Параллельная реализация построения индекса поисковой системы с применением платформы Hadoop

Ф.В. Борисюк, В.И. Швецов, И.И. Белоусова

Нижегородский госуниверситет им. Н.И. Лобачевского
fedorvb@gmail.com

Введение

В данной статье рассматривается задача индексирования текстовой коллекции документов с использованием платформы для распределенных вычислений Hadoop [1].

Данная задача возникла при реализации поисковой системы основанной на алгоритме “Иерархической кластеризации по областям текстовых документов”, который подробно описан в [2]. При проведении вычислительных экспериментов [2] было обнаружено, что значительные вычислительные затраты требуются на этапе подготовки исходных данных для алгоритма кластеризации, когда каждый документ представляется в виде вектора ключевых слов (индексирование текстовой коллекции документов). Поэтому было решено применить вычислительную парадигму MapReduce [3] для реализации индексирования коллекции текстовых документов.

Основное внимание в работе уделяется применению парадигмы MapReduce к решению поставленной задачи. Приводятся результаты сравнения последовательного и параллельных вычислений.

Постановка задачи

Индексирование текстовой коллекции документов в поисковых системах представляет собой процесс добавления сведений о документах в поисковую базу данных. При построении индекса из документа выделяется набор наиболее значимых слов (ключевых слов). Индекс поисковой системы используется для улучшения скорости и быстродействия при поиске релевантных документов по коллекции текстовых документов. В рассматриваемой работе образом каждого документа в индексе является набор пар <слово, важность слова по отношению к документу>. Для оценки важности слова в документе использовалась формула TFxIDF, которая рассматривает, насколько часто слово встречается в данном документе при учете встречаемости в других документах [4]. При вычислении TFxIDF поискового индекса документа система считает частоту слова в документе, и вычисляет отношение частоты слова к количеству всех слов документа.

Построение индекса для больших коллекций текстовых документов требует больших временных затрат. К примеру, из проведенных в данной работе экспериментов для построения индекса одного гигабайта текстовых данных на одном вычислительном узле требуется 21 мин. 36 сек. Для сокращения времени построения индекса текстовой коллекции в данной работе предлагается использовать парадигму MapReduce.

Парадигма MapReduce

Программная модель MapReduce была предложена компанией Google для произведения распределенных вычислений. Концепция MapReduce состоит в том, что производимые вычисления разбиваются на функции Map и Reduce. Функция Map трансформирует входные данные в промежуточный список пар ключ/значение. Reduce функция берет список пар ключ/значение, который генерирует map и свертывает его по ключу (только одна пара ключ/значение для каждого ключа).

Для представленной задачи функции Map и Reduce будут иметь следующий вид:

```
map(String key, String value):
// key: имя документа
// value: текст документа
// ResultVector: вектор основ слов документа
ResultVector intermediateResult;
for each word w in value: {
        stem = ExtractStem(w);
          // добавить основу и посчитать их количество
          intermediateResult.add(stem);
}
EmitIntermediate(key, intermediateResult);

reduce(String key, Iterator values):
// key: имя документа
// values: список результатов с шага Map
ResultVector result;
PriorityQueue priorityqueue;
int i=0;
for each v in values: {
    intermediateResult = v;
        for each w in intermediateResult
        {
                priorityqueue.add(w,countWeight(w));
          }
          // берем ключевые слова с наибольшим весом
          while (i<=LIMIT)
          {
                result.add(priorityqueue.top());
                i = i+1;
          }
}
Emit(result);
```

Схема параллельных вычислений

Для хранения данных, которые необходимо обработать, в Hadoop используется распределенная файловая система (Hadoop Distributed File System, HDFS), при записи в которую, данные распределяются по вычислительным узлам сети. В реализации MapReduce от Hadoop процесс JobTracker на одном из узлов играет роль планировщика задач и распределителя кластера. Он распределяет задачи по TaskTracker'ам каждого узла кластера. JobTracker - это ведущий сервер MapReduce, а TaskTracker'ы - это ведомые узлы MapReduce. Таким образом, при запуске задачи, реализованной с использованием парадигмы MapReduce, на кластере Hadoop ее выполнение автоматически распределяется по вычислительным узлам сети. Map-процессы запускаются над подмножествами исходных данных и выполняются независимо друг от друга. Reduce-процессы обрабатывают результаты Map-фазы, разбивая их по значениям ключей на непересекающиеся блоки, что также позволяет выполнять их независимо.

