第十四讲　Flink编程

徐 辰
cxu@dase.ecnu.edu.cn
華東師範大學

大纲

2

- 编程方式
  - Scala Shell
  - Java/Scala IDE
- 编程接口
- 编程实例

Scala Shell

3

大纲

4

- 编程方式
  - Scala Shell
  - Java/Scala IDE
- 编程接口
- 编程实例

应用程序编写流程

5

- 在IDE中编写Java/Scala程序
- 利用IDE打包jar
- 命令行提交jar包

大纲

6

- 编程方式
- 编程接口
- 编程实例

Flink API

7

SQL — High-level Language

Table API — Declarative DSL

DataStream / DataSet API — Core APIs

Stateful Stream Processing — Low-level building block (streams, state, [event] time)

Programming Sketch

8

```
StreamExecutionEnvironment env =
StreamExecutionEnvironment.getExecutionEnvironment();

//设置并行度
env.setParallelism(1);
…

env.execute();
```

大纲

9

- 编程方式
- 编程接口
  - DataStream
  - DataSet
- 编程实例

DataStream API

10

	API
DataSource	● readTextFile() ● socketTextStream() ● …
DataStream Transformation	● map() ● keyBy() ● aggregation() ● window() ● coGroup() ● …
DataSink	● writeAsText() ● writeAsCsv() ● print() ● writeToSocket() ● …

DataStream API

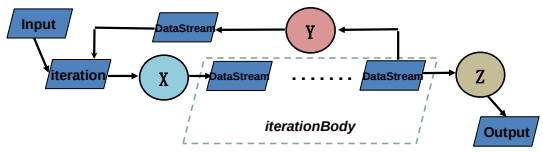
11

```
val lines: DataStream[String] = env.addSource(
                                    new FlinkKafkaConsumer09<>(…))

val events: DataStream[Event] = lines.map((line) => parse(line))

val stats: DataStream[Statistic] = stream
   .keyBy("sensor")
   .timeWindow(Time.seconds(5))
   .sum(new MyAggregationFunction())

stats.addSink(new RollingSink(path))
```

DataStream Iteration

12

```
StreamExecutionEnvironment env = StreamExecutionEnvironment.getExecutionEnvironment();

IterativeStream<> iteration = input.iterate();
DataStream<> iterationBody = iteration.X(/* this is executed many times */);
…
iteration.closeWith(iterationBody.Y(/* one part of the stream */));
DataStream<> output = iterationBody.Z(/* some other part of the stream */);

env.execute();
```



大纲

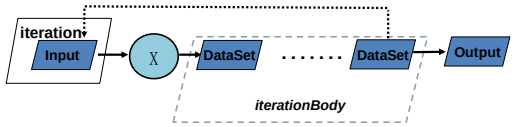
13

- 编程方式
- 编程接口
  - DataStream
  - DataSet
- 编程实例

DataSet Iteration

14

```
ExecutionEnvironment env = ExecutionEnvironment.getExecutionEnvironment();

IterativeDataSet<> iteration = input.iterate();
DataSet<> iterationBody = iteration.X(/* this is executed many times */);
...
DataSet<> output = iteration.closeWith(iterationBody) // Iteratively transform the IterativeDataSet;

env.execute();
```



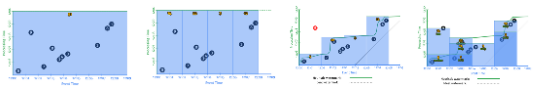
大纲

15

- 编程方式
- 编程接口
- 编程实例
  - WWWH
  - PageRank
  - Stateful WordCount

What Where When How

16



1 Classic Batch

2 Windowed Batch

3 Streaming

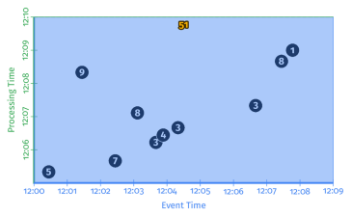
4 Streaming + Accumulation

For more information see https://cloud.google.com/dataflow/examples/gaming-example

