируемых в n-ом узле сети. Предполагаем, что все запросы обрабатываются, так как система работает в установившемся режиме.

Введем обозначения:

1) $dis(x_n, x_j)$ – длина наикратчайшего пути между узлом x_n и узлом x_j.

2) s - стоимость передачи единицы данных по пути единичной длины.

3) d^e, d^u–средний объем данных, передаваемый по сети между пользовательским сервером и сервером реплики при обработке информационного запроса и запроса на модификацию данных. Будем считать их одинаковыми.

4) $cost\_S(x_n)$ – стоимость хранения в узле x_n одной реплики за единицу времени.

5) $cost\_E(x_n)$, $cost\_U(x_n)$ - стоимость обработки в узле x_n информационного запроса и запроса на модификацию данных.

6) $V = \{v_1, \dots, v_n, \dots, v_N\}$ – вектор с «весами» узлов компьютерной сети, в котором элемент v_n равен среднему объему данных, которые узел n передает и получает при обработке всех запросов, сгенерированных в этом узле за единицу времени, т.е. $v_n = (\lambda_n^e d^e + p\lambda_n^u d^u)$.

Для поиска оптимального размещения p реплик будем использовать обобщенную минисуммную задачу о p-медиане. Будем использовать минимум затрат на функционирование системы в качестве критерия оптимизации этой задачи [2, 3].

Для узла x_i определим величину $\sigma(x_i)$:

$$\sigma(x_i) = \sum_{n=1}^{N} [dis(x_i, x_n) s v_n] \qquad (1)$$

В формуле (1) $\sigma(x_i)$ это передаточное число, которое равно сумме затрат на обмен данными между узлом x_i и узлами множества X, который производится при обработке запросов, возникающих в системе за единицу времени. Вершина $\bar{x}$ называется медианой графа G, если для нее достигается минимум передаточного числа:

$$\sigma(\bar{x}) = \min_{x_i \in X} \sigma(x_i) = \min_{x_i \in X} \left\{ \sum_{n=1}^{N} [dis(x_i, x_n) s v_n] \right\}$$

Обозначим через X_p подмножество вершин из X, в которых размещено p реплик. Минимальное расстояние от вершины x_n множества X до одной из вершин множества X_p обозначим как $d(X_p, x_n)$. Оно равно:

$$d(X_p, x_n) = \min_{x_j \in X_p} dis(x_n, x_j) \qquad (2)$$

Если для вершины x_j из множества X_p достигается минимум выражения (2), то запросы, возникающие в узле x_n, будут пересылаться для обработки в узел x_j. Обозначим этот факт как $x_n \to x_j$.

По аналогии с формулой (1) определим $\sigma(X_p)$ как передаточное число для подмножества вершин X_p [12]:

$$\sigma(X_p) = \sum_{n=1}^{N} v_n s d(X_p, x_n) = \sum_{n=1}^{N} v_n s \min_{x_j \in X_p} dis(x_n, x_j)$$

Множество $\tilde{X}_p$ называется p-медианой графа G, если для этого множества достигается минимум значения передаточного числа: $\sigma(\tilde{X}_p) = \min_{X_p \subseteq X} [\sigma(X_p)]$.

Обозначим через $COST(X_p)$ стоимость функционирования РСОД за единицу времени в зависимости от распределения реплик X_p. Эта стоимость будет равна:

$$COST(X_p) = \sum_{j=\frac{1}{x_j} \in X_p}^{p} cost_{S(x_j)} +$$

$$+ \sum_{j=1/x_j \in X_p}^{p} cost\_E(x_j) \left(\sum_{n=1/x_n \to x_j}^{N} \lambda_n^e \right) +$$

$$+ \sum_{n=1}^{N} \lambda_n^u \left(\sum_{j=1/x_j \in X_p}^{p} cost\_U(x_j) \right)$$

Стоимость функционирования РСОД включает в себя три компоненты: затраты системы на хранение реплик, затраты на обработку информационных запросов в узлах с репликами и затраты на обработку запросов на модификацию данных во всех узлах с репликами.

Задача оптимального размещения p реплик массива данных по узлам РСОД будет иметь следующую формулировку.

Для графа $G = (X, \Gamma)$ необходимо найти подмножество $\tilde{X}_p$ из p вершин этого графа, для которого достигается минимум функционала: $F_p(\tilde{X}_p) = COST(\tilde{X}_p) + \sigma(\tilde{X}_p)$.

