**Apache Flink**

|  |  |
| --- | --- |
| **审核人** |  |
| **重要性** | 中 |
| **紧迫性** | 中 |
| **拟制人** | 张包峰 |
| **提交日期** |  |

**作者：张包峰**

(版权所有,翻版必究)

**修改记录**

|  |  |  |
| --- | --- | --- |
| **更新时间** | **变更内容** | **变更理由** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

[**作者：张包峰** 1](#_Toc400794574)

[1.1 Overview 4](#_Toc400794575)

[1.2 DataSet数据集表示 4](#_Toc400794576)

[1.3 Functions函数表示 5](#_Toc400794577)

[1.4 Aggregators聚合器 6](#_Toc400794578)

[1.5 Operators 6](#_Toc400794579)

[1.6 Accumulators累加器 6](#_Toc400794580)

[1.7 Broadcast 6](#_Toc400794581)

[1.8 Distributed Cache 7](#_Toc400794582)

[1.9 内存管理 7](#_Toc400794583)

[1.10 Record及类型 7](#_Toc400794584)

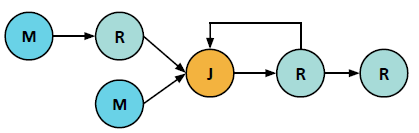
[1.11 Visitor访问者模式 7](#_Toc400794585)

[1.12 例子 7](#_Toc400794586)

## Overview

前身是Stratosphere，介绍可以参考[stratosphere-at-hadoop-summit-2014](http://www.slideshare.net/StephanEwen/stratosphere-at-hadoop-summit-2014)。

Stratosphere的DAG，含迭代



J指的是Join操作

## DataSet数据集表示

DataSet的主要计算API：

map, mapPartition, flatMap, project, aggregator, join, corss, coGroup, union, reduce, reduceGroup, groupBy, min, max, sum, distinct, iterate, iterateDelta

应该说，除了最后iterate的两个api外，其他api基本上和RDD是一样的。

iterate的使用:

DataSet<Double> input = ...;

DataSet<Double> startOfIteration = input.iterate(10);

DataSet<Double> toBeFedBack = startOfIteration

.map(new MyMapper())

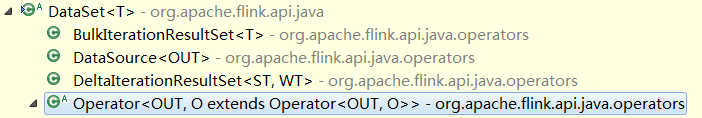
.groupBy(...).reduceGroup(new MyReducer());

DataSet<Double> result = startOfIteration.closeWith(toBeFedBack);

iterateDelta的使用…

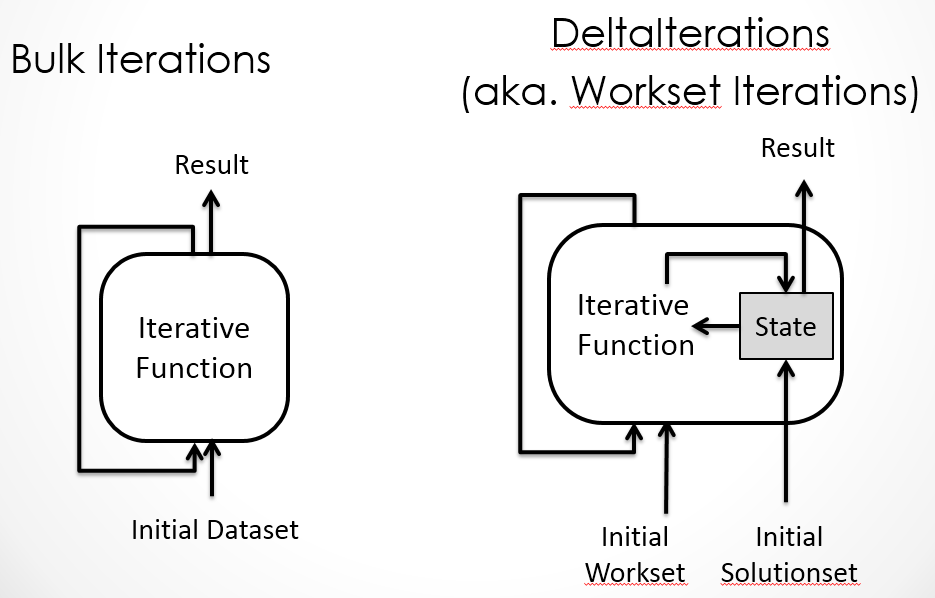
似乎是增量迭代的语义。

DataSet的子类体系



Bulk和Delta寓意批量计算和迭代计算。

最后一类Operator体系在[Operators](#_Operators)一节里有提及。



## Functions函数表示

org.apache.flink.api.common.functions包下。

Function体系，操作对象是DataSet。

简单的map，flatMap，filter，mapPartition(同RDD)

combine，本地聚合，即Iterable<T> -> T，对sub-group可见。

特别地，变种flatCombine，对每个group输出多份T。

cross，为每一对数据，进行一次笛卡尔映射，

set1.cross(set2).with(new CrossFunction());

cogroup，

set1.coGroup(set2).where(<key-definition>).equalTo(<key-definition>)

.with(new MyCoGroupFunction());

join，默认语义是SQL里的inner join，

set1.join(set2).where(<key-definition>).equalTo(<key-definition>)

.with(new MyJoinFunction());

reduce，

input.groupBy(<key-definition>).reduce(new MyReduceFunction());

groupReduce，

input.groupBy(<key-definition>).reduceGroup(

new MyGroupReduceFunction());

除了基本的Function体系，还有一套RichFunction体系，额外提供open()，close()和获得runtime context的方法，也实现了上述一系列基本操作。这种设计方式可以学习。

**关于Function与RuntimeContext**

让function能得到上下文的意义在于，获得一些重要信息，如task并行度、distributed cache，或做accumulator、broadcast。这个点似乎可以参考。

## Aggregators聚合器

org.apache.flink.api.common.aggregators包里。

public enum的方式值得学习下

## Operators

org.apache.flink.api.java.operators包下。

对应Functions体系里的操作。

JoinOperator，一系列实现子类

JoinHint提供的几种策略：

优化器决定；广播并hash前者/后者(适用于小很多)；重分区并hash前者/后者(适用于稍小)；常规的重分区、sort、merge。

## Accumulators累加器

org.apache.flink.api.common.accumulators包里。

累加器来源于Hadoop counters，数据在UDF里被写入，在job结束的时候在系统层面做merge，计算结果可以被client在job结束的时候读取到。

为离散数据设计的Histogram直方图累加器，用一个Integer->Integer的treeMap存。

## Broadcast

## Distributed Cache

## 内存管理

org.apache.flink.core.memory包下

## DAG

## Streaming

抽象为DataStream

单独有流的项目，以及流的api，所以本质上也是区分开流和批的

## Spargel

## 例子