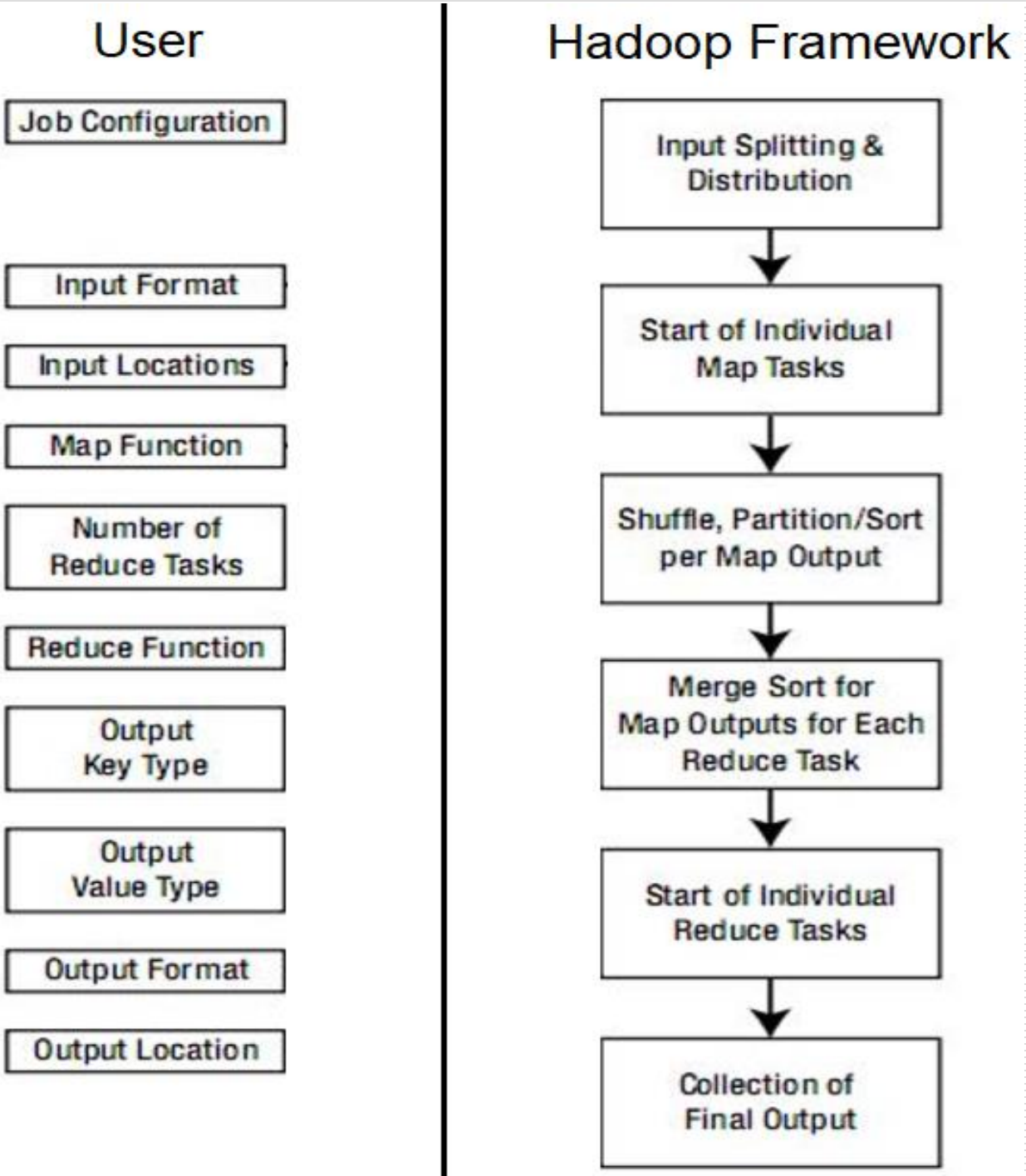
Môi trường lập trình MapReduce

- Phân chia nhiệm vụ
- Các kiểu dữ liệu Hadoop hỗ trợ
- Lớp Mapper
- Lớp Reducer
- Hadoop I/O