Сформулированная задача может иметь различные ограничения. Например, ограничение на максимальное число используемых в РСОД реплик: $|\tilde{X}_p| \le \bar{P}$. Или ограничение на объем информации, которой узел x_j с репликой обменивается с узлами, из которых он получает запросы: $\sum_{n=1/x_n \to x_j}^{N} (\lambda_n^e d^e + p \lambda_n^u d^u) \le \bar{D}$.

Сформулированная задача об оптимальном размещении реплик принадлежит классу задач о нахождении p-медианы графа. Задачи данного класса являются *NP* трудными. Для их решения разработаны различные методы, которые позволяют получить решение, близкое к оптимальному, за разумное время. Краткое описание методов можно найти в [13-15].

4. Алгоритм решения задачи размещения реплик

Для решения сформулированной выше задачи предлагается использовать эффективный эвристический алгоритм, позволяющий за приемлемое время и с достаточной точностью найти размещение реплик, близкое к оптимальному. Алгоритм является модификацией эвристического метода поиска *p*-медиан графа, описанного в работе [17], который обладает быстродействием, превышающим в 1,2 – 1,4 раза быстродействие ряда известных алгоритмов, обзоры которых даны в [18, 19]. Краткое описание алгоритма:

1. Задается искомое количество p медиан графа ($p < J$, $J = |X|$). Случайным образом из множества X вершин графа выбирается p вершин, которые составят множество X_p. Вершины из X, не включенные в X_p, отмечаются как «не протестирована».

2. Случайным образом из множества $\{X \backslash X_p\}$ выбирается «не протестированная» вершина x_j. Если таких вершин нет, то переход к шагу 6.

3. В цикле по каждой вершине x_i из X_p вычисляется величина Δ_{ij}, на которую изменится функционал F_p, если вершину x_i из X_p заменить на вершину x_j, отобранную на шаге 2):

$$\Delta_{ij} = F_p(X_p) - F_p(X_p{}^i),$$

где $X_p{}^i = (X_p \cup \{x_j\}) \backslash \{x_i\}$

В этом цикле запоминаем индекс i^* вершины из X_p, при замене которой на x_j достигается максимальное изменение функционала, т.е. $\Delta_{i^*j} = \max_{x_i \in Xp} \Delta_{ij}$.

4. Если $\Delta_{i^*j} \leq 0$, то замена на вершину x_j любой из вершин множества X_p не производится, так как замена не улучшает значение функционала задачи. Маркируем вершину x_j как «протестирована» и возвращаемся к пункту 2.

5. При $\Delta_{i^*j} > 0$ замена вершины x_{i^*} из X_p на вершину x_j улучшает значение функционала. Производим замену вершин и маркируем обе вершины x_j и x_{i^*} как «протестированные». В итоге получено новое множество $X_p = (X_p \cup \{x_j\}) \backslash \{x_i\}$. Возвращаемся к пункту 2.

6. Завершение работы алгоритма. В итоге получено множество X_p, близкое к оптимальному p-медианное множество узлов РСОД для размещения реплик.

Описанный в данном разделе алгоритм реализован в среде MS Visual Studio в виде модуля на языке C++. Модуль использовался при решении задачи оптимального размещения реплик на примере распределенной системы. Описание распределенной системы и результаты работы алгоритма представлены ниже.

5. Пример решения задачи

Имеется распределенная система, функционирующая на базе компьютерной сети из 20-ти узлов. Задано количество медиан p = 6. Топология сети представлена на Рис. 2. Узлы сети пронумерованы, на дугах проставлены их длины.

Характеристики распределенной системы и компьютерной сети следующие:

- s - стоимость передачи единицы данных по пути единичной длины равна 2.
- Интенсивности λ_j^e и λ_j^u запросов, возникающих в узлах сети, приведены в Табл. 1.
- Объем данных d^e равен 150 байт, а d^u равно 50 байт.

