什么是实时流计算

所谓实时流计算，就是近几年由于数据得到广泛应用之后，在数据持久性建模不满足现状的情况下，急需数据流的瞬时建模或者计算处理。在这种实时计算的应用实例有金融服务、网络监控、电信数据管理、Web应用、生产制造、传感检测，等等。在这种数据流模型中，单独的数据单元可能是相关的元组（Tuple），如网络测量、呼叫记录、网页访问等产生的数据。但是，这些数据以大量、快速、时变（可能是不可预知）的数据流持续到达，由此产生了一些基础性的新的研究问题 —— 实时计算。实时计算的一个重要方向就是实时流计算。

1. 实时流计算背景

数据的价值随着时间的流逝而降低，所以事件出现后必须尽快对它们进行处理，最好事件出现时便立刻对其进行处理，发生一个事件进行一次处理，而不是缓存起来成一批处理。例如商用搜索引擎，像Google、Bing和Yahoo!等，通常在用户查询响应中提供结构化的Web结果，同时也插入基于流量的点击付费模式的文本广告。为了在页面上的最佳位置展现最相关的广告，通过一些算法来动态估算给定上下文中一个广告被点击的可能性。上下文可能包括用户偏好、地理位置、历史查询、历史点击等信息。一个主搜索引擎可能每秒钟处理成千上万次查询，每个页面都可能会包含多个广告。为了几十处理用户反馈，需要一个低延迟、可扩展、高可靠的处理引擎。

对于这些实时性要求很高的应用，若把持续到达的数据简单地放到传统数据库管理系统DBMS中，并在其中进行操作，是不切实际的。传统的DBMS并不是为快速连续地存放单个的数据单元而设计的，而且也不支持“持续处理”，而“持续处理”是数据流应用的典型特征。另外，现在人们都认识到，“近似性”和“自适应性”是对数据流进行快速查询和其它处理（如数据分析和数据采集）的关键要素，而传统DBMS的主要目标恰恰与之相反：通过稳定的查询设计，得到精确的答案。

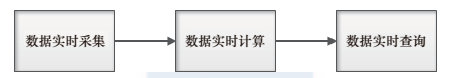
另外一些方案是采用MapReduce来处理实时数据流。但是，尽管MapReduce做了实时性改进，也很难稳定地满足应用需求。这是因为Hadoop MapReduce框架为批处理做了高度优化，典型的是通过调度批量任务来操作静态数据，任务不是常驻服务，数据也不是实时流入；而数据流计算的典型范式之一是不确定数据速率的事件流流入系统，系统处理能力必须与事件流量匹配。

1. 实时计算应用场景

互联网领域的实时流计算一般都是针对海量数据进行的，除了非实时计算的需求（如计算结果准确）以外，实时计算最重要的一个需求是能够实时响应计算结果，一般要求为秒级。个人理解，互联网行业的实时计算可以分为以下两种应用场景。

1. 数据源是实时的、不间断的，要求对用户的响应时间也是实时的。主要用于互联网流式数据处理。所谓流式数据，是指将数据看作数据流的形式来处理。数据流则是在时间分布和数量上无限的一系列数据记录的集合体；数据记录是数据流的最小组成单元。例如，对于大型网站，活跃的流式数据非常常见，这些数据包括网站的访问PV/UV、用户访问的内容、搜索的内容等。实时的数据计算和分析可以动态实时地刷新用户访问数据，展示网站实时流量的变化情况，分析每天各小时的流量和用户分布情况，这对于大型网站来说具有重要的实际意义。
2. 数据量大且无法或没必要预算，但要求对用户的响应时间是实时的。主要用于特定场合下的数据分析处理。当数据量很大，同时发现无法穷举所有可能条件的查询组合或者大量穷举出来的条件组合无用时，实时计算就可以发挥作用，将计算过程推迟到查询阶段进行，但需要为用户提供实时响应。
3. 实时计算处理流程