Phân chia nhiệm vụ

- Giữa người dùng và framework Hadoop



Người dùng

- Thiết lập thông số cấu hình hệ thống của MapReduce Job
- Định nghĩa hàm map, reduce
- Có thể thiết lập thông tin số lượng reduce task (mặc định do hệ thống thiết lập)
- Truyền vào:
  - Kiểu format, đường dẫn của dữ liệu input
  - Kiểu format cho từng record (key, value) của dữ liệu output của hàm reduce
  - Kiểu format cho dữ liệu output cuối cùng, vị trí mà file output sẽ lưu

Framework Hadoop

- Với thông tin cấu hình, hệ thống thực hiện:
  - Kiểm tra các thông tin này liệu có hợp lệ hay không
  - Thông báo người dùng là ứng dụng có thể bắt đầu (nếu thông tin hợp lệ)
- Dựa vào 2 thông tin về kiểu format (format đọc dữ liệu và format từng record input) và đường dẫn dữ liệu input
  - Hệ thống tính toán và thực hiện việc chia nhỏ dữ liệu input này thành các input split
- Sau khi có tập dữ liệu input split, hệ thống phân tán map task trên các TaskTracker thực hiện
- Thực hiện hàm map trả về từng record với format như người dùng định nghĩa

Framework Hadoop

- Với tập record đầu ra hàm map, hệ thống thực hiện
  - Phân chia chúng vào từng vào từng partition (số lượng partition đúng bằng số lượng reduce task)
  - Sắp xếp theo khóa trong từng partition
- Sau khi tất cả maptask hoàn thành, hệ thống thực hiện
  - Lấy dữ liệu của một partition trên các ouput của maptask
  - Trộn các dữ liệu này lại và sắp xếp để cho ra tập các record (key, danh sách value)
  - Xử lý các record bằng việc chạy hàm reduce task
- Reduce task trả về từng record với kiểu format output cuối cùng đã được người dùng định nghĩa trước
- Reduce task sẽ lưu dữ liệu output đường dẫn của mà file đầu ra đã được người dùng định nghĩa trước

Các kiểu dữ liệu Hadoop hỗ trợ

- BooleanWritable: Lớp bao (wapper) của biến kiểu Boolean chuẩn
- ByteWritable: Lớp bao của biến kiểu byte đơn
- DoubleWritable: Lớp bảo của biến kiểu Double
- FloatWritable: Lớp bao của biến kiểu Float
- IntWritable: Lớp bao của biến kiểu Integer
- LongWritable: Lớp bao của biến kiểu Long
- Text: Lớp bao của biến kiểu văn bản định dạng UTF-8
- NullWritable: Lớp để giữ chỗ khi mà key hoặc value không cần thiết

Lớp Mapper

- Là lớp liên quan đến cài đặt hàm map
- Cài đặt từ interface Mapper và kế thừa từ lớp MapReduceBase
- Lớp MapReduceBase
  - Là lớp cơ sở cho cả mapper và reducer
  - Bao gồm hai phương thức hoạt động hiệu quả
    - void configure(JobConf job): Trong hàm này, người dùng có thể trích xuất các thông số cài đặt từ biến job. Hàm này được gọi trước khi xử lý dữ liệu
    - void close() – Như hành động cuối trước khi chấm dứt nhiệm vụ map, hàm này nên được gọi bất cứ khi nào kết thúc: kết nối cơ sở dữ liệu, các file đang mở

Lớp Mapper

- Sử dụng mẫu Mapper<K1, V1, K2, V2>
  - K1, V1: Kiểu dữ liệu tương ứng của key, value đầu vào
  - K2, V2: Kiểu dữ liệu tương ứng của key, value trung gian
- Sử dụng phương thức map duy nhất để xử lý các cặp (key/value) như sau:
  - void map(K1 key, V1 value, OutputCollector<K2, V2> output, Reporter reporter) throws IOException
  - Tạo ra một danh sách output (có thể rỗng) các cặp (K2, V2) từ một cặp đầu vào (K1, V1)
  - Nội dung hàm cài đặt quá trình mapping dữ liệu đầu vào thành dữ liệu trung gian
  - Reporter cung cấp các tùy chọn để ghi lại thông tin thêm về mapper như tiến triển công việc

