Распределённые объектные технологии: **Модель RDD** и Apache Spark

Д. А. Усталов

УрФУ и ИММ УрО РАН

19 апреля 2016 г.

Концептуальный вопрос

Насколько быстро работают программы, построенные на модели MapReduce?

- Чтение из HDFS перед шагом Map.
- Выполнение шага Map.
- Запись временных файлов после шага Map.
- Чтение временных файлов перед шагом Reduce.
- Выполнение шага Reduce.
- Запись в HDFS после шага Reduce.

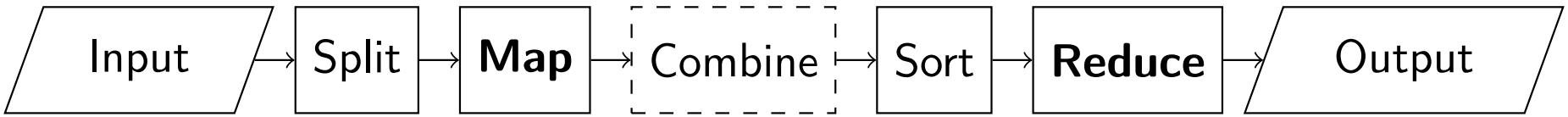
Выполняется *очень много* операций дискового ввода-вывода.

Подходы к решению проблемы

- Использование быстрых систем хранения.
  - Твёрдотельные накопители и другие дорогие решения.
- Сжатие данных перед сохранением.
  - Дополнительная нагрузка на ЦПУ.
- Хранение промежуточных результатов в ОЗУ.
  - Проблема восстановления после сбоев.

Потоковая модель вычислений

Представление вычислительного процесса в виде последовательности операций по обработке исходных данных.



- `output = input.`
  `split(...).`
  `map(mapper).`
  `combine(...).`
  `sort(...).`
  `reduce(reducer)`

Apache Spark

- Фреймворк для построения распределённых приложений.
- Создан в UC Berkeley и не является клоном технологии Yahoo! или Google.
- Написан на Scala, поддерживается Java, Python, R.
- Специализированная модель вычислений в памяти.

Эластичный распределённый набор данных

RDD (resilient distributed dataset) — фрагмент данных, доступный только для чтения.

- Операции над RDD: преобразования и действия.
- Композиция операций задаёт граф родословной, описывающий процесс обработки данных.
- Наличие графа родословной позволяет перезапускать задачи и выполнять их на разных узлах.

Можно писать сложные программы, состоящие из цепочки операторов.

Преобразования наборов данных

Ленивые операции, порождающие новый RDD на основе исходного RDD.

- $map(f : T \Rightarrow U)$
- $filter(f : T \Rightarrow \text{Boolean})$
- $sample(fraction : \text{Float})$
- $union()$, $join()$, $crossProduct()$
- $sort(p : \text{Comparator}[\mathbf{K}])$
- $partitionBy(p : \text{Partitioner}[\mathbf{K}])$
- ...