Classic Batch

17

```
PCollection<KV<String, Integer>> scores = input
    .apply(Sum.integersPerKey());
```



Classic Batch

18

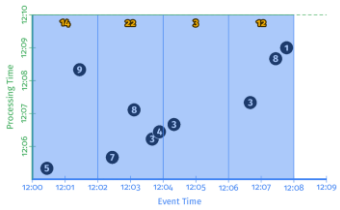
```
DataStream<Tuple2<String,Integer>> source =env.addSource(new Producer4Window(false));      //不使用watermarker

DataStream<Tuple2<String,Integer>> sink =source
        .map(new MapFunction<Tuple2<String, Integer>, Tuple2<String, Integer>>() {
            …})
        .keyBy(0)
        .sum(1);
```

https://github.com/dasebigdata-ecnu/BigDataSystems_Example/blob/master/flink/src/main/java/wwwh/ClassicBatch.java

Windowed Batch

19

```
PCollection<KV<String, Integer>> scores = input
    .apply(Window.into(FixedWindows.of(Duration.standardMinutes(2))))
    .apply(Sum.integersPerKey());
```



Windowed Batch

20

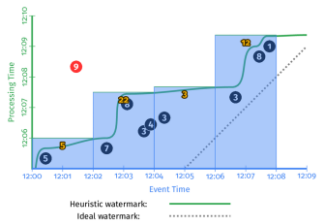
```
DataStream<Tuple2<String,Integer>> source =env.addSource(new
Producer4Window(false));      //不使用watermarker

DataStream<String> sink =source
//放入基于事件时间的滚动窗口
.windowAll(TumblingEventTimeWindows.of(Time.seconds(120L)))
.apply(new AllWindowFunction<Tuple2<String, Integer>, String, TimeWindow>() {
            @Override
            public void apply(…){…}
}
```

https://github.com/dasebigdata-ecnu/BigDataSystems_Example/blob/master/flink/src/main/java/wwwh/WindowedBatch.java

Streaming

21

```
PCollection<KV<String, Integer>> scores = input
    .apply(Window.into(FixedWindows.of(Duration.standardMinutes(2))
          .triggering(AtWatermark()))
    .apply(Sum.integersPerKey());
```



Streaming

22

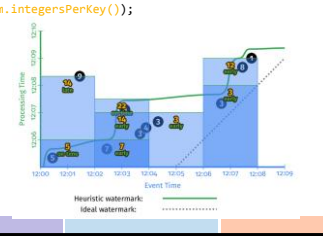
```
DataStream<Tuple2<String,Integer>> source =env.addSource(new
Producer4Window(true));      //使用watermarker

DataStream<String> sink =source
//放入基于事件时间的滚动窗口
.windowAll(TumblingEventTimeWindows.of(Time.seconds(120L)))
.apply(new AllWindowFunction<Tuple2<String, Integer>, String, TimeWindow>() {
            @Override
            public void apply(…){…}
}
```

https://github.com/dasebigdata-ecnu/BigDataSystems_Example/blob/master/flink/src/main/java/wwwh/Streaming.java

Streaming Accumulation

23

```
PCollection<KV<String, Integer>> scores = input
    .apply(Window.into(FixedWindows.of(Duration.standardMinutes(2))
          .triggering(AtWatermark()
            .withEarlyFirings(AtPeriod(Duration.standardMinutes(1)))
            .withLateFirings(AtCount(1)))
          .accumulatingFiredPanes())
    .apply(Sum.integersPerKey());
```