互联网上海量数据（一般为日志流）的实时计算过程可以划分为3个阶段：数据的产生与收集阶段、传输与分析处理阶段、存储对外提供服务阶段，下面分别进行简单介绍。



1. 数据实时采集

需求：功能上保证可以完整地收集到所有日志数据，为实时应用提供实时数据；响应时间上要保证实时性、低延迟（在1s左右）；配置简单，部署容易；系统稳定可靠等。

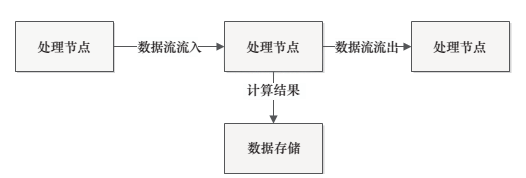
目前，互联网企业的海量数据采集工具有Facebook开源的Scribe、LinkedIn开源的Kafka、Cloudera开源的Flume，淘宝开源的TimeTunnel、Hadoop的Chukwa等，它们均可以满足每秒数百MB的日志数据采集和传输需求。

1. 数据实时计算

传统的数据操作，首先是将数据采集并存储在DBMS中，然后通过查询和DBMS进行交互，得到用户想要的答案。在整个过程中，用户是主动的，而DBMS系统是被动的，过程操作如图所示。



但是，对于现在大量存在的实时数据，如股票交易的数据，这类数据实时性强，数据量大，没有止境，传统的架构并不合适。流计算就是专门针对这种数据类型准备的。在流数据不断变化的运动过程中实时地进行分析，捕捉到可能对用户有用的信息，并把结果发送出去。在整个过程中，数据分析处理系统是主动的，而用户却处于被动接收的状态，处理流程如图所示。



需求：适应流式数据、不间断查询；系统稳定可靠、可扩展性好、可维护性好等。有关计算的一些注意点：分布式计算、并行计算（节点间的并行、节点内的并行）、热点数据的缓存策略、服务端计算。

1. 实时查询服务

全内存：直接提供数据读取服务，定期转存到磁盘或数据库进行持久化。

半内存：使用Redis、Memcache、MongoDB、BerkeleyDB等内存数据库提供数据实时查询服务，由这些系统进行持久化操作。

全磁盘：使用HBase等分布式文件系统（HDFS）为基础的NoSQL数据库，对于Key-Value内存引擎，关键是设计好Key的分布。

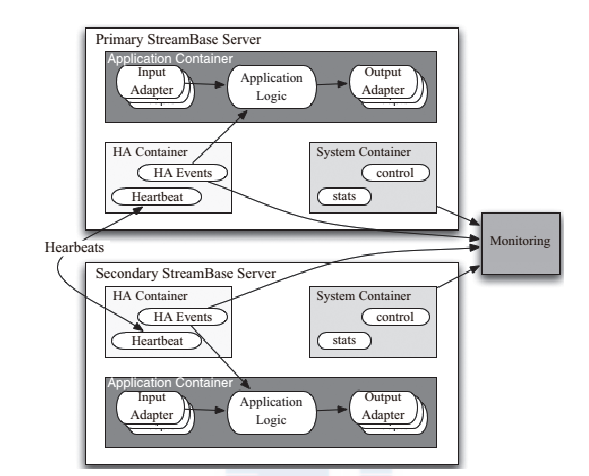
1. 实时计算框架

最近这几年随着实时计算的流行，相继出现了以下实时计算框架。

1. IBM的StreamBase

StreamBase是IBM开发的一款商业流式计算系统，在金融行业和政府部门使用，其本身是商业应用软件，但提供了开发版。相对于付费使用的企业版，开发版的功能更少，但这并不妨碍我们从外部使用API接口来对StreamBase本身进行分析。

StreamBase使用Java开发，IDE是基于Eclipse进行二次开发，功能非常强大。StreamBase也提供了相当多的Operator、Fuctor以及其它组件来帮助构建应用程序。用户只需要通过IDE拖拉空间，然后关联，设置好传输的Schema并且设置控件计算过程，就可以编译出一个高效处理的流式应用程序。同时，StreamBase还提供了类SQL来描述计算过程。StreamBase的架构如图所示：



StreamBase Server是节点上启动的管理进程，它负责管理节点上Container的实例，每个Container通过Adapter获得输入，交给应用逻辑计算，然后通过Adapter输出。各个Container相互连接，形成一个计算流图。