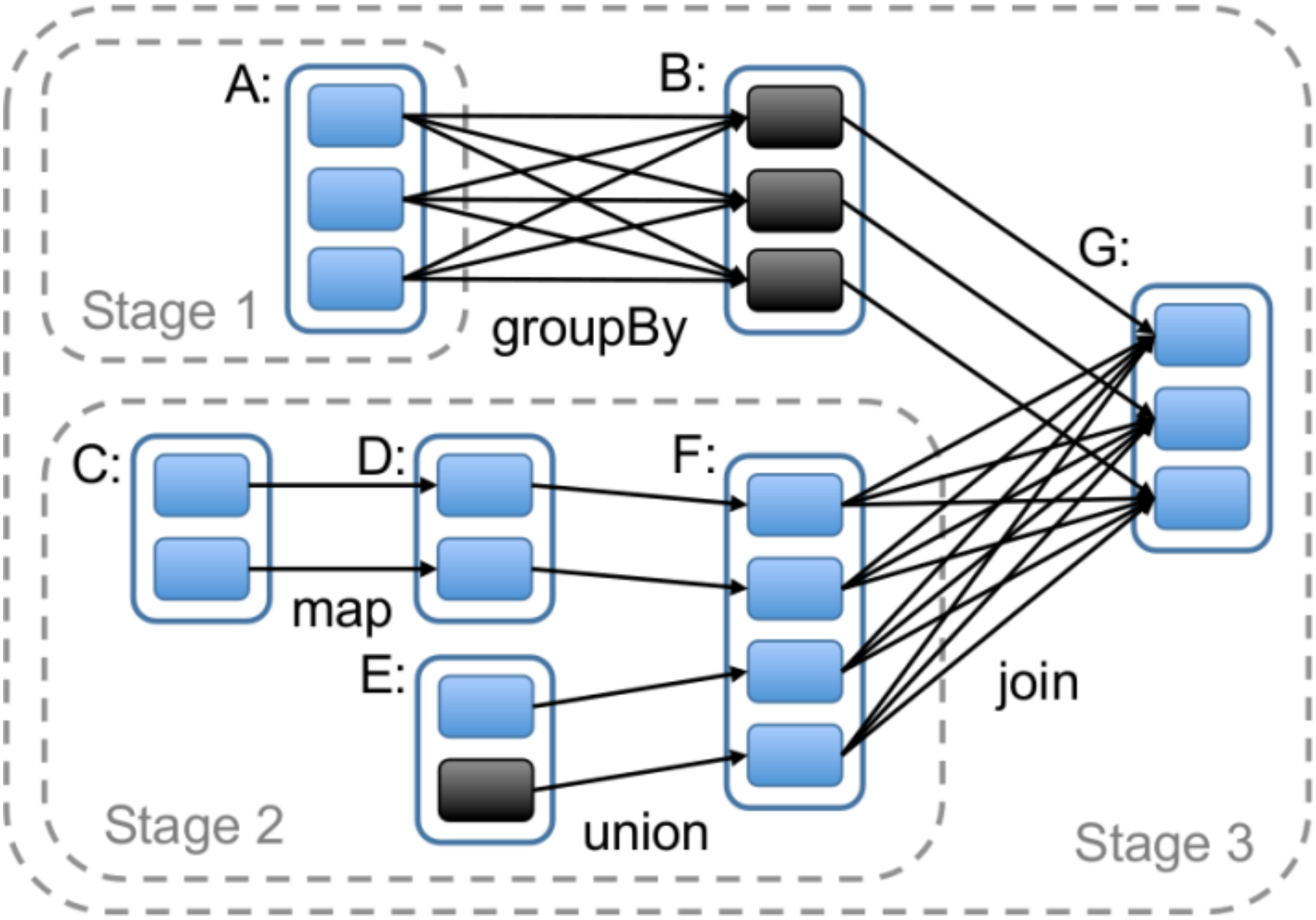
Преобразования не выполняются, пока не вызвано действие.

Действия над наборами данных

Операции, запускающие поток вычислений для получения результата.

- $count()$
- $collect()$
- $reduce(f : (T, T) \Rightarrow T)$
- $lookup(k : K)$
- $save(path : \text{String})$

Выполнение вычислений



Демонстрация работы I

Вычисление числа π методом квази-Монте-Карло (как на прошлом занятии).

- pi.py
- `source /etc/spark/conf/spark-env.sh`
  `spark-submit --deploy-mode client \`
  `  --master yarn pi.py 10`

Задача выполняется на нескольких узлах, о чём свидетельствуют показания `top` и `ps aux` на рабочих узлах.

Демонстрация работы II

Вычисление количества слов в корпусе текстов (как на прошлом занятии).

- wordcount.py
- ```
  source /etc/spark/conf/spark-env.sh
  spark-submit --deploy-mode client \
    --master yarn wordcount.py test.txt
  ```

Файл `test.txt` находится в HDFS, а не в локальной ФС.

