обращения: 26.04.2022).
3. *Жигач А.* Почему интернет-мессенджеры снова стали популярными. URL: http://www.dp.ru/a/2015/04/07/ Tak_v_chem_zhe_messedzh (дата обращения: 26.04.2022).
4. Oracle Corporation Что такое чат-бот? URL: https://www.oracle.com/cis/chatbots/what-is-a-chatbot/ (дата обращения: 26.04.2022).

УДК 004.4

КЭШ-ХРАНИЛИЩЕ НА ГРАФОВОЙ СТРУКТУРЕ ДАННЫХ

Халикова А.И.
Научный руководитель – Прихожий А.А., д.т.н., профессор

Кэш является промежуточным уровнем между быстрым процессором и медленной основной памятью. Он предназначен для хранения копий часто используемых данных и сокращения времени доступа к основной памяти. В основном кэш в промышленном программировании представлен сервисами кэширования, осуществляющими кэширование в виде пар “ключ - значение”. При анализе конкретной доменной области, которую реализует то или иное приложение, иногда возникает потребность в иной структуре данных кэш-хранилища. Данная статья представляет описание системы кэш-хранилища, построенного на графовой структуре данных.

Основная идея такой системы заключается в размещении ключей и комплексных значений для этих ключей в виде ориентированного графа, в котором повторяющиеся значения для различных ключей не дублируются при занесении в кэш, а вместо этого используются ссылки на эти значения. Также такие значения оптимизируют алгоритм вытеснения данных из кэш. В сравнении с классическими алгоритмами вытеснения кэш, такими как LRU, LFU, MRU, в кэш-хранилище учитывается наличие активных ссылок на узлы-значения, что позволяет очищать кэш рациональнее.

Проведем сравнение хранения сериализованных данных в формате JSON и в графовой структуре данных и в виде пар ключ-значение. Рис. 1 дает пример роедставления данных в формате JSON.

```
["id": 5547]:{
    "name": "Client CA",
    "industryName": "RE",
    "numberOfAccounts": 72,
    "formYear": 2021,
    "jurisdiction": "CA",
},
```

```
["id": 6268]:{
    "name": "Client CA 2",
    "industryName": "RE",
    "numberOfAccounts": 10,
    "jurisdiction": "CA",
    "formYear": 2021}
```

Рис. 1. Сериализованные данные в формате JSON

Теперь представим эти же данные, но в виде ориентированного графа (Рис.2). Как видно из рисунка, мы уменьшили количество дублирующихся записей почти в два раза. Такое размещение значений по ключам позволяет реализовать основную идею построения кэш-хранилища, задав ключи и значения в виде ориентированного взвешенного графа G = (V, E), где V – множество вершин графа, представляющих ключи и их значения, а E – множество ориентированных ребрер, направленных от ключей к их значениям.

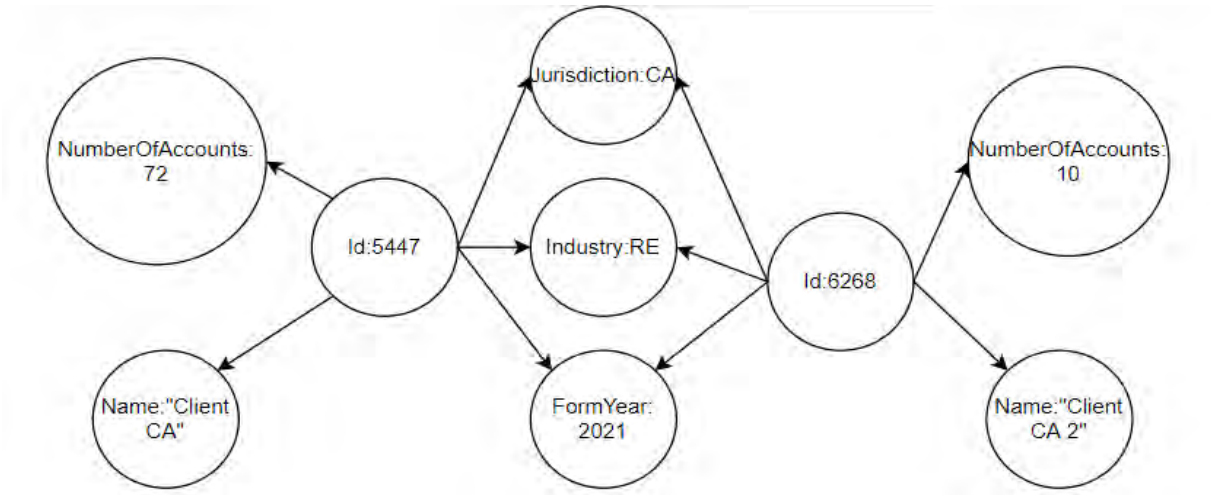

Рис. 2. Представление данных ориентированным графом

В качестве весов ребер будем использовать времена создания ключей и их значений в кэш-хранилище. Так, используя политику замены LRU, будем раскрывать вершины, к которым ведут ребра с наибольшими весами и очищать кэш, если к вершинам, к которым ведут такие ребра, не ведут другие ребра, т.е. вершина не является частью другого ключа.

Так как основной структурой данных для хранения объектов, представляющих вершины графа, была выбрана хеш-таблица, немаловажным пунктом будет выбор метода разрешения коллизий, т.к. они неизбежны при использовании данной структуры данных.

