**1) Чем характеризуется понятие Big Data, проблема с какими характеристи-**

**ками считается Big Data проблемой. Какие аспекты решения Big Data про-**

**блем можно охарактеризовать.**

Интуитивно большие данные относятся к проблемам, когда данные — из-за их огромного объема, высокой скорости их создания или их сложности — превосходят традиционные подходы к их анализу или хранению.

Volume

Velocity

variety

**2) Что такое распределённое распределение. Какие проблемы решает распре-**

**делённое вычисление в контексте Big Data проблем.**

**Распределенные вычислительные системы** — это физические компьютерные, а также программные **системы**, реализующие тем или иным способом параллельную обработку данных на многих **вычислительных** узлах.

**3) Характеристика MapReduce схемы построения алгоритмов. В чём преиму-**

**щество MapReduce алгоритмов в распределении исполнения алгоритмов.**

**Что такое Map-этап алгоритма. Что такое Reduce-этап алгоритма.**

1. Есть несколько задач Map (распределителей), каждая из которых получает

одну или несколько порций файла из распределенной файловой системы.

Распределители преобразуют порцию в последовательность пар ключ-

значение. Как именно из входных данных порождаются эти пары, опреде-

ляет функция Map, написанная пользователем.

2. Пары ключ-значение от каждого распределителя собираются главным кон-

троллером и сортируются по ключу. Затем ключи раздаются задачам Reduce

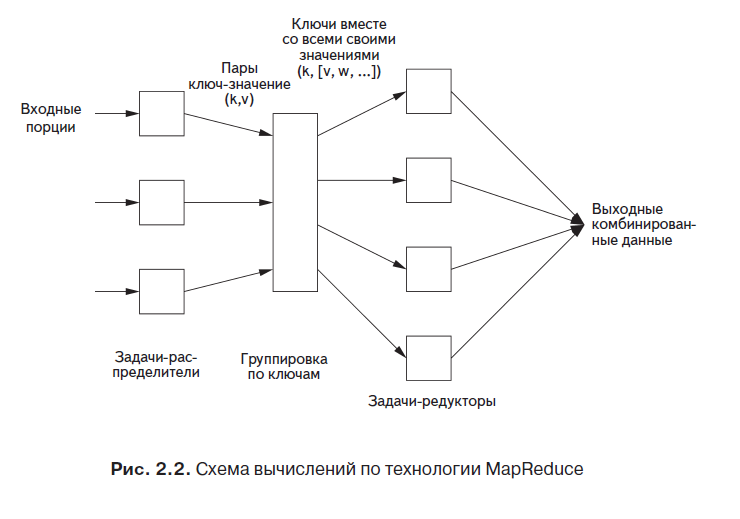
(редукторам), так что все пары с одинаковым ключом попадают одно-

му и тому же редуктору.

3. Редукторы обрабатывают по одному ключу за раз и каким-то образом ком-

бинируют значения, ассоциированные с этим ключом. Способ комбиниро-

вания определяется функцией Reduce, написанной пользователем.



Ключевыми достоинствами MapReduce являются следующие [2]:

1. возможность распределенного выполнения операций предварительной обработки (map) и свертки (reduce) большого объема данных. При этом функции map работают независимо друг от друга и могут выполняться параллельно на разных узлах кластера. Отметим, что на практике количество одновременно исполняемых функций map ограничивается источником входных данных и числом используемых процессоров. Аналогичным образом множество узлов производят свертку (reduce) после того, как каждый из них обработал все результаты функции map с одним конкретным значением ключа.
2. быстрота обработки больших объёмов данных за счет распределения операций по вышеописанному принципу. В частности, всего за пару часов MapReduce может отсортировать целый петабайт данных.
3. отказоустойчивость и оперативное восстановления после сбоев: при отказе рабочего узла, производящего операцию map или reduce, его работа автоматически передается другому рабочему узлу в случае доступности входных данных для проводимой операции.