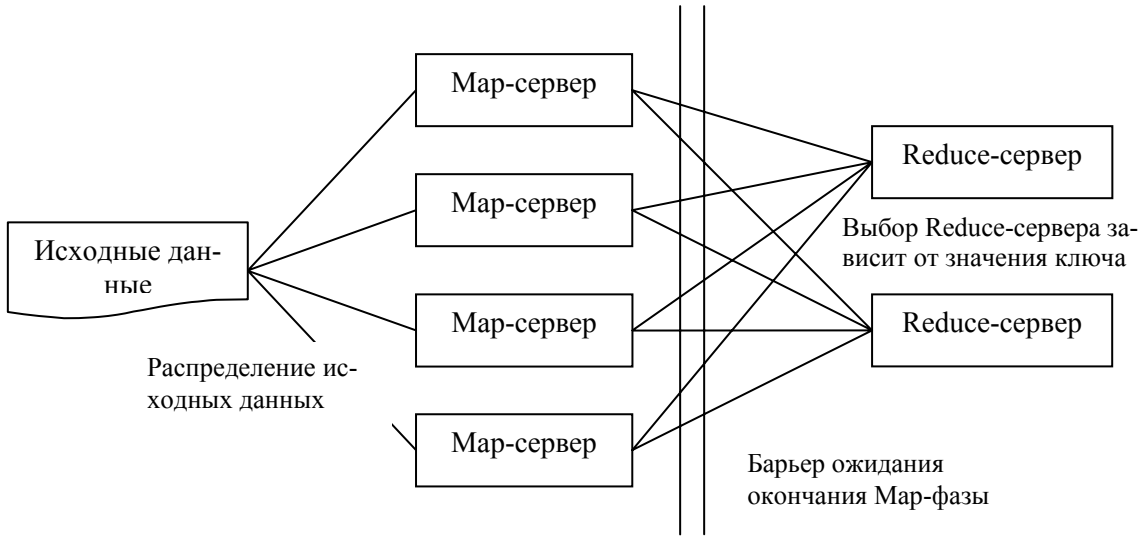

Рисунок 1. Схема вычислений MapReduce.

Таким образом, каждая из фаз может обрабатываться на любом количестве узлов параллельно. На рис.1 изображена схема вычислений MapReduce.

Результаты экспериментов, масштабируемость алгоритма.

Результаты вычислительных экспериментов были получены при использовании следующего программного и аппаратного обеспечения:

- Intel® Core(TM) 2 Duo CPU T2310 1.46 ГГц, 2 Гб оперативной памяти;
- AMD® Turion(TM) Mobile technology ML-34 1.8 ГГц, 1 Гб оперативной памяти;
- AMD® Turion(TM) Mobile technology TL-56 1.8 ГГц, 2 Гб оперативной памяти;
- Беспроводная сеть 50 Mbps;
- Microsoft Windows XP;
- Hadoop 19.2.

Проведенные вычислительные эксперименты показали, что на проведение индексирования текстовой коллекции документов размером 1Гб требуется 21 мин. 36 сек. Применение вычислительной парадигмы MapReduce, реализованной в Hadoop, на двух вычислительных узлах позволило нам, как показано в Таблице, ускорить процесс индексации примерно в 2 раза. В полностью распределенном режиме Hadoop работает с нескольких узлов с распределенными NameNode, JobTracker, узлами DataNode и TaskTracker. Для работы Hadoop в этом режиме необходимо как минимум три узла. Масштабирование на трех вычислительных узлах дает нам увеличение производительности практически в 3 раза.

Таблица. Ускорение работы приложения в зависимости от количества вычислительных узлов.

Операция	На одном выч. узле	На двух выч. узлах	На трех выч. узлах
Индексирование текстовой коллекции размером 1 Гб	21 мин. 36 сек.	10 мин. 47 сек.	7 мин. 45 сек.

Из проведенных экспериментов следует, что подобное распараллеливание теоретически может привести к линейному ускорению при увеличении количества вычисли-

тельных узлов. Это объясняется тем, что каждая из фаз MapReduce запускаются над подмножествами исходных данных (при этом важным фактором является их независимость), что позволяет выполнять обработку данных на любом количестве узлов параллельно.

Заключение

В результате применения парадигмы MapReduce удалось существенно сократить время построения индекса коллекции текстовых документов. Вычисления были проведены на платформе для распределенных вычислений Hadoop, реализующей концепцию MapReduce. Применение Hadoop позволило масштабировать решение задачи на сеть из трех вычислительных узлов. Проведенные эксперименты показали, что применение Hadoop удобно для параллельной обработки больших объемов данных, используя при этом обычные компьютеры.

Литература

1. Apache Hadoop project. Режим доступа: http://hadoop.apache.org/
2. Ф. В. Борисюк, В. И. Швецов. Новый метод поиска на основе иерархической кластеризации по областям текстовых документов // Вестник Нижегородского университета им. Н.И. Лобачевского, 2009, № 4, с. 165–171.
3. Jeffrey Dean, Sanjay Ghemawat. MapReduce: simplified data processing on large clusters // Communications of the ACM. 2008. V. 51 P. 107-113.
4. Daniel Kelleher, Saturnino Luz. Automatic Hypertext Keyphrase Detection // Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, Edinburgh, Scotland, UK. 2005. P. 1608-1610.