Adapter负责与异构输入或输出交互，源或目的地可能包括CSV文件、JDBC、JMS、Simulation( StreamBase提供的流产生模拟器 )或用户定制。

每个StreamBase Server上面都会有一个System Container，主要是产生系统监控信息和流式数据。

HA Container用于容错恢复，可以看出它实际包含两个部分：Heartbeat和HA Events，其中Heartbeat也是Tuple在Container之间传输。在HA方案下，HA Container监控Primary Server的活动情况，然后将这些信息转换成为HA Events交给StreamBase Monitor来处理。

Monitor就是从System Container和HA Container中获取数据并进行处理。StreamBase认为HA问题应该通过CEP方式处理，也就是说出现问题的部件肯定会反映在SystemContainer和HA Container的输出流上面，Monitor如果通过复杂事件处理这些Tuples就能够检测到机器故障等问题，并做出相应处理。

1. Yahoo的S42

Yahoo! S4（Simple Scalable Streaming System）是一个通过的、分布式的、可扩展的、分区容错的、可插拔的流式系统。基于S4框架，开发者可以容易地开发面向持续流数据处理的应用。S4的最新版本是v0.6.0，是Apache孵化项目，其设计特点有以下几个方面。

* Actor计算模型：

为了能在普通机型构成的集群上进行分布式处理，并且在集群内部不使用共享内存，S4架构采用了Actor模式，这种模式提供了封装和地址透明语义，因此在允许应用大规模并发的同时，提供了简单的编程接口。S4系统通过处理单元（Processing Elements, PEs）进行计算，消息在处理单元间以数据事件的形式传统，PE消费事件，发出一个或多个可能被其他PE处理的事件，或者直接发布结果。每个PE的状态对于其他PE不可见，PE之间唯一的交互模式就是发出事件和消费事件。

* 对等集群架构：

S4采用对等架构，集群中的所有处理节点都是等同的，没有中心控制节点，这使得集群的扩展性很好，处理节点的总数理论上无上限；同时，S4没有单点容错的问题。

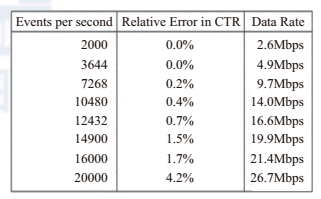
* 可插拔体系架构：

S4系统使用Java语言开发，采用了极富层次的模块化编程，每个通用功能点都尽量抽象出来作为通用模块，而且尽可能地让各模块实现可定制化。

* 支持部分容错：

基于ZooKeeper服务的集群管理层会自动路由事件从失效节点到其他节点。除非显式保存到持久性存储，否则节点故障时，节点上处理事件的状态会丢失。

S4的重要应用场景是预估点击通过率（CTR）。CTR是广告点击数除以展现数得到的比率，拥有足够历史的展现和点击数据后，CTR是用户点击广告可能性的一个很好的估算，精确的来源点击对于个性化和搜索排名来说都价值无限。据S4的开发者称，在线流量上的实验显示基于S4系统的新CTR计算框架可以在不影响收入的前提下将CTR值提高3%，这主要是通过快速检测低质量的广告并把它们过滤出去而获得的收益。S4系统提供的低延迟处理能够使得商务广告部门获益，但是潜在的风险也不能忽视，那就是事件流的速率快到一定程度后，S4可能无法处理，会导致事件的丢失，如图所示：



1. Twitter的Storm

Twitter的Storm是一个分布式的、容错的实时计算系统。Storm的用途，可用于处理消息和更新数据库（流处理），在数据流上进行持续查询，以流的形式返回结果到客户端（持续计算），并行化一个类似实时查询的热点查询（分布式的RPC）。

Storm为分布式实时计算提供了一组通用原语，可被用于“流处理”中，实时处理消息并更新数据库。这是惯例队列及工作者集群的另一种方式。Storm也可用于“连续计算”（continuous computation），对数据流做连续查询，在计算时将结果以流的形式输出给用户。它还用于“分布式RPC”，以并行的方式运行昂贵的运算。