**4) Communication-Cost модель. Характеризация и в чём её важность с точки**

**зрения исполнения Big Data алгоритмов.**

Коммуникационной стоимостью задачи будет размер ее

входных данных. Этот размер можно измерить в байтах. Но поскольку в наших

примерах будут использоваться операции в реляционных базах данных, то часто

мы будем выражать размер в кортежах.

Коммуникационная стоимость алгоритма равна сумме коммуникационных

стоимостей всех задач, реализующих этот алгоритм. Мы будем рассматривать

коммуникационную стоимость как основную меру эффективности алгоритма.

Объяснить и обосновать важность коммуникационной стоимости

можно следующим образом.

• Алгоритм, выполняемый каждой задачей, обычно очень простой, его время

работы часто линейно зависит от размера входных данных.

• Типичная скорость передачи данных между узлами вычислительного кла-

стера составляет один гигабит в секунду. На первый взгляд, очень много,

но это ничто по сравнению со скоростью выполнения команд процессором.

Более того, во многих кластерных архитектурах возникает конкуренция за

соединительный канал, когда несколько узлов хотят обмениваться данны-

ми одновременно. Таким образом, узел может выполнить огромный объем

работы над полученным входным элементом за время, которое требуется

для его доставки.

• Даже если задача выполняется в узле, где имеется копия порции данных,

которую эта задача обрабатывает, эта порция обычно хранится на диске,

и время, необходимое для ее перемещения в оперативную память, может

оказаться больше, чем время обработки данных, находящихся в памяти.

**5) Характеризация сложности алгоритма через Reducer Size и Replication**

**Rate. Взаимоотношения между Reducer Size и Replication Rate.**

Введем два параметра, характеризующие семейства алгоритмов MapReduce. Пер-

вый – размер редукции, который мы обозначим q. Это верхняя граница числа зна-

чений в списке, ассоциированном с одним ключом. При выборе размера редукции

руководствуются, по меньшей мере, двумя соображениями.

1. Выбор небольшого размера может привести к увеличению числа операций

редукции, т. е. количества различных ключей, по которым задача разбивает-

ся распределителями. Если мы также создадим много редукторов – вплоть

до отдельного редуктора на каждую операцию редукции – то повысится

степень параллелизма, и можно будет ожидать уменьшения физического

времени.

2. Можно выбрать размер редукции настолько малым, что соответствующее

вычисление заведомо будет производиться в оперативной памяти узла, на

котором размещен редуктор. Вне зависимости от характера вычисления

время выполнения редукции существенно уменьшится, если мы сможем

избежать постоянного перемещения данных между оперативной памятью

и диском.

Второй параметр – коэффициент репликации, который мы обозначим r. Он опре-

деляется как количество пар ключ-значение, порождаемых всеми распределите-

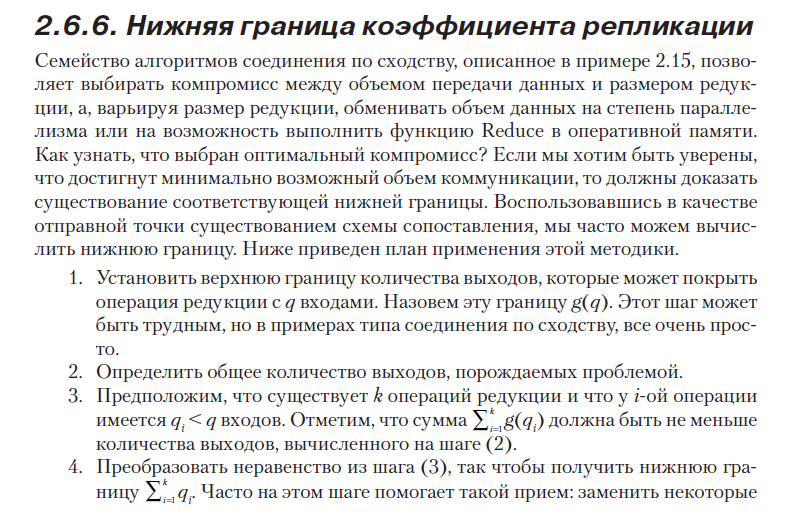
лями для всех входных файлов, поделенное на количество входных файлов. Иначе говоря, коэффициент репликации – это средний объем данных, передаваемых от

