в) Удобство использования – высокая стоимость «ремонтопригодности». Необходимо написать большое количество интерфейсов для классов с очень малыми обязанностями.

Таким образом, VIPER помогает нам быть более точными, что касается разделения проблем, разделяя большое количество кода одного класса на несколько меньших классов. За счёт поддержания единственной ответственности в каждом классе это упростит разработку классов, используя разработку через тестирование (англ. TDD), которое позволяет нам более быстро реагировать на изменяющиеся требования и создавать лучшее программное обеспечение. В тоже время использовать данную архитектуру неудобно при отсутствии большого количества бизнес-логики на экран. Возникают проблемы при необходимости использования повторно одних и тех же экранов при разных сценариях, так как необходимо хранить зависимости между модулями, который сейчас активен и который будет показан следующим. Сложно «наследоваться» от других модулей приложения.

БЫСТРОЕ ОПРЕДЕЛЕНИЕ ПРИНАДЛЕЖНОСТИ ПОЛЬЗОВАТЕЛЯ К СЕГМЕНТУ ПО IP-АДРЕСУ

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Базаревский В. Э.

Брацевич П. Ю. – кандидат технических наук

Рассматриваются алгоритмы быстрого определения принадлежности пользователя к сегменту IP-адресов, особенности хранения и организации сегментов для ускорения работы. Рассмотрена структура данных дерево отрезков, особенности работы с ним и построение.

В современных интернет-технологиях одним из основных способов монетизации приложений является продажа интернет-рекламы, тем самым делая её одним из важнейших элементов экосистемы сети интернет. Вместе с тем следует отметить следующий тренд вместо показа рекламы на определенных сайтах, становится возможным эффективно показывать рекламу её целевой аудитории за счет четкого таргетирования по интересам. Одним из наиболее распространенных методов таргетирования рекламы, является сегментации пользователей на основе их интересов, географического местоположения, используемых устройств, гендерных и иных признаков.

Среди различных средств сегментирования пользователей следует выделить метод сегментирования пользователей по IP-адресам. Так как фактически имеет место географическая сегментация пользователей, то на основе дополнительных сведений об отличительных особенностях абонентов конкретных IP-адресов результирующие сегменты несут в себе более сложную информацию (например, интересы, возраст, пол, профессию и т.д. пользователей). В качестве примера, всех абонентов, выходящих с IP-адресов, принадлежащих университету, можно отнести к сотрудникам университета или студентам.

Зачастую, сегментация по IP является не тривиальной задачей, так как пространство IP-адресов сильно разрежено, представляет собой большое количество интервалов разной длинны, при этом сильно пересекающихся друг с другом.

Ввиду описанных выше требований, необходима реализация эффективной структуры данных для определения сегмента по отдельно взятому IP-адресу, оптимальной с точки зрения поиска, масштабируемости и потребления ресурсов. Для решения этой задачи был предложен ряд решений, которые помогли оптимизировать процесс определения принадлежности пользователя к тому или иному сегменту.

Сегмент представляется в виде областей IP-адресов с начальным и конечным IP-адресами. IP-адрес в 4-ой версии представляет собой 32-битовое число. Удобной формой записи IP-адреса (IPv4) является запись в виде четырёх десятичных чисел значением от 0 до 255, разделённых точками. Для того чтобы оптимизировать операцию сравнения адресов и ускорить проверку на вхождение адреса в сегмент, IP-адрес из строкового представления был конвертирован к типу Long. В таком численном виде операция сравнения занимает меньше процессорного времени.

Кроме задачи сравнения адресов существует задача организации сегментов в структуру, оптимизирующую операцию поиска и определения принадлежности IP-адреса к конкретному сегменту. Самый простой способ решать представленную задачу, это завести линейный массив. При такой реализации время нахождения ответа на запрос имеет порядок O(n). Изменение же значения какого-либо сегмента требует O(1) времени. Операция изменения значения для выбора архитектуры представления сегментов в данном случае является не определяющей, так как изменение сегментов происходит редко, а операция поиска в свою очередь выполняется при каждом обращении клиента к сервису. Альтернативной структурой организации данных о сегментах пользователей для оперативного поиска является дерево отрезков.