Streaming Accumulation

24

```
DataStream<Tuple2<String,Integer>> source =env.addSource(new
Producer4Window(true));      //使用watermarker

DataStream<String> sink =source
//放入基于事件时间的滚动窗口
.windowAll(TumblingEventTimeWindows.of(Time.seconds(120L)))
  //自定义trigger，每秒触发，watermark 触发，延迟数据到达触发
.trigger(CustomerTrigger.<TimeWindow>of(Time.seconds(60L)))
  //允许延迟最大时间
.allowedLateness(Time.seconds(300L))
.apply(new AllWindowFunction<Tuple2<String, Integer>, String, TimeWindow>() {
            @Override
            public void apply(…){…}
}
```

https://github.com/dasebigdata-ecnu/BigDataSystems_Example/blob/master/flink/src/main/java/wwwh/StreamingWithAccumulation.java

大纲

25

- 编程方式
- 编程接口
- 编程实例
  - WWWH
  - PageRank
  - Stateful WordCount

PageRank

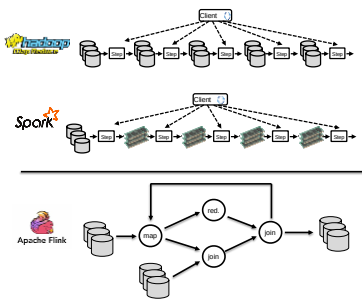
26

```
IterativeDataSet<Tuple2<Long, Double>> iteration =
pagesWithRanks.iterate(maxIterations);

DataSet<Tuple2<Long, Double>> newRanks = iteration
            // join pages with outgoing edges and distribute rank
            .join(adjacencyListInput).where(0).equalTo(0)
            .flatMap(new JoinVertexWithEdgesMatch())
            // collect and sum ranks
            .groupBy(0).aggregate(SUM, 1)
            // apply dampening factor
            .map(new Dampener(DAMPENING_FACTOR, numPages));

DataSet<Tuple2<Long, Double>> finalPageRanks = iteration.closeWith(
                        newRanks,
                        newRanks.join(iteration).where(0).equalTo(0)
                        // termination condition
                        .filter(new EpsilonFilter()));
```

https://github.com/apache/flink/blob/master/flink-examples/flink-examples-batch/src/main/java/org/apache/flink/examples/java/graph/PageRank.java

迭代比较

27



每一迭代步骤(Step)结束时将结果写入HDFS，下一步将该结果再次从HDFS读出迭代次数决定了MapReduce作业的个数

每一迭代步骤 (Step)结束时生成一个RDD，下一步使用RDD再次计算，整个迭代是一个作业

系统提供迭代算子，用户编程时不需要使用for、while等循环，整个迭代是一个作业

大纲

28

- 编程方式
- 编程接口
- 编程实例
  - WWWH
  - PageRank
  - WordCount with State

Stateless flatMap

29

```
DataStream<Tuple2<String, Integer>> counts = text
            // split up the lines in pairs (2-tuples) containing: (word,1)
            .flatMap(new Tokenizer())
            // group by the tuple field "0" and sum up tuple field "1"
            .keyBy(0)
            .sum(1);
```

https://github.com/apache/flink/blob/master/flink-examples/flink-examples-streaming/src/main/java/org/apache/flink/streaming/examples/wordcount/WordCount.java

Stateful flatMap

30

```
DataStream<WordCount> counts = text
            // split up the lines in pairs (2-tuples) containing: (word,1)
            .flatMap(new Tokenizer())
            // group by the tuple field "0" and sum up tuple field "1"
            .returns(WordCount.class)
            .keyBy("word")
            .flatMap(new WordCountFunction());
```

```
class WordCountFunction extends RichFlatMapFunction<WordCount, WordCount> {

 private transient ValueState<WordCountState> countState;

@Override
public void flatMap(WordCount in, Collector<WordCount> out) throws Exception {
            WordCountState lastState = countState.value();
            ……
 }
```

https://github.com/dasebigdata-ecnu/BigDataSystems_Example/blob/master/flink/src/main/java/wordcount/MainWithDefinedState.java

本讲小节

31

- 编程方式
- 编程接口
- 编程实例

谢谢！Q&A

Credits

32

- Some slides from the presenter
  - Frances Perry
  - Tyler Akidau