УДК 004.4

СРЕДСТВА ХЕДЖИРОВАНИЯ И МОНИТОРИНГА В YTSAURUS JAVA SDK ДЛЯ СЕРВИСОВ ДОСТУПА К ЭМБЕДДИНГАМ РЕКОМЕНДАТЕЛЬНОЙ СИСТЕМЫ

Семенов Г.В. (Университет ИТМО, VK)
Научный руководитель – к.т.н. Васильев А.В. (VK)

Введение. В основе современных рекомендательных систем лежат эмбеддинги – векторные разложения, на основе которых производится инференс моделей машинного обучения на различных этапах рекомендательного пайплайна. Решающим фактором при проектировании архитектуры рантайма высоконагруженных рекомендательных систем является высокая доступность и отказоустойчивость средств обеспечения доступа к пользовательским эмбеддингам, реализуемых на основе конкретного хранилища больших данных.

Платформа для распределенного хранения и обработки больших данных YTsaurus [1], разрабатываемая в Яндекс и развиваемая в open-source с 2022 г., позволяет реализовать хранение пользовательских эмбеддингов в динамических таблицах. Однако для обеспечения надежности доступа к эмбеддингам необходимы дополнительные механизмы для компенсации частичных отказов кластеров YTsaurus, снижения числа параллельных запросов к RPC-прокси (которые на практике являются точкой отказа платформы), а также средства для мониторинга динамики выполнения запросов к нескольким кластерам.

В работе рассматриваются потокобезопасная реализация хеджирующего клиента MultiYTsaurusClient на основе конечного автомата состояний с механизмами мониторинга запросов и реализация метода MultiLookupRows в API динамических таблиц, реализованные автором в YTsaurus Java SDK в процессе разработки средств обеспечения доступа к эмбеддингам в рекомендательных системах.

Основная часть. Для хранения эмбеддингов в YTsaurus могут быть использованы *динамические таблицы*, реализующие интерфейс точечного чтения и записи данных по ключу. Запись данных в реплицированных динамических таблицах производится в master-кластер, а чтение может быть произведено с любого из slave-кластеров. Синхронизация данных реализуется с помощью репликатора, применяющего лог операций к slave-кластерам в фоновом режиме, при этом обеспечивается слабая согласованность данных (weak consistency). Пользовательские эмбеддинги в динамических таблицах могут быть представлены как кортежи с полями пользовательского идентификатора, данными эмбеддинга и дополнительными метаданными, опционально включающими тип, версию и дату выпуска разложения. Ключ таблицы может включать как пользовательский идентификатор, так и тип эмбеддинга.

При эксплуатации реплицированных динамических таблиц следует учитывать частичный отказ каждого кластера, часто проявляющийся в ошибке «Chunk data not preloaded», которая может возникать из-за периодической работы репликатора или отказа единичного таблета. Для обеспечения надежности системы требуется *хеджирование* (hedging) [2] – метод повышения надежности и минимизации времени отклика при запросах к удаленным узлам, заключающийся в спекулятивной отправке дублирующих (*хеджирующих*) запросов в пессимистичном предположении об отказе узла при выполнении первоначального запроса. В YTsaurus Java SDK реализацией хеджирующего клиента является класс MultiYTsaurusClient.

С*тратегия устойчивости* (resilience strategy) хеджирующего клиента определяет условия эскалации дублирующих запросов: истечение *мягкого таймаута* запроса (soft timeout), соблюдение бюджета повторных запросов (attempts), соблюдение бюджета запросов по типам

ошибки (4xx, 5xx, таймаут и пр.), использование пула стабильных узлов (circuit breaking) и др. Применительно к задаче получения эмбеддингов из таблиц YTsaurus достаточно единожды хеджировать запрос на соседние кластеры после истечения мягкого таймаута, а также выводить из использования отказывающие кластеры с помощью временных штрафов.

Для корректности применения метода запрос должен соответствовать идемпотентной операции, а в случае наличия множественных узлов требуется спецификация модели согласованности данных, возвращаемых различными репликами. Для задачи получения эмбеддингов такой идемпотентной операцией является LookupRows динамических таблиц, а соответствующая модель согласованности – слабая, т.е. на разных кластерах могут находиться отличающиеся версии пользовательских эмбеддингов.

В прежней реализации MultiYTsaurusClient из-за отсутствия явного состояния отмененного подзапроса тривиальное добавление callback-вызовов для мониторинга исполнения запросов было невозможным, поэтому был предложен рефакторинг прежней реализации, вводящий явный конечный автомат состояний запроса в классе MultiExecutorRequestTask: IN_PROGRESS, COMPLETED, CANCELLED. Результат всего запроса определяется либо первым успешным подзапросом, либо последним неуспешным подзапросом.

Новый интерфейс MultiExecutorMonitoring позволяет подписываться на события успешного и неуспешного выполнения запросов и подзапросов, а также на события запуска подзапросов. В штатном случае при равномерном распределении нагрузки по YTsaurus-кластерам число событий запуска подзапросов по каждому кластеру должно быть распределено равномерно, а их суммарное число может незначительно превышать общее число запросов. Особенно практически полезной визуализацией является рейт событий успешного выполнения запросов по кластерам на одном графике, позволяющий заметить падение вклада в успешные запросы конкретного кластера. Также важной визуализацией является график доли различных типов ошибок по YTsaurus-кластерам, позволяющий оценивать SLA кластеров.

Поскольку на практике для единичной выдачи рекомендательной системе требуются десятки различных типов эмбеддингов для пользователя, представляется важным группировать запросы, поскольку в ином случае неизбежно будет происходить перегрузка RPC-прокси, а клиенты станут получать ошибку «Request queue size limit». Для этого YTsaurus наряду с методом LookupRows поддерживает метод MultiLookupRows, поддержка которого была добавлена в Java SDK впервые. С его помощью стало возможно группировать запросы на получение эмбеддингов из разных таблиц и делегировать исполнение сложного запроса планировщику YTsaurus.

Выводы. Таким образом, были рассмотрены реализованные автором улучшения клиента динамических таблиц в YTsaurus Java SDK 1.2.7, позволяющие реализовать отказоустойчивый доступ к эмбеддингам рекомендательных систем с мониторингом выполнения запросов. Направления дальнейших исследований включают разработку математической модели отказов YTsaurus-кластеров с целью подбора мягкого таймаута, максимизирующего SLA средств доступа к эмбеддингам.

Список использованных источников:

1. Документация YTsaurus [Электронный ресурс]. URL: https://ytsaurus.tech/docs/ru/
2. G. Orosz. Resiliency in distributed systems [Электронный ресурс] // The Pragmatic Engineer Newsletter – URL: https://blog.pragmaticengineer.com/resiliency-in-distributed-systems/