Lớp Reducer

- Là lớp liên quan đến cài đặt hàm reduce
- Mở rộng từ lớp MapReduceBase để cho phép cấu hình và dọn dẹp
- Thực hiện ngầm: Sắp xếp và nhóm các giá trị trung gian theo các key trung gian để tạo ra danh sách các cặp (K2 key, Iterator<V2> values) phục vụ cho hàm reduce
- Sử dụng phương thức reduce duy nhất như sau
  - void reduce(K2 key, Iterator<V2> values, OutputCollector<K3,V3> output, Reporter reporter) throws IOException
  - Nội dung hàm cài đặt quá trình reducing dữ liệu trung gian thành dữ liệu đầu ra cuối cùng
  - Sinh ra một danh sách (có thể rỗng) các cặp (K3, V3)

Hadoop I/O

- Dữ liệu đầu vào thường là các tập tin lớn, có thể đến hàng chục hoặc hàng trăm gigabyte và cũng có thể nhiều hơn
- Một trong những nguyên tắc cơ bản của xử lý MapReduce là sự phân tách dữ liệu đầu vào thành các khối (chucks)
- MapReduce có thể xử lý các khối này một cách song song sử dụng nhiều máy tính khác nhau
- Trong Hadoop thuật ngữ "khối" này được gọi là "input splits"
- Việc quản lý truy cập và phân chia các khối được thực hiện bởi hệ thống file HDFS của Hadoop
- Định dạng file mà HDFS hỗ trợ:
  - InputFormat
  - OutputFormat

InputFormat

- Mỗi record mô tả của một cặp key/value
- TextInputFormat
  - Key: LongWritable, Value: Text
  - Mỗi dòng trong file text là một record
  - Key là một byte offset của dòng, Value là nội dung của dòng đó
- KeyValueTextInputFormat:
  - Key: LongWritable, Value: Text
  - Mỗi dòng trong file text là một record. Kí tự phân chia đầu tiên trên mỗi dòng
  - Tất cả mọi thứ đứng trước kí tự phân chia là key và đứng sau là value
  - Kí tự phân chia (mặc định là dấu tab (\t)) được thiết lập bởi thuộc tính key.value.separator.input.line với lệnh: conf.set("key.value.separator.in.input.line", ",");

Ví dụ InputFormat

- TextInputFormat phù hợp các tập tin đầu vào có dữ liệu đơn giản, không cấu trúc như file văn bản chỉ chứa từ (bài toán đếm từ)
- KeyValueTextInputFormat được sử dụng trong các tập tin đầu vào có cấu trúc hơn
  - 17:16:18 http://hadoop.apache.org/core/docs/r0.19.0/api/index.html
    17:16:19 http://hadoop.apache.org/core/docs/r0.19.0/mapred_tutorial.html
    17:16:20 http://wiki.apache.org/hadoop/GettingStartedWithHadoop
    17:16:20 http://www.maxim.com/hotties/2008/finalist_gallery.aspx
    17:16:25 http://wiki.apache.org/hadoop/

InputFormat

- SequenceFileInputFormat<K,V>:
  - Để đọc các file tuần tự (sequence files)
  - Key và value được người sử dụng định nghĩa
  - Sequence file là một dạng file nhị phân nén đặc biệt của hadoop
  - Nó tối ưu cho truyền dữ liệu giữa đầu ra của một MapReduce job tới đầu vào của một MapReduce job khác.
- NlineInputFormat:
  - Giống như TextInputFormat, nhưng mỗi split được đảm bảo phải có chính xác N dòng.
  - Thuộc tính mapred.line.input.format.linespermap (mặc định là 1) để thiết lập N
  - Key: LongWritable, Value: Text

OutputFormat

- Dữ liệu đầu ra của MapReduce lưu vào trong các file sử dụng lớp OutputFormat, tương tự lớp InputFormat
- Các định dạng đầu ra: TextOutputFormat, KeyValueTextOutputFormat, SequenceFileOutputFormat
- Sử dụng lệnh setOutputFormat để thiết lập định dạng đầu ra
- Đầu ra thì không split, mỗi reducer ghi một output riêng
- Sử dụng lệnh *setOutputPath* thiết lập thư mục đầu ra
- Các tập tin đầu ra nằm trong thư mục đầu ra và tên là part-nnnnn, với nnnnn là ID vùng của reducer