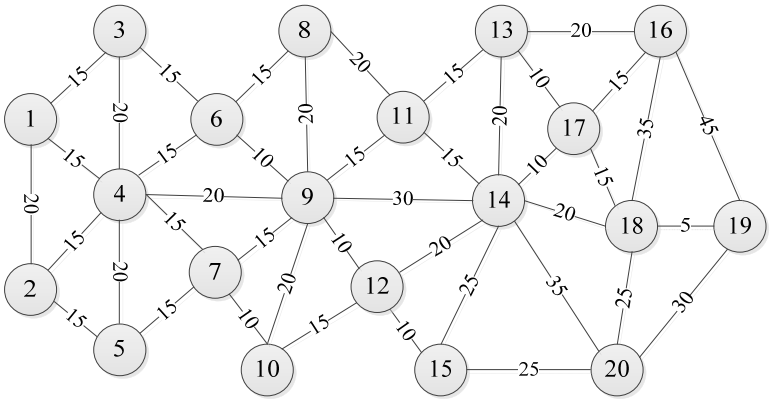

Рис. 2. Топология компьютерной сети

Табл. 1. Матрица интенсивностей запросов в узлах сети

Узел	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
λ_j^e	140	135	140	25	115	118	8	25	20	7	30	25	145	20	25	130	25	40	145	135
λ_j^u	5	2	8	6	7	2	2	8	5	2	9	8	8	4	4	2	3	7	8	7

- В Табл. 2 перечислены «Веса» v_j узлов сети.

- Значения стоимости хранения реплик - $cst\_s(x_j)$, обработки информационных запросов - $cst\_e(x_j)$ и запросов на модификацию данных - $cst\_u(x_j)$ заданы в Табл. 3.

- Длины кратчайших путей рассчитаны с помощью алгоритма Флойда [13] и показаны в Табл. 4.

В результате работы алгоритма при заданных параметрах найдено распределение реплик по узлам сети (1, 3, 5, 11, 13, 18) со значением функционала задачи, равным 762 553 единиц. Полученные значения компонент функционала задачи приведены в Табл. 5.

Из данных, приведенных в Табл. 5 видно, что при заданных параметрах задачи наибольший вклад в величину функционала вносят две компоненты. Первая компонента это передаточное число, т.е. величина затрат на обмен данными по каналам связи сети при обработке всех запросов. Вторая по величине компонента - стоимость обработки запросов на модификацию данных.

Табл. 2. Веса узлов сети

Узел	1	2	3	4	5	6	7	8	9	10
v_j	2 525	2 195	2 780	885	2 320	1 940	290	1 055	725	275
Узел	11	12	13	14	15	16	17	18	19	20
v_j	1 215	1 055	2 855	640	715	2 120	630	1 195	2 855	2 620

Табл. 3. Стоимость хранения реплик и обработки запросов

Узел	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
cost_S	340	440	550	220	440	330	440	250	330	550	330	330	440	330	550	550	550	540	550	440
cost_E	15	15	13	11	12	14	15	13	12	14	10	12	15	12	13	10	11	12	10	14
cost_U	400	350	370	314	410	382	340	410	365	371	412	412	350	374	411	410	412	410	390	400

Табл. 4. Матрица длин кратчайших путей

Узел	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0	20	15	15	35	30	30	45	35	40	50	45	65	65	55	85	75	85	90	80
2	20	0	35	15	15	30	30	45	35	40	50	45	65	65	55	85	75	85	90	80
3	15	35	0	20	40	15	35	30	25	45	40	35	55	55	45	75	65	75	80	70
4	15	15	20	0	20	15	15	30	20	25	35	30	50	50	40	70	60	70	75	65
5	35	15	40	20	0	35	15	50	30	25	45	40	60	60	50	80	70	80	85	75
6	30	30	15	15	35	0	25	15	10	30	25	20	40	40	30	60	50	60	65	55
7	30	30	35	15	15	25	0	35	15	10	30	25	45	45	35	65	55	65	70	60
8	45	45	30	30	50	15	35	0	20	40	20	30	35	35	40	55	45	55	60	65
9	35	35	25	20	30	10	15	20	0	20	15	10	30	30	20	50	40	50	55	45
10	40	40	45	25	25	30	10	40	20	0	35	15	50	35	25	70	45	55	60	50
11	50	50	40	35	45	25	30	20	15	35	0	25	15	15	35	35	25	35	40	50
12	45	45	35	30	40	20	25	30	10	15	25	0	40	20	10	60	30	40	45	35
13	65	65	55	50	60	40	45	35	30	50	15	40	0	20	45	20	10	25	30	50
14	65	65	55	50	60	40	45	35	30	35	15	20	20	0	25	40	10	20	25	35
15	55	55	45	40	50	30	35	40	20	25	35	10	45	25	0	65	35	45	50	25
16	85	85	75	70	80	60	65	55	50	60	35	45	20	25	50	0	15	30	35	55
17	75	75	65	60	70	50	55	45	40	45	25	30	10	10	35	30	0	15	20	40
18	85	85	75	70	80	60	65	55	50	55	35	40	25	20	45	35	15	0	5	25
19	90	90	80	75	85	65	70	60	55	60	40	45	30	25	50	40	20	5	0	30
20	80	80	70	65	75	55	60	65	45	50	50	35	50	35	25	60	40	25	30	0