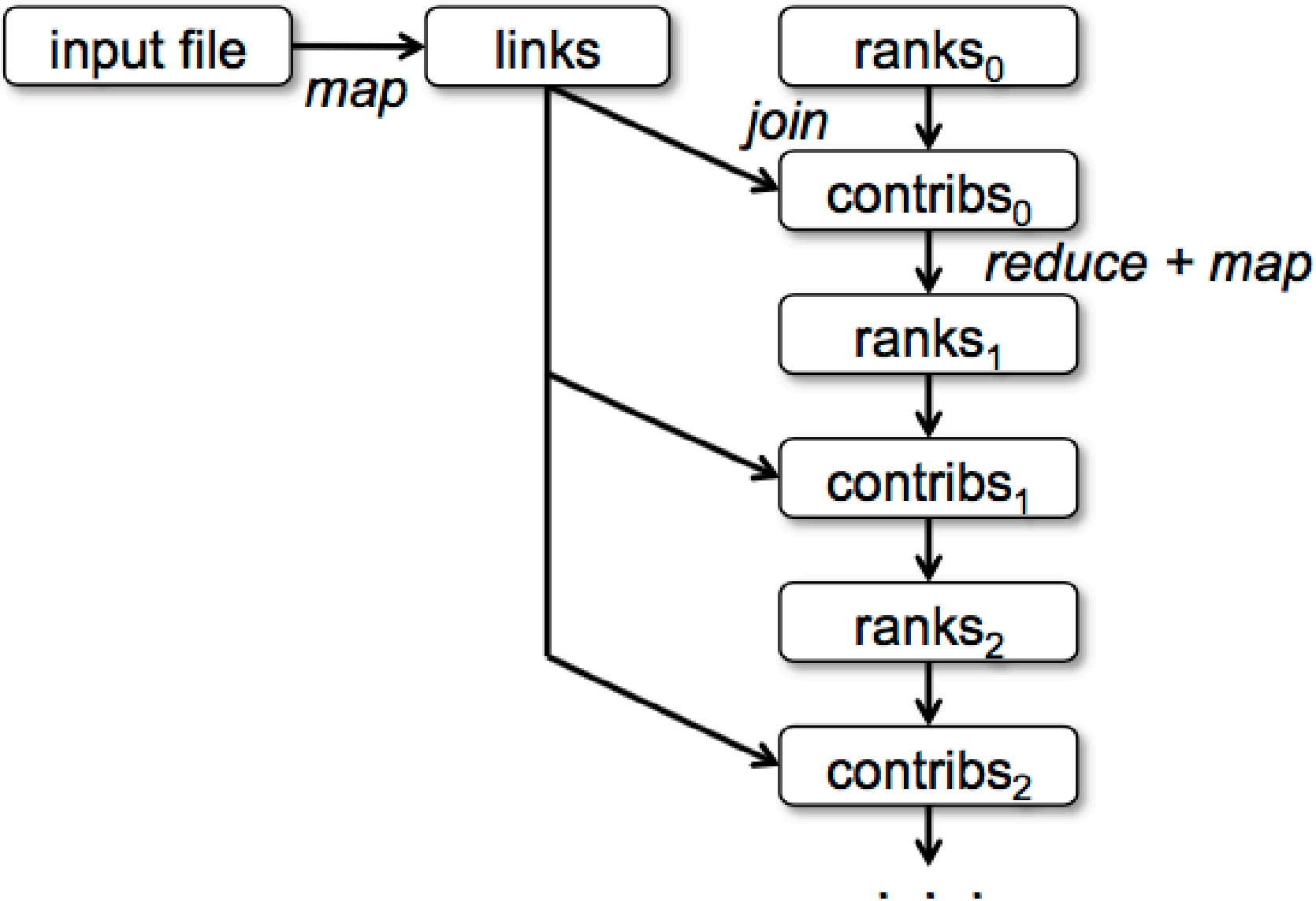
- ```
  hdfs dfs -put test.txt
  hdfs dfs -ls
  hdfs dfs -cat test.txt
  ```

Демонстрация работы III

Взвешивание важности веб-страницы на основе входящих в неё ссылок (и не только).