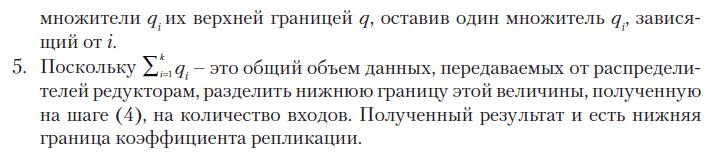
распределителей редукторам (измеренный в количестве пар ключ-значение), в

расчете на один входной файл.

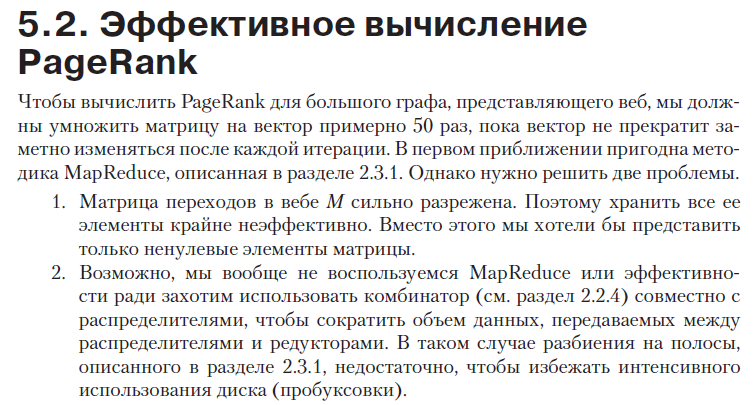
**6) Алгоритм построения нижней границы для Replication Rate. В чём может**

**быть польза от построения нижней границы.**





**8) Описание задачи PageRank и алгоритма её решения.**



**9) Описание MapReduce алгоритма перемножения матрицы на вектор.**

• Функция Map. Функция Map применяется к одному элементу M. Однако

если v еще не находится в оперативной памяти узла, на котором выполняется распределитель, то он сначала считывается – полностью – а затем ста-

новится доступен всем клиентам функции Map, выполняемой этим распре-

делителем. Каждый распределитель работает с порцией матрицы M. Для

каждого элементы матрицы mij он порождает пару (i, mijvj). Таким образом,

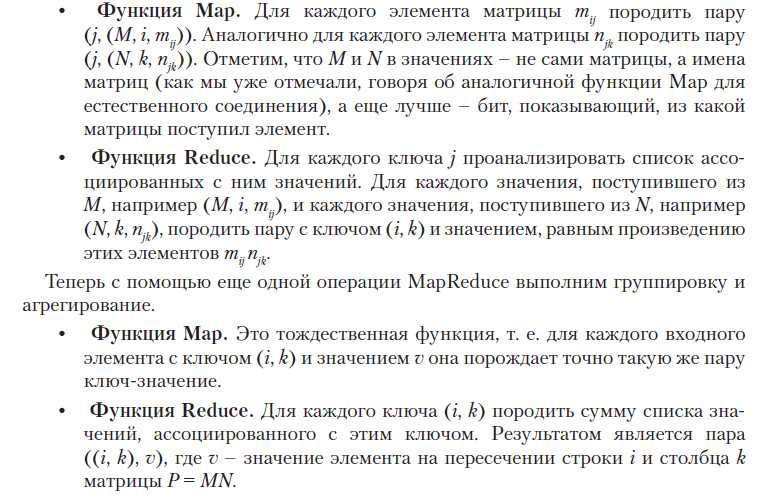
все члены суммы, составляющей компонент xi произведения матрицы на

вектор, получают один и тот же ключ i.