Storm的主要特点如下：

* 简单的编程模型。类似于MapReduce编程降低了并行批处理复杂性，Strorm降低了进行实时处理的复杂性。
* 可以使用各种编程语言。可以在Storm上使用各种编程语言。默认支持Clojure、java、Ruby和Python。要增加对其他语言的支持，只需实现一个简单的Storm通信协议即可。
* 容错性。Storm会管理工作进程和节点的故障。
* 水平扩展。计算是在多个线程、进程和服务器之间并行进行的。
* 可靠的消息处理。Storm保证每个消息至少能得到一次完整处理。当任务失败时，它会负责从消息源重试消息。
* 快速。系统的设计保证了消息能得到快速的处理，使用ZeroMQ作为其底层消息队列。
* 本地模式。Storm有一个“本地模式”，可以在处理过程中完全模拟Storm集群。这可以使用户快速进行开发和单元测试。

1. Twitter的Rainbird

Rainbird是一款分布式实时统计系统，可以用于实时数据的统计。

1. 统计网站中每一个页面、域名的点击次数。
2. 内部系统的运行监控（统计被监控服务器的运行状态）。
3. 记录最大值和最小值。

Rainbird构建在Cassandra上，使用Scala编写，依赖于ZooKeeper、Scribe和Thrift。每秒可以写入10万个事件，而且都带有层次结构，或者各种查询，延迟小于100ms。目前Twitter已经在Promoted Tweets、微博中的链接、短地址、每个用户的微博交互等生产环境使用了Rainbird。其主要组件的功能如下。

* ZooKeeper：

是Hadoop子项目中的一款分布式协调系统，用于控制分布式系统中各个组件的一致性。

* Cassandra：

是NoSQL中一款非常出色的产品，集合了Dynamo和BigTale特性的分布式存储系统，用于存储需要统计的数据，并提供客户端查询统计数据（需要使用分布式Counter补丁CASSANDRA-1072）。

* Scribe：

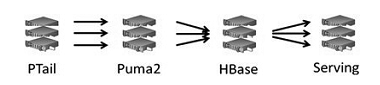
是Facebook开源的一款分布式日志收集系统，用于在系统中将各个需要统计的数据源收集到Cassandra中。

* Thrift：

是Facebook开源的一款跨语言C/S网络通信框架，开发人员基于该框架可以轻松地开发C/S应用。

1. Facebool的Puma

Puma是Facebook的数据流处理系统，早期的处理系统如图所示，这里所说的PUMA即二代Puma。



在Puma 2中，PTail将数据以流的方式传递给Puma 2，Puma 2每秒需要处理百万级的消息，处理多为Aggregation方式的操作，遵循时间序列，涉及的复杂Aggregation操作诸如独立访次、最频繁事件等等。

对于每条消息，Puma 2发送”Increment”操作到HBase。考虑到自动负载均衡、自动容错和写入吞吐等因素，Puma选择HBase而不是MySQL作为其存储引擎。Puma 2的服务器都是对等的，即同时可能有多个Puma 2服务器向HBase中修改同一行数据。因此，Facebook为HBase增加了新的功能，支持一条Increment操作修改同行数据的多列。

Puma 2的架构非常简单并且易于维护，其涉及的状态仅仅是PTail的Checkpoint，即上游数据位置周期性地存储在HBase中。由于是对称结构，集群扩容和机器故障的处理都非常方便。不过，Puma 2的缺点也很突出，首先，HBase的Increment操作是非常昂贵的，因为它涉及读和写，而HBase的随机读效率比较差；另外，复杂Aggregation操作也不好支持，需要在HBase上写很多用户代码；再者，Puma 2在故障时会产生少量重复数据，因为HBase的Increment和PTail的Checkpoint并不是一个原子操作。

但值得一提的是，Puma并没有开源出来，用户可以了解和借鉴其实现原理。

1. 阿里的JStorm

JStorm是一个Alibaba开源的分布式实时计算引擎，可以认为是Twitter Storm的Java版本，用户按照指定的接口实现一个任务，然后将这个任务递交给JStorm系统，JStorm会启动后台服务进行7x24小时运行，一旦某个Worker发生故障，调度立即分配一个新的Worker替换这个失效的Worker。

JStorm处理数据的方式是基于消息的流水线处理，因此特别适合无状态计算，也就是计算单元依赖的数据全部可以在接受的消息中找到，并且最好一个数据流不依赖另外一个数据流。因此，JStorm适用于下面的场景：