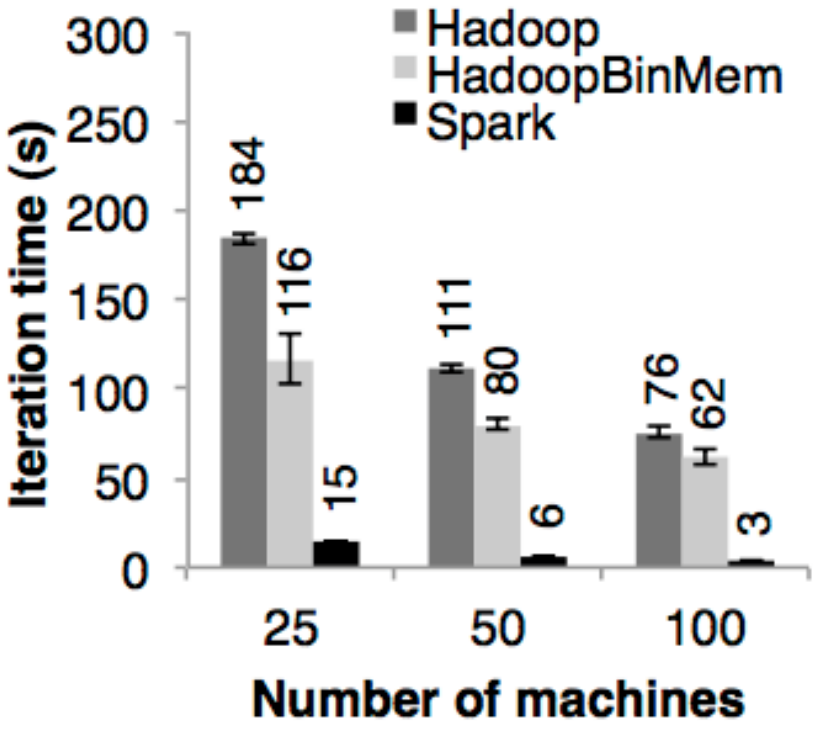
- pagerank.py и followers.txt
- ```
  source /etc/spark/conf/spark-env.sh
  spark-submit --deploy-mode client \
    --master yarn pagerank.py followers.txt 10
  ```
  ```

Демонстрация работы III: граф родословной



Обсуждение

- Интуитивно, RDD — обобщение модели MapReduce с более удобным API.
- Ускорение в десятки и сотни раз по сравнению с MapReduce.
- Неудобно использовать RDD для асинхронной работы с разделяемым состоянием.
- Spark популярен и имеет развитую экосистему.



Экосистема

- Собственные инструменты: Spark SQL, Spark Streaming (не путать с Hadoop Streaming), и др.
- Аналоги технологий Hadoop: Mahout $\approx$ MLlib, Giraph $\approx$ GraphX, и т. д.
- Spark может функционировать как автономно, так и под управлением YARN в кластере Hadoop.
- Дистрибутивы Hadoop включают Spark в стандартную поставку.

Домашнее задание

Разработать поисковую машину на основе Spark с использованием функции ранжирования **BM25**.

- Пользователь вводит запрос Q, система печатает идентификаторы 10 самых релевантных документов.
- $score(d, Q) = \sum_{i=1}^{|Q|} idf(q_i) \cdot \frac{f_d(q_i) \cdot (k_1+1)}{f_d(q_i)+k_1 \cdot (1-b+b \cdot \frac{|d|}{\text{avgdl}})}$, где $f_d(q_i)$ — количество вхождений q_i в документ d, avgdl — средняя длина документа, $k_1 = 1.2$, $b = 0.5$.
- $idf(q_i) = \frac{N-n(q_i)+0.5}{n(q_i)+0.5}$, где N — количество документов, $n(q_i)$ — количество документов, содержащих q_i.
- Тексты из дампа OpenCorpora (см. opencorpora2txt.rb).

Спасибо за внимание!

Вопросы?

Дмитрий Усталов

https://linkedin.com/in/ustalov

http://kvkt.urfuclub.ru/courses/dot/

https://telegram.me/doturfu

dmitry.ustalov@urfu.ru