Табл. 5. Компоненты функционала задачи

Распределение реплик по узлам сети	Функционал задачи	Передаточное число	СТОИМОСТЬ:		
			хранения реплик	обработки информационных запросов	обработки запросов на модификацию данных
1, 3, 5, 11, 13, 18	762 553	489 450	2 640	18 799	251 664

Рис. 3. Зависимость значения функционала задачи от количества медиан

Ниже представлены результаты эксперимента, в котором изучалось то, как изменяется значение функционала задачи и его компонент при изменении числа медиан. Задача решалась при значениях медиан в диапазоне от 2 до 17.

Результаты эксперимента представлены на графике Рис. 3 и в Табл. 6.

При анализе Табл. 6 и Рис. 3 можно сделать следующие выводы. При небольшом количестве медиан (например, 2 и 3) функционал задачи имеет максимальные значения при больших значениях передаточного числа. Причина заключается в том, что при небольшом количестве узлов (медиан), в которых размещены реплики, каждый информационный запрос и ответ на него должны преодолеть большое расстояние по каналам сети до узла с репликой и обратно к узлу-источнику запроса. Это приводит к большим затратам на использование каналов связи.

При увеличении количества медиан значение функционала постепенно уменьшается и наименьшие значения он принимает при количестве медиан 9 и 10. Это достигается за счет приближения узлов с репликами к узлам – источникам запросов, что приводит к сокращению затрат на использование каналов связи. При этом происходит увеличение затрат на обработку запросов на модификацию данных, так как увеличивается количество узлов с репликами, информацию которых необходимо поддерживать в идентичном и актуальном состоянии. Для этого каждый запрос на модификацию данных маршрутизируется для обработки во всех узлах с репликами. Но это увеличение компенсируется тем, что при заданных параметрах задачи затраты на использование каналов связи уменьшаются быстрее, чем растет стоимость обработки запросов на модификацию данных.

Заключение

В статье представлен эффективный метод повышения сохранности информации в распределенных системах, основанный на размеще-

Табл. 6. Зависимость функционала и его компонент от количества медиан

Количество медиан	Значение функционала	Передаточное число (использования каналов связи)	Стоимость:		
			хранения реплик	обработки информационных запросов	обработки запросов на модификацию данных
2	1 018 841	927 400	770	15 343	75 328
3	918791	773 100	1 210	18 756	125 725
4	766 956	592 050	1 650	17 678	155 578
5	798 539	573 450	2 020	19 769	203 300
6	762 553	489 450	2 640	18 799	251 664
7	681 618	369 550	3 090	19 757	289 221
8	699 346	353 350	3 230	19 198	323 568
9	626 430	234 600	3 970	19 031	368 829
10	631 620	193 000	4 320	18 712	415 588
11	650 984	179 300	4 540	18 921	448 223
12	659 218	139 050	4 990	18 591	496 587
13	663 804	97 100	5 760	18 882	542 062
14	678 959	74 250	5 980	18 682	580 047
15	710 635	70 500	6 160	18 832	615 143
16	730 807	50 100	6 520	18 812	655 375
17	760 383	39 200	7 070	18 827	695 286

нии в узлах системы реплик массивов данных. Оптимальное размещение реплик в узлах распределенной сети позволяет также увеличить производительность и надежность работы системы. Дано описание эвристического алгоритма, который позволяет с достаточной точностью и за приемлемое время находить решение задачи оптимального размещения реплик. Приведен пример работы алгоритма для распределенной системы, состоящей из 20 узлов. Выполнен анализ результатов работы алгоритма. Оптимизация распределения реплик с помощью предложенного алгоритма позволяет на 10-12% снизить затраты на функционирование распределенной системы.

