МЕТОД УДАЛЕННОЙ РАЗНОСТНОЙ СИНХРОНИЗАЦИИ ФАЙЛОВ ОРИЕНТИРОВАННЫЙ НА МИНИМИЗАЦИЮ ПЕРЕДАВАЕМОЙ ИНФОРМАЦИИ

A METHOD FOR REMOTE DIFFERENTIAL FILE SYNCHRONIZATION AIMED AT MINIMIZING THE TRANSMITTED INFORMATION

УДК 004.021

Архипов Д.Р., Бакалавр, факультет «Информатика и системы управления»
МГТУ им Н.Э. Баумана Россия, г. Москва
Arkhipov D. R., enfiol@mail.ru

Аннотация

Данная статья посвящена методу удаленной разностной синхронизации файлов ориентированному на минимизацию передаваемой информации. В статье рассматривается существующий и активно используемый метод rsync и его недостатки. Формируется формальное описания алгоритма разделения данных на сегменты динамического размера, а также формальное описание метода удаленной разностной синхронизации его использующее. Также приводятся графические примеры ситуаций, в которых метод более эффективно минимизирует передаваемую информацию.

Abstract

This article is devoted to the method of remote differential file synchronization focused on minimizing the transmitted information. The article discusses the existing and actively used rsync method and its disadvantages. A formal description of the algorithm for dividing data into segments of dynamic size is formed, as well as a formal

description of the remote difference synchronization method using it. There are also graphical examples of situations in which the method minimizes the transmitted information more efficiently.

Ключевые слова: синхронизация, вычисление разностей, динамические сегменты, хэш-суммы.

Keywords: synchronization, calculation of differences, dynamic segments, hash sums.

Множество существующих подходов к удаленной разностной синхронизации файлов ориентированы на достижение наименьшего времени выполнения, а используемые в них алгоритмы оптимизируются по сложности, однако существуют системы в которых время синхронизации не так важно как нагрузка на сеть. Для того чтобы уменьшить количество информации необходимой для передачи, в первую очередь необходимо рассмотреть существующий и активно используемый алгоритм rsync. В данной статье рассматривается односторонняя синхронизация. Файл, который синхронизируется, будет называться “синхронизируемым”, а файл, с которым производится синхронизация, “эталонным”.

Для того чтобы определить был ли изменен файл алгоритм rsync использует комбинацию из времени модификации файла и размера или определяет хэш-сумму его содержимого. Первый вариант имеет проблему неоднозначности, в обе версии файла могли быть внесены разные изменения одновременно, и изменение размера также могло совпасть. Альтернативным вариантом является более затратная по времени проверка хэш-суммы данных файла, которая производится например с использованием алгоритма MD5, но и она имеет небольшую вероятность коллизий. Предпочтительным является использование хэша, как более надежного варианта.

Для того чтобы минимизировать объем передаваемых данных, необходимо передавать только разницу между файлами. Вместо того чтобы искать разницу,

проще найти сегменты которые не были изменены, а все что останется обозначить как изменившееся.

Формальное описание алгоритма вычисления разностей rsync:

- синхронизируемый файл делится на сегменты фиксированного размера N;
- для каждого блока синхронизируемого файла вычисляются две функции данных:
    - 128-битный хэш с использованием алгоритма MD5, использующийся для точной идентификации данных блока;
    - "скользящая" хэш-сумма для быстрой идентификации содержимого.
- вычисленные хэш суммы отправляются для сравнения с "эталонным" файлом;
- в эталонном файле ищутся сегменты с эквивалентными хэш-суммами полученными из синхронизируемого файла. Блоки с одинаковыми хэш-суммами определяются как идентичные и вместо своего содержимого передают информацию только о своем положении (если оно изменилось);
- формируется список инструкций для приведения синхронизируемого файла к эталонному, после чего эти инструкции вместе с данными, которые не попали в неизменившиеся сегменты, передаются процессу запросившему синхронизацию.

Из формального описания алгоритма rsync можно сделать вывод, что данный алгоритм ориентирован на минимизацию времени синхронизации и осуществляет грубую минимизацию передаваемой информации. Разделение на блоки статического размера имеет свои недостатки. В случае, если были внесены изменения в сегменты и их содержимое сдвинулось, алгоритм rsync может неэффективно определить разность. Например, в случае, если в блок был добавлен символ, весь блок будет определен как изменившийся и подлежащий полной передаче. Также алгоритм rsync плохо показывает себя при изменениях связанных с перестановкой данных внутри файла.

Для передачи изменений, необходимых для приведения синхронизуемого файла к эталонному, необходимо сформировать инструкции, которые будут переданы стороне запросившей синхронизацию. Можно выделить основные виды операций производимых при редактировании файлов:

- вставка;
- удаление;
- изменение;
- перемещение.

Под вставкой понимается добавление символов в определенное место файла. Для того чтобы осуществить операцию вставки необходимо передать информацию о том куда она должна быть осуществлена и какие данные необходимо вставить. При вставке все данные находящиеся после места вставки сдвигаются "вправо" на ее длину.

