# 背景

参考：

<https://developer.aliyun.com/article/724685>

# 概述

参考：

[一文读懂Apache Flink技术](https://juejin.cn/post/6844903700964589576)

## Flink vs Spark

参考：

[Flink与Spark的比较与区别](https://juejin.cn/post/7327353198867021851)

[Apache Flink vs Apache Spark：数据处理的详细比较](https://cloud.tencent.com/developer/article/2329145)

Spark 和 Flink 是目前最主流的大数据计算引擎，它们在设计理念、处理模型、运行机制等方面有很多区别，但也有一定的联系。下面我们从多个维度系统地对比它们的异同点。

1、背景与联系



* 联系点：

都是开源 Apache 项目，支持分布式大规模数据处理。

都支持批处理 + 流处理的统一编程模型。

都支持SQL、机器学习、图计算。

都有强大的状态管理和容错机制（Checkpoint / Savepoint）。

2、核心差异对比



3、使用场景对比



4、编程模型示意图对比

1）Spark 的微批处理模型：

流数据 →

[micro-batch 1] → [micro-batch 2] → [micro-batch 3]

每一批次为一小段时间的数据，统一处理。

2）Flink 的原生流处理模型：

流数据 →

→ 事件1 → 事件2 → 事件3 → 事件4

每个事件单独处理，低延迟、高精度。

5、总结

Spark：批处理强、生态丰富、适合大规模离线数据处理和轻量级流处理。

Flink：原生为流而生，低延迟、状态强、适用于高实时性、事件驱动的场景。

建议选择指南：



## 特点

可以看出有状态流计算将会逐步成为企业作为构建数据平台的架构模式，而目前从社区来看，能够满足的只有Apache Flink。Flink通过实现Google Dataflow流式计算模型实现了高吞吐、低延迟、高性能兼具实时流式计算框架。同时Flink支持高度容错的状态管理，防止状态在计算过程中因为系统异常而出现丢失，Flink周期性地通过分布式快照技术Checkpoints实现状态的持久化维护，使得即使在系统停机或者异常的情况下都能计算出正确的结果。

Flink具有先进的架构理念、诸多的优秀特性，以及完善的编程接口，而Flink也在每一次的Release版本中，不断推出新的特性，例如Queryable State功能的提出，容许用户通过远程的方式直接获取流式计算任务的状态信息，数据不需要落地数据库就能直接从Flink流式应用中查询。对于实时交互式的查询业务可以直接从Flink的状态中查询最新的结果。在未来，Flink将不仅作为实时流式处理的框架，更多的可能会成为一套实时的状态存储引擎，让更多的用户从有状态计算的技术中获益。

Flink的具体优势有以下几点。

1、同时支持高吞吐、低延迟、高性能

Flink是目前开源社区中唯一一套集高吞吐、低延迟、高性能三者于一身的分布式流式数据处理框架。像Apache Spark也只能兼顾高吞吐和高性能特性，主要因为在Spark Streaming流式计算中无法做到低延迟保障；而流式计算框架Apache Storm只能支持低延迟和高性能特性，但是无法满足高吞吐的要求。而满足高吞吐、低延迟、高性能这三个目标对分布式流式计算框架来说是非常重要的。

2、支持事件时间（Event Time）概念

在流式计算领域中，窗口计算的地位举足轻重，但目前大多数框架窗口计算采用的都是系统时间（Process Time），也是事件传输到计算框架处理时，系统主机的当前时间。Flink能够支持基于事件时间（Event Time）语义进行窗口计算，也就是使用事件产生的时间，这种基于事件驱动的机制使得事件即使乱序到达，流系统也能够计算出精确的结果，保持了事件原本产生时的时序性，尽可能避免网络传输或硬件系统的影响。

3、支持有状态计算

Flink在1.4版本中实现了状态管理，所谓状态就是在流式计算过程中将算子的中间结果数据保存在内存或者文件系统中，等下一个事件进入算子后可以从之前的状态中获取中间结果中计算当前的结果，从而无须每次都基于全部的原始数据来统计结果，这种方式极大地提升了系统的性能，并降低了数据计算过程的资源消耗。对于数据量大且运算逻辑非常复杂的流式计算场景，有状态计算发挥了非常重要的作用。

4、支持高度灵活的窗口（Window）操作

在流处理应用中，数据是连续不断的，需要通过窗口的方式对流数据进行一定范围的聚合计算，例如统计在过去的1分钟内有多少用户点击某一网页，在这种情况下，我们必须定义一个窗口，用来收集最近一分钟内的数据，并对这个窗口内的数据进行再计算。Flink将窗口划分为基于Time、Count、Session，以及Data-driven等类型的窗口操作，窗口可以用灵活的触发条件定制化来达到对复杂的流传输模式的支持，用户可以定义不同的窗口触发机制来满足不同的需求。

5、基于轻量级分布式快照（Snapshot）实现的容错

Flink能够分布式运行在上千个节点上，将一个大型计算任务的流程拆解成小的计算过程，然后将tesk分布到并行节点上进行处理。在任务执行过程中，能够自动发现事件处理过程中的错误而导致数据不一致的问题，比如：节点宕机、网路传输问题，或是由于用户因为升级或修复问题而导致计算服务重启等。在这些情况下，通过基于分布式快照技术的Checkpoints，将执行过程中的状态信息进行持久化存储，一旦任务出现异常停止，Flink就能够从Checkpoints中进行任务的自动恢复，以确保数据在处理过程中的一致性。