Литература

1. Восстановление данных. URL: http://www.datarecovery.ru/datarecovery.htm (дата обращения 22.01.2018).
2. Фролов А., Фролов Г. Сохранность и восстановление компьютерных данных: теория и практика//Byte. Россия - 2001, - №1(30).
3. The Value of Availability. Sombers Assosiates, Inc. and W.H. Highleyman. 2011. URL: http://www.availabilitydigest.com/public_articles/0606/value_of_availability.pdf (дата обращения: 2018-01-23).
4. Микрин Е.А., Сомов С.К. Обзор моделей и методов обеспечения сохранности данных в распределенных системах обработки данных / Информационные технологии и вычислительные системы, 4/2017. С. 5-28.
5. Kulba V.V., Somov S.K. Problem of optimal placement of data files in large-scale unreliable distributed systems / 2017 Tenth International Conference Management of Large-Scale System Development (MLSD), Moscow, Russia, 2017, pp. 1-5. URL: http://ieeexplore.ieee.org/document/8109649/ (дата обращения 2018-01-10).
6. Таненбаум Э., ван Стеен М. Распределенные системы. Принципы и парадигмы/ Пер. с англ. — СПб.: Питер, 2003. — 877 с. (A.S. Tanenbaum, M. van Steen. 2003. Distributed Systems: Principles and Paradigms. St. Petersburg.: Piter. 877 p.
7. Чернышев Г.А. Обзор подходов к организации физического уровня в СУБД// Труды СПИИРАН. - Санкт-Петербург, 2013. Вып. 1(24). - С. 222 – 275.
8. Chu W.W, File Allocation in a Multiple Computer System. IEEE Transactions on Computers, 1969, V. C-18, N. 10, p. 885-889.
9. Azzam Sleit and oth. A Dynamic Object Fragmentation and Replication Algorithm In Distributed Database

Systems//American Journal of Applied Sciences 4 (8): 613-618, 2007
10. Szymaniak M., Pierre G., Steen V. Latency-Driven Replica Placement//IPSJ Digital Courier. – 2006. – Vol.2. p.12
11. Rabinovich, M., Rabinovich, I., Rajaraman, R., and Aggarwal, A.: «A Dynamic Object Replication and Migration Protocol for an Internet Hosting Service» // Proc. 19th Int'l Conf. on Distributed Computing Systems. ACM, - 1999. pp. 101-113.
12. Singh. A., Kahlon S.K., Virk R.S. Nonreplicated Static Data Allocation in Distributed Databases Using Biogeography-Based Optimization// Chinese Journal of Engineering, 2014. p. 1-9.
13. Кристофидес Н. Теория графов. Алгоритмический подход / Пер. с англ. – М.: Мир, 1978. – 432 с.
14. Reese J. Methods for Solving the p–Median Problem: An Annotated Bibliography // Networks. – 2006. – Vol. 48. N 3. – P.125–142.
15. Daskin M.S., Maass K.L. The p–Median Problem // Location Science. Springer. – 2015. – P. 21–45. – URL: https://link.springer.com/book/10.1007/978-3-319-13111-5#toc (дата обращения 2018-01-10).
16. Mladenovic N., Brimberg J., Hansen P., The p-median problem: A survey of metaheuristic approaches // European Journal of Operational Research. – 2007. – Vol. 179. N 3. – p. 927–939.
17. Teitz M. B., Bart P. Heuristic methods for estimating the generalized vertex median of a weighted graph // Operations Research. – 1968. – Vol.16. – P. 955-961.
18. Hauglid J. O., Ryeng N. H., Norvag K. Dyfram: dynamic fragmentation and replica management in distributed database systems // Distrib. Parallel Databases. – 2010. – Vol. 28. - P. 157–185.
19. Loukopoulos T., Ahmad I., Papadias D. An Overview of Data Replication on the Internet // Proceedings of the International Symposium on Parallel Architectures, Algorithms and Networks (ISPAN.02). – 2002. – P.6.

Сомов Сергей Константинович. Институт проблем управления им. В.А. Трапезникова РАН, г. Москва. Старший научный сотрудник, кандидат технических наук. Количество печатных работ: более 30 (в т.ч. 2 монографии). Область научных интересов: распределенные системы, сохранность информации, резервирование и репликация данных.
E-mail: ssomov2016@ipu.ru

Data replication as a tool to improve the reliability of distributed systems

S.K. Somov

V. A. Trapeznikov Institute of Control Sciences of Russian Academy of Sciences, Moscow, Russia

The problem of improving the reliability of distributed data processing systems is considered. A method for replicating data sets is presented as an effective tool for ensuring a high level of data safety in distributed systems. Examples of methods of static and dynamic replication are given. The formulation and algorithm of the problem of finding the optimal distribution of replicas over nodes of a computer network is given. An example of operation of the proposed algorithm for solving the formulated problem is presented. The analysis of the results obtained by the algorithm are analyzed.
Keywords: distributed data processing systems, data replication