Удаление требует передачи информации о месте удаления и количестве удаляемых символов. Все данные расположенные после удаляемой области сдвигаются "влево" на ее длину.

Для операции изменения передается информация о положении и длине изменяемой области, а также новые данные на которые необходимо произвести замену. Операция изменения не сдвигает данные находящиеся до и после нее.

Перемещение - представляет собой изменение положения блока данных. Для операции перемещения необходимо передать его старую и новую позицию в файле, а также его длину. Операция перемещения сдвигает "влево" все данные находящиеся после ее старой позиции, и "вправо" все данные находящиеся после ее новой позиции.

Исходя из описания данных операций, необходимо ориентироваться на минимизацию количества вставок и изменений, так как для их осуществления необходима передача данных.

Ниже приведено формальное описание предлагаемого метода удаленной синхронизации файлов ориентированного на минимизацию передаваемой информации, с учетом ранее указанных замечаний.

Формальное описание метода:

а. факт изменения файла определяется путем вычисления хэш-суммы данных. Если хэш сумма локального файла совпадает с хэш суммой удаленного, принимается факт того что в файл не были внесены изменения;
б. если было обнаружено изменение файла, то локальный файл делится на сегменты размера N_0;
в. для этих сегментов высчитывается хэш-сумма и передается для поиска в удаленном файле;
г. получается и сохраняется информация о неизменившихся сегментах;
д. оставшиеся части файла делятся на сегменты меньшего размера N_{i+1}, пока $N_{i+1} > N_{min}$, и для них повторяются пункты (в)-(д);
е. из информации о неизменившихся сегментах определяются и отделяются перемещенные сегменты;
ж. оставшиеся части файла классифицируются по операциям вставки, изменения и удаления;
з. формируется инструкция модификации файла и отправляется стороне запросившей синхронизацию;
и. получив набор инструкций проводятся соответствующие преобразования файла.

Коэффициенты N_{min} и N_{max} определяют минимальный и максимальный размер сегментов, в дальнейшем их оптимальные значения могут быть исследованы.

Определение операций перемещения возможно из информации о неизменившихся сегментах. Если сегмент находится не на своем месте (присутствует в эталонном файле, но нарушает порядок следования сегментов в синхронизуемом) - этот сегмент был перемещен. Например при содержимом исходного файла ABCDEF и содержимом "эталонного" ABFDC и размере сегмента в 1 символ, перемещенными сегментами будут являться C и D. Операции вставки и удаления могут быть определены исходя из расстояния между неизменившимися сегментами, учитывая изменения расстояний вызванные перемещением:

- если расстояние уменьшилось – удаление;
- если расстояние увеличилось – вставка.

Оставшиеся части файла определяются как операции изменения.

Сравнение разделения на сегменты алгоритма rsync и предложенного алгоритма приведено на изображениях ниже.

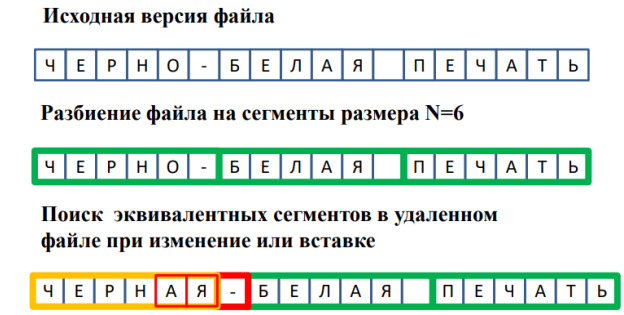

Рисунок 1: Разделение на сегменты алгоритмом rsync (при изменении или вставке)



Рисунок 2: Разделение на сегменты предложенным алгоритмом (при изменении или вставке)



Рисунок 3: Разделение на сегменты алгоритмом rsync (при перемещении)



Рисунок 4: Разделение на сегменты предложенным алгоритмом (при перемещении)

Данный метод может быть использован для осуществления задач синхронизации в системах, где минимизация сетевой нагрузки является более важным фактором чем скорость синхронизации (например, видео-хостинги и стриминговые сервисы).

Использованные источники:

1. Ramsey Norman. rsync. — URL: https://www.cs.tufts.edu/~nr/rsync.html (дата обращения: 22.03.2020).
2. Marc Stevens. Single-block collision for MD5. — URL: http://marc-stevens.nl/research/md5-1block-collision/ (дата обращения: 22.03.2020).
3. R. Rivest. The MD5 Message-Digest Algorithm. — URL: https://tools.ietf.org/html/rfc1321 (дата обращения: 22.03.2020).

Sources used:

1. Ramsey Norman. rsync. - URL: https://www.cs.tufts.edu/~nr/rsync.html (accessed: 22.03.2020).
2. Marc Stevens. Single-block collision for MD5. - URL: http://marc-stevens.nl/research/md5-1block-collision/ (accessed: 22.03.2020).
3. R. Rivest. The MD5 Message-Digest Algorithm. — URL: https://tools.ietf.org/html/rfc1321 (date accessed: 22.03.2020).