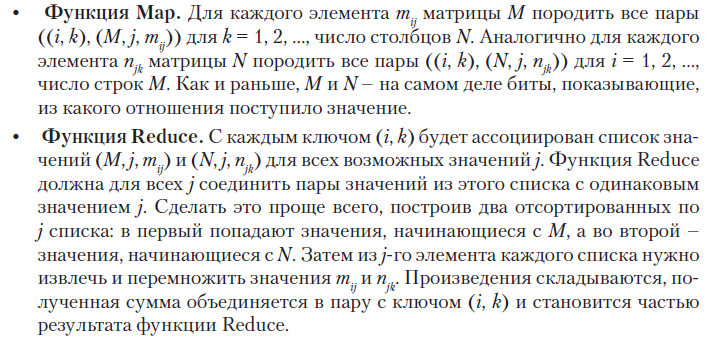
• Функция Reduce. Функция Reduce просто складывает все значения, ассо-

циированные с данным ключом i. В результате получается пара (i, xi).

**10) Описание MapReduce алгоритма перемножения матрицы на матрицу.**

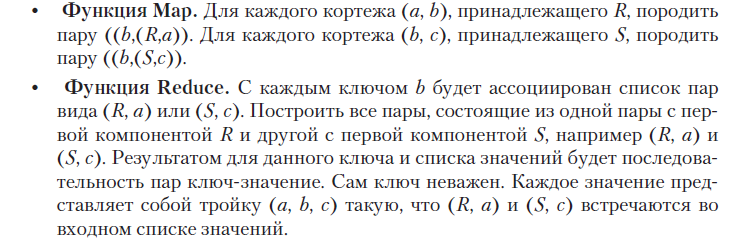


За один шаг



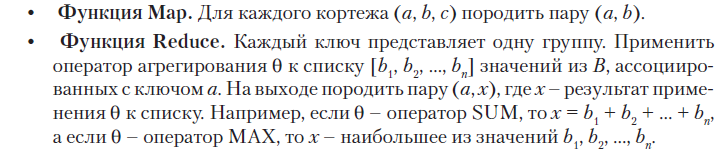
**11) Описание MapReduce алгоритма для объединения двух реляционных отно-**

**шений по ключу (Natural join).**

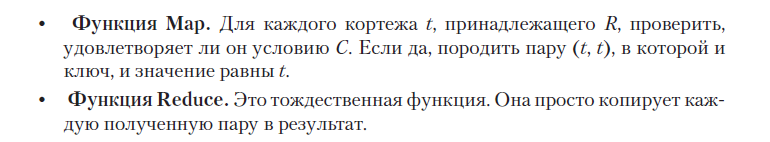


**12) Описание MapReduce алгоритма для группировки и агрегирования значе-**

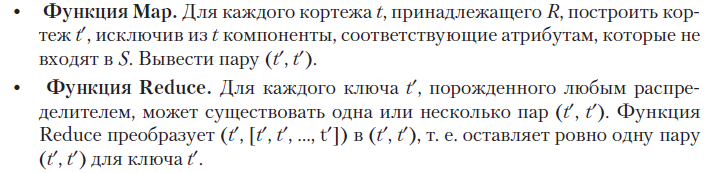
**ний реляционного отношения (Grouping and aggregation).**



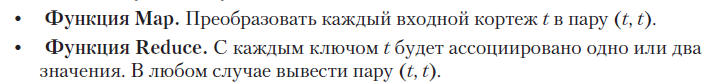
***2.3.4. Вычисление выборки с помощью MapReduce***



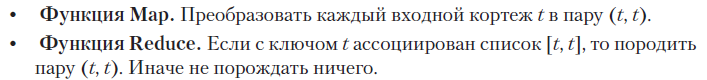
***2.3.5. Вычисление проекции с помощью MapReduce***



***2.3.6. Вычисление объединения и с помощью MapReduce***



***2.3.6. Вычисление пересечения с помощью MapReduce***



***2.3.6. Вычисление разности с помощью MapReduce***