6、基于JVM实现独立的内存管理

内存管理是所有计算框架需要重点考虑的部分，尤其对于计算量比较大的计算场景，数据在内存中该如何进行管理显得至关重要。针对内存管理，Flink实现了自身管理内存的机制，尽可能减少JVM GC对系统的影响。另外，Flink通过序列化/反序列化方法将所有的数据对象转换成二进制在内存中存储，降低数据存储的大小的同时，能够更加有效地对内存空间进行利用，降低GC带来的性能下降或任务异常的风险，因此Flink较其他分布式处理的框架会显得更加稳定，不会因为JVM GC等问题而影响整个应用的运行。

7、Save Points（保存点）

对于7\*24小时运行的流式应用，数据源源不断地接入，在一段时间内应用的终止有可能导致数据的丢失或者计算结果的不准确，例如进行集群版本的升级、停机运维操作等操作。值得一提的是，Flink通过Save Points技术将任务执行的快照保存在存储介质上，当任务重启的时候可以直接从事先保存的Save Points恢复原有的计算状态，使得任务继续按照停机之前的状态运行，Save Points技术可以让用户更好地管理和运维实时流式应用。

## 应用场景

在实际生产的过程中，大量数据在不断地产生，例如金融交易数据、互联网订单数据、GPS定位数据、传感器信号、移动终端产生的数据、通信信号数据等，以及我们熟悉的网络流量监控、服务器产生的日志数据，这些数据最大的共同点就是实时从不同的数据源中产生，然后再传输到下游的分析系统。针对这些数据类型主要包括实时智能推荐、复杂事件处理、实时欺诈检测、实时数仓与ETL类型、流数据分析类型、实时报表类型等实时业务场景，而Flink对于这些类型的场景都有着非常好的支持。

1、实时智能推荐

智能推荐会根据用户历史的购买行为，通过推荐算法训练模型，预测用户未来可能会购买的物品。对个人来说，推荐系统起着信息过滤的作用，对Web/App服务端来说，推荐系统起着满足用户个性化需求，提升用户满意度的作用。推荐系统本身也在飞速发展，除了算法越来越完善，对时延的要求也越来越苛刻和实时化。利用Flink流计算帮助用户构建更加实时的智能推荐系统，对用户行为指标进行实时计算，对模型进行实时更新，对用户指标进行实时预测，并将预测的信息推送给Wep/App端，帮助用户获取想要的商品信息，另一方面也帮助企业提升销售额，创造更大的商业价值。

2、复杂事件处理

对于复杂事件处理，比较常见的案例主要集中于工业领域，例如对车载传感器、机械设备等实时故障检测，这些业务类型通常数据量都非常大，且对数据处理的时效性要求非常高。通过利用Flink提供的CEP（复杂事件处理）进行事件模式的抽取，同时应用Flink的Sql进行事件数据的转换，在流式系统中构建实时规则引擎，一旦事件触发报警规则，便立即将告警结果传输至下游通知系统，从而实现对设备故障快速预警监测，车辆状态监控等目的。

3、实时欺诈检测

在金融领域的业务中，常常出现各种类型的欺诈行为，例如信用卡欺诈、信贷申请欺诈等，而如何保证用户和公司的资金安全，是来近年来许多金融公司及银行共同面对的挑战。随着不法分子欺诈手段的不断升级，传统的反欺诈手段已经不足以解决目前所面临的问题。以往可能需要几个小时才能通过交易数据计算出用户的行为指标，然后通过规则判别出具有欺诈行为嫌疑的用户，再进行案件调查处理，在这种情况下资金可能早已被不法分子转移，从而给企业和用户造成大量的经济损失。而运用Flink流式计算技术能够在毫秒内就完成对欺诈判断行为指标的计算，然后实时对交易流水进行规则判断或者模型预测，这样一旦检测出交易中存在欺诈嫌疑，则直接对交易进行实时拦截，避免因为处理不及时而导致的经济损失。

4、实时数仓与ETL

结合离线数仓，通过利用流计算诸多优势和SQL灵活的加工能力，对流式数据进行实时清洗、归并、结构化处理，为离线数仓进行补充和优化。另一方面结合实时数据ETL处理能力，利用有状态流式计算技术，可以尽可能降低企业由于在离线数据计算过程中调度逻辑的复杂度，高效快速地处理企业需要的统计结果，帮助企业更好地应用实时数据所分析出来的结果。

5、流数据分析

实时计算各类数据指标，并利用实时结果及时调整在线系统相关策略，在各类内容投放、无线智能推送领域有大量的应用。流式计算技术将数据分析场景实时化，帮助企业做到实时化分析Web应用或者App应用的各项指标，包括App版本分布情况、Crash检测和分布等，同时提供多维度用户行为分析，支持日志自主分析，助力开发者实现基于大数据技术的精细化运营、提升产品质量和体验、增强用户黏性。

6、实时报表分析

实时报表分析是近年来很多公司采用的报表统计方案之一，其中最主要的应用便是实时大屏展示。利用流式计算实时得出的结果直接被推送到前端应用，实时显示出重要指标的变换情况。最典型的案例便是淘宝的双十一活动，每年双十一购物节，除疯狂购物外，最引人注目的就是天猫双十一大屏不停跳跃的成交总额。在整个计算链路中包括从天猫交易下单购买到数据采集、数据计算、数据校验，最终落到双十一大屏上展现的全链路时间压缩在5秒以内，顶峰计算性能高达数三十万笔订单/秒，通过多条链路流计算备份确保万无一失。而在其他行业，企业也在构建自己的实时报表系统，让企业能够依托于自身的业务数据，快速提取出更多的数据价值，从而更好地服务于企业运行过程中。

# 架构

# 原理

参考：

[Flink 原理与实现：深入理解Flink核心技术](https://juejin.cn/post/6844903760762765326)

<http://xueai8.com/course/73/article>

[Flink原理、架构与实现Part2 - 原理与架构](https://demonyangyue.github.io/2020/05/20/flink-2020-05-20-flink2/)

# 数据分布