Основными операциями, которые должна поддерживать хеш-таблица, являются вставка, поиск и удаление элементов. Проведем анализ коэффициента заполнения α в обоих методах. Пусть у нас есть хеш-таблица T с m ячейками, в которых хранятся n элементов. Определим коэффициент заполнения таблицы T как $\alpha = n/m$, т.е. как среднее количество элементов, хранящихся в одной цепочке.

Если количество ячеек в хеш-таблице, как минимум, пропорционально количеству элементов, хранящихся в ней, то $n = O(m)$ и, следовательно, $\alpha = n/m = O(m)/m = O(1)$, а значит, поиск элемента в хеш-таблице в среднем требует постоянного времени. Поскольку в худшем случае вставка элемента в хеш-таблицу занимает O (1) времени (как и удаление элемента

при использовании двусвязных списков), можно сделать вывод, что все словарные операции в хеш-таблице в среднем выполняются за время O (1).

Выбор стратегии разрешения коллизий накладывает на эти операции определенные ограничения. Процедура удаления из хеш-таблицы с открытой адресацией достаточно сложна. При удалении ключа из ячейки i мы не можем просто пометить ее значением *NIL*. Поступив так, мы можем сделать невозможным выборку ключа k, в процессе вставки которого исследовалась и оказалась занятой ячейка i. Одно из решений состоит в том, чтобы помечать такие ячейки специальным значением *DELETED* вместо *NIL*. При этом мы должны слегка изменить процедуру *HASH_INSERT*, с тем чтобы она рассматривала такую ячейку как пустую и могла вставить в нее новый ключ. В процедуре *HASH_SEARCH* никакие изменения не требуются, поскольку мы просто пропускаем такие ячейки при поиске и исследуем следующие ячейки в последовательности. Однако при использовании специального значения *DELETED* время поиска перестает зависеть от коэффициента заполнения α, и по этой причине, при необходимости удалений из хеш-таблицы в качестве метода разрешения коллизий выбирается метод цепочек.

Для снижения вероятности промаха кэш (cache miss) предусмотрено использование фильтров Блума. Фильтр Блума представляет собой битовый массив из m бит и k различных хеш-функций $h_1 \ldots h_k$, равновероятно отображающих элементы исходного множества в множество $\{0,1,\ldots m-1\}$, соответствующее номерам битов в массиве. Изначально все m бит обнулены. Для добавления элемента e необходимо записать единицы на каждую из позиций $h_1(e) \ldots h_k(e)$ битового массива. Чтобы проверить, что элемент e принадлежит множеству хранимых элементов, необходимо проверить состояние битов $h_1(e) \ldots h_k(e)$. Если хотя бы один из них равен нулю, элемент не принадлежит множеству. Если все они равны единице, то структура данных сообщает, что элемент принадлежит множеству. При этом могут возникнуть две ситуации: либо элемент действительно принадлежит множеству, либо все эти биты оказались установлены при добавлении других элементов, что и является источником ложных срабатываний в этой структуре данных. По сравнению с хеш-таблицами, фильтр Блума может обходиться на несколько порядков меньшими объемами памяти, жертвуя детерминизмом.

Таким образом, в данной статье рассмотрен подход к организации кэш-хранилища, позволяющий оптимизировать затраты на ресурс памяти, а также временной ресурс, затрачиваемый системой на проведение основных операций вставки, поиска и удаления. Также система является весьма гибким решением для кеширования данных предметных областей, в которых имеет место изменение часто используемых данных, так как она

позволяет перезаписывать измененные значения для всех ссылающихся на эти значения ключей.

Литература

1. Prihozhy A.A. Simulation of direct mapped, k-way and fully associative cache on all pairs shortest paths algorithms. System analysis and applied information science. – 2019, No. 4, pages 10–18.
2. Prihozhy, A.A. Analysis, transformation and optimization for high performance parallel computing. Minsk: BNTU, 2019. – 229 p.
3. Кормен, Т.Х., Лейзерсон, Ч.И., Ривест, Р.Л., Штайн, К. В24 Алгоритмы: построение и анализ, 2-е издание. : Пер. с англ. — М. : Издательский дом “Вильямс”, 2011. — 1296 с.

УДК 004.415.2

WEB-СЕРВИС ПО ОКАЗАНИЮ СТРАХОВЫХ УСЛУГ

Дубоделов А.В., Балкис И.С.
Научный руководитель – Прибыльская Н.М., ст. преподаватель

В Республике Беларусь существует как обязательное, так и добровольное страхование. Обязательное страхование – это вид страхования, при котором страховые договоры заключаются не по желанию клиента, а в силу обязательства, предусмотренного законом [1].

В Республике Беларусь страховую деятельность осуществляют шестнадцать различных компаний.

К самым актуальным видам добровольного страхования в Республике Беларусь относятся: страхования здоровья и страхование имущества. В связи с тем, что особо ценным движимым и недвижимым имуществом в РБ обладают далеко не все граждане, более актуальным для большинства жителей страны является медицинское страхование.

Самой большой страховой компанией в РБ с чистой прибылью в пятьдесят четыре миллиона белорусских рублей является «Белгосстрах», что несоизмеримо мало с мировыми практиками.

Такой разрыв можно обусловить несколькими факторами. Множество операций по управлению, администрированию, менеджменту осуществляются в ручном режиме, что приводит к неэффективной модели управления бизнесом. Большая часть взаимодействия с клиентом также проходит офлайн режиме, что не позволяет оперативно реагировать на страховые случае, моментально осуществлять выплаты по ним, а также делает невозможным быстрое подключение большого количества клиентов, например корпоративное страхование.