Дерево отрезков представляет собой дерево, листьями которого являются элементы исходного массива. Остальные вершины являются результатом какой-либо операции над детьми. В нашем случае удобнее всего выбрать операцию объединения сегментов. Таким образом, корень хранит результат

Библиотека БГУИР

объединения всех сегментов, его левый ребенок - результат объединения элементов с индексом от 0 до (n / 2 - 1), а правый - результат объединения элементов с индексом от (n / 2) до n и так далее до листьев. Существует две реализации построения дерева отрезков, но обе они, как от корня к листьям, так и от листьев к корню, реализуются на базе массива, размерностью ближайшей следующей степени 2. При построении снизу алгоритм поднимается от листьев к корню. Для этого просто начинаем заполнять элементы массива от большего индекса к меньшему, таким образом при заполнении элемента i его дети 2 * i + 1 и 2 * i + 2 уже будут заполнены, а при построении сверху спускается от корня к листьям.

Алгоритм построения дерева сегмента является коротким и действенным способом решения подобных задач. Временная сложность O(logN), что значительно лучше, чем O(N). Например, при массиве длинной 10^9 элементов, необходимо примерно 32 сравнения. Изменить отрезок в этой структуре так же просто — надо пройти по всем вершинам от заданной до 1-ой, и заменить его. Это также занимает O(log N) времени.

Список использованных источников:

1. Н. Вирт Алгоритмы и структуры данных. СПб.: Невский диалект, 2001. – 352 с.
2. Т. Кормен и др. Алгоритмы: построение и анализ, М.: МЦНМО, 2002.

МОДЕЛЬ СИСТЕМЫ РАСПРЕДЕЛЕННОГО ХРАНЕНИЯ ДАННЫХ

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Бернацкий В.В.

Бранцевич П.Ю. - кандидат технических наук, доцент

Надежность, доступность, согласованность данных являются основными требованиями предъявляемыми к базам данных. Рост объема и трафика данных, а также использование распределенных вычислений требуют возможности горизонтальной масштабируемости баз данных. В докладе предложена модель системы распределенного хранения данных удовлетворяющая данным требованиям.

Целью данной работы является разработка модели системы распределенного хранения данных с сохранением основных требований к базам данных. Эта система должна быть отказоустойчивой, выход из троя отдельных узлов системы не должны приводить к остановке работы всей системы. Также эта система должна быть доступной: операции чтения и записи должны быть быстрыми. Операции записи не должны приводить систему в несогласованное состояние. Вся нагрузка в системе должна быть равномерно распределенной между всеми узлами (вершинами).

Исходя из требований к данной системе были выделены следующие задачи: определение структуры хранимых данных, обеспечение равномерного распределения нагрузки по всем узлам, обеспечение высокой доступности и согласованности данных.

Разрабатываемая система будет представлять собой коллекцию элементов ключ, имя признака, значение признака, где ключ является уникальным идентификатором логической сущности. Имя признака является именем атрибута этой логической сущности, а значение признака является значением атрибута заданного в поле имя признака. Данный подход позволяет обеспечить равномерное распределение данных основываясь на значении ключ, а добавление понятия признак позволяет уменьшить размер передаваемых данных и уменьшить количество неудавшихся транзакций (за счет применения изменений не ко всей сущности, а лишь к ее части).

Ключевым моментом для обеспечения высокой доступности и надежности является реплицируемость данных - хранение копий одних и тех же данных на нескольких вершинах. Это позволяет распределить запросы для работы с этими данными по вершинам, а в случае выхода из троя отдельных вершин продолжить работу с этими данными используя оставшиеся копии. В данной системе репликации будут подвержены сущности.

В распределенных системах хранения добиться полной согласованности данных практически невозможно, поэтому здесь применяют концепцию согласованности в конечном счете. Данная концепция является менее строгой формой согласованности. Согласно ней данные будут согласованы если никаких новых изменений не происходило. Для обеспечения выполнения данного условия можно использовать подход основной копии. Согласно нему есть 2 категории копий данных (сущности): основная и неосновная. Операции по изменению данных происходят лишь над основной копией данных, а все изменения затем транслируются на оставшиеся копии. Таким образом мы получаем надежность изменения данных как в синхронной модели, но скорость чтения увеличивается пропорционально количеству неосновных копий.

Равномерное распределение значений по узлам осуществляется за счет подсчета хеша ключа сущностей. В зависимости от того, в какой диапазон попало значение, определяется вершина для хранения сущности. Аналогичным образом определяются вершины для хранения других копий значений сущности.

Для обеспечения отказоустойчивости необходимо минимизировать вред от выхода из строя (недоступность) узлов системы. Это можно достичь путем равномерного распределения основных и неосновных копий сущностей и ведением списка приоритета узлов для каждой сущности. Данный список

Библиотека БГУИР