DOI 10.14357/20718632180307

References

1. Vosstanovlenie dannyh [Data recovery]. 2018. Available at: http://www.datarecovery.ru/datarecovery.htm (Accessed January 22, 2018).
2. Frolov A., Frolov G. 2001. Sohrannost' i vosstanovlenie komp'juternyh dannyh: teorija i praktika [Safety and recovery of computer data: theory and practice]. Byte. Russia. 1(30). Available at: https://www.bytemag.ru/articles/detail.php?ID=9020 [Accessed November 01, 2017].
3. The Value of Availability. Sombers Assosiates, Inc. and W.H. Highleyman. 2011. Available at: http://www.availabilitydigest.com/public_articles/0606/value_of_availability.pdf (Accessed January 23, 2018).
4. Mikrin E.A., Somov S.K. 2017. Obzor modelej i metodov obespechenija sohrannosti dannyh v raspredelennyh sistemah obrabotki dannyh [Overview of models and methods to ensure information integrity in a distributed data processing systems]. Journal of information technologies and computing systems, 4/2017. p. 5-28.
5. Kulba V.V., Somov S.K. Problem of optimal placement of data files in large-scale unreliable distributed systems. 2017 Tenth International Conference Management of Large-Scale System Development (MLSD), Moscow, Russia, 2017, pp. 1-5. Available at: URL: http://ieeexplore.ieee.org/document/8109649/ (Accessed January 10, 2018).
6. Tanenbaum A.S., M. van Steen. 2003. Distributed Systems: Principles and Paradigms. St. Petersburg.: Piter. 877 p.

7. Chernyshev G.A. 2013. Obzor podhodov k organizacii fizicheskogo urovnja v SUBD [Overview of approaches to the organization of the physical layer in the DBMS]. Proceedings of SPIIRAS. St. Petersburg. 1(24):222 – 275.
8. Chu W.W. 1969. File Allocation in a Multiple Computer System. IEEE Transactions on Computers. C-18(10):885-889.
9. Azzam Sleit and oth. 2007. A Dynamic Object Fragmentation and Replication Algorithm In Distributed Database Systems. American Journal of Applied Sciences 4 (8):613-618.
10. Szymaniak M., Pierre G., Steen V. 2006. Latency-Driven Replica Placement. IPSJ Digital Courier. 2:12.
11. Rabinovich, M., Rabinovich, I., Rajaraman, R., and Aggarwal A. 1999. A Dynamic Object Replication and Migration Protocol for an Internet Hosting Service. Proc. 19th Int'l Conf. on Distributed Computing Systems. ACM. 101-113.
12. Singh. A., Kahlon S.K., Virk R.S. 2014. Nonreplicated Static Data Allocation in Distributed Databases Using Biogeography-Based Optimization. Chinese Journal of Engineering. 9p.
13. Christofides N. Graph Theory: An Algorithmic Approach. Academic Press. 1975.
14. Reese. J. Methods for Solving the p–Median Problem: An Annotated Bibliography. 2006. Networks. 48(3):125–142.
15. Daskin M.S., Maass K.L. 2015. The p–Median Problem. Location Science. Springer. 21–45. Available at: https://link.springer.com/chapter/10.1007/978-3-319-13111-5_2 (Accessed January 10, 2017).
16. Mladenovic N., Brimberg J., Hansen P. 2007. The p-median problem: A survey of metaheuristic approaches. European Journal of Operational Research. 179(3):927–939.
17. Teitz M. B., Bart P. 1968. Heuristic methods for estimating the generalized vertex median of a weighted graph. Operations Research. 16:955-961.
18. Hauglid J. O., Ryeng N. H., Norvag K. 2010. Dyfram: dynamic fragmentation and replica management in distributed database systems. Distrib. Parallel Databases. 28:157–185.
19. Loukopoulos T., Ahmad I., Papadias D. 2002. An Overview of Data Replication on the Internet. Proceedings of the International Symposium on Parallel Architectures, Algorithms and Networks (ISPAN.02). 6 p.

Somov Sergej Konstantinovich. Senior researcher, V. A. Trapeznikov Institute of Control Sciences of Russian Academy of Sciences. Candidate of Science, Engineering. Graduated from Nizhny Novgorod State University named after N.I. Lobachevsky in 1977. Author of more than 30 publications, including 2 monographs. Area of expertise: distributed system, information integrity, data replication and reservation. E-mail: ssomov2016@ipu.ru