МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ Н. Э. БАУМАНА

Факультет информатики и систем управления

Кафедра теоретической информатики и компьютерных технологий

Курсовой проект

по курсу «Компьютерные системы и сети»

«Фреймворк и файловая система для распределённой обработки больших данных в рамках концепции map-reduce»

Выполнил:
студент ИУ9-91
Выборнов А. И.

Руководитель:
Дубанов А. В.

Москва 2014

Содержание

Введение

1. Теоретическая часть

1.1. Map-reduce

- Структура (key, value) - пара (ключ, значение).
- Программирование представляет собой определение двух функций:
  - $map : (key, value) \to [(key, value)]$
  - $reduce : (key, [value]) \to [(key, value)]$
- Между стадиями *map* и *reduce* происходит группировка и сортировка данных.

Картинка иллюстрирующая процесс, более подробное описание как он работает.

1.1.1. Зачем нужен map-reduce

- Обработка больших данных (Big Data).
  - Вычисления превосходят возможности одной машины.
  - Данные не помещаются в памяти, необходимо обращаться к диску.
  - Можно хранить много данных, но задержки и пропускная способность оборудования растут пропорционально данным.
- Удобная абстракция для построения алгоритмов обработки больших данных.
- Устойчивость к отказам.

1.1.2. Пример применения map-reduce

- **Задача:** Есть граф пользователей некоторого ресурса, заданный в виде строчек: «пользователь - друг1 друг2 ...». Для каждой пары пользователей найти общих друзей.
- Разберём задачу на следующих входных данных:
  - A - B C D
  - B - A C
  - C - A B D

  - D - A C

- На стадии map преобразовываем пару (пользователь, друзья) в множество пар следующим образом:
  - (A, B C D) -> (A B, B C D), (A C, B C D), (A D, B C D)
  - (B, A C) -> (A B, A C), (B C, A C)
  - (C, A B D) -> (A C, A B D), (B C, A B D), (C D, A B D)
  - (D, A C) -> (A D, A C), (C D, A C)
- Сливаем результаты полученные на стадии map, получаем список пар:
  - (A B, [B C D, A C])
  - (A C, [B C D, A B D])
  - (A D, [B C D, A C])
  - (B C, [A B D, A C])
  - (C D, [A B D, A C])
- На стадии reduce пересекаем с друг другом все элементы списка значений и получаем:
  - (A B, C)
  - (A C, B D)
  - (A D, C)
  - (B C, A)
  - (C D, A)

1.2. Распределённая файловая система

Что такое, зачем требуется для данного проекта

1.2.1. Архитектура распределённой файловой системы

картинка с описанием

1.3. Распределённый map-reduce

Что такое, зачем он нужен map-reduce удобная концепция, но ...

1.3.1. Архитектура распределённого map-reduce

картинка с подробным описанием

1.4. Решаемый класс задач

мат выкладки

2. Объекты и методы

Характеристики программного обеспечения:

- Операционная система — Ubuntu 14.04.1 LTS 64-bit.
- IDE — Syblime Text 2.
- Язык программирования — Python 2.7.3.

Характеристики оборудования:

- Процессор — Intel Core i7-3770k 3.5Ghz×8.
- Оперативная память — 16Gb DDR3.
- Видеокарта — ATI Radeon 7860.

3. Реализация

3.1. Используемые технологии

внешние технологии используемые в проекте сериализация, zmq и прочее Парам пам пам

3.2. Работа с большими данными

какие есть проблемы

3.2.1. python generator

nee

3.2.2. split

бла

3.2.3. dfs

проблемы в dfs как эти проблемы решаются в фс

3.2.4. map-reduce

проблемы в map-reduce как эти проблемы решаются в map-reduce

3.3. Взаимодействие между узлами

Описание реализации взаимодействия между различными узлами сети.

3.3.1. класс nodesmanager

описание класса

3.4. Интерфейс

есть dfs, есть mr

3.4.1. dfs

Distributed file system is required to map-reduce framework.

On each node, you should run *dfsnode.py* with two arguments - port and storagepath. Like this:

python dfsnode.py -p 5556 -s /home/username/storage

Then you should fill *config.json* with information about nodes. Now you can use *dfs.py*. Samples of use dfs.py:

python dfs.py -ls /user/ python dfs.py -mkdir /user/username/userdatafolder python dfs.py -put ./test.in /user/username/userdatafolder/testfile python dfs.py -get /user/username/userdatafolder/testfile python dfs.py -rm /user/username

3.4.2. mr

4. Тестирование

5. Заключение

Список литературы

[1] SMILES — A Simplified Chemical Language // Daylight Chemical Information Systems, Inc: URL: http://www.daylight.com/dayhtml/doc/theory/theory.SMILES.html

[2] Atomic Coordinate Entry Format Description // Penn State University: URL: http://www.wwpdb.org/documentation/format33/v3.3.html

[3] Periodic Table Datan Files // Protein Data Bank: URL: http://php.scripts.psu.edu/djh300/cmpsc221/p3s11-pt-data.htm

[4] Three.js — javascript 3D library // Three.js: URL: http://mrdoob.github.io/three.js/

[5] File: Tubby-1c8z-pymol.png // Wikipedia: URL: http://en.wikipedia.org/wiki/File:Tubby-1c8z-pymol.png

[6] GLmol - Molecular Viewer on WebGL/Javascript // GLmol: URL: http://webglmol.sourceforge.jp/index-en.html