* 日志分析

从日志中分析出特定的数据，并将结果存入外部存储器，如数据库。

* 管道系统

将数据从一个系统传输到另外一个系统，如将数据库同步到Hadoop。

* 消息转化器

将接收到的消息按照某种格式转化，存储到另外一个系统，如消息中间件中。

* 统计分析器

从日志或消息中提炼出某个字段，然后进行COUNT或SUM计算，最后将统计值存入外部存储器。

但是，JStorm的活跃度并不高，截至本章节书写时，整个JStorm项目共提交过36次，并且只有1个Committer，相比Twitter Storm，不管活跃度还是认可度都还不是一个数量级的产品。

1. 其他实时计算系统
   * 1. HStreaming

HStreaming是建立在Hadoop上的可扩展的、可持续的数据分析系统。它可以分析、可视化并处理大量连续数据，如一个金融交易系统实时展示数据图。HStreaming由Jana Uhlig与Volkmar Uhlig联合创立，该公司没有提供相关产品的开源版本，从官网信息来看，只提供相关的解决方案。

HStreaming公司尝试为Hadoop环境添加一个实时的组件，当数据提交到系统，在存储到磁盘之前会进行数据处理，类似开源的Storm和Kafka。目前HStreaming已经建立了一个完整的系统，该系统能够利用实时的引擎来处理视频、服务器、传感器以及其它机器上生成的数据流，而且完全兼容Hadoop作为一个归档和批量处理系统。

* + 1. Esper

Esper是EsperTech公司使用Java开发的事件流处理（Event Stream Processing， ESP）和复杂事件处理（Complex Event Processing，CEP）引擎。CEP是一种实时事件处理并从大量事件数据流中挖掘复杂模式的技术。ESP是一种从大量事件数据流中过滤、分析有意义的事件，并能够实时取得这些有意义的信息的技术。该引擎可应用于网络入侵探测、SLA监测、RFID读取、航空运输调控、金融（风险管理、欺诈探测）等领域。Esper可以用在股票系统、风险监控系统等实时性要求比较高的系统中。

* + 1. Borealis

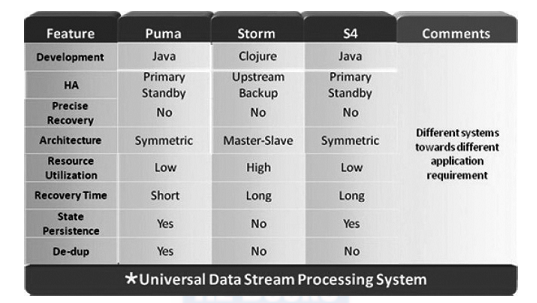
Borealis是有Brandeis University、Brown University和MIT合作开发的一个分布式流式系统，由之前的流式系统Aurora、Medusa演化而来，是学术研究的一个产品，2008年已经停止维护。

Borealis具有丰富的论文、完整的用户/开发者文档，系统是用C++实现的，运行于x86-based Linux平台。系统是开源的，同时使用了较多的第三方开源组件，包括用于查询语言翻译的ANTLR、C++的网络编程框架库NMSTL等。

Borealis系统的流式模型和其他流式系统基本一致：接受多元的数据流和输出流，为了容错，采用确定性计算，对于容错性要求高的系统，会对输入流使用算子进行定序。

1. 框架对比

实时数据流计算是近年来分布式、并行计算领域研究和实践的重点，无论是工业界，还是学术界，都诞生了多个具有代表性的数据流计算系统，用于解决实际生产问题和进行学术研究。不同的系统满足不同应用的需求，系统并无好坏之分，关键在于服务的对象是谁。下图从开发语言、高可用机制、支持精确恢复、主从架构、资源利用率、恢复时间、支持状态持久化及支持去重等几个方面比较了典型的3个数据流计算系统Puma、Storm和S4。因为StreamBase是厂商发行的商用版本，HStreaming只提供解决方案，而JStorm和Storm非常相似，所以这几种产品并没有罗列在图中。



可以看到，为了高效开发，两个系统使用Java，另一种系统使用函数式编程语言Clojure;高可用方案，有两个系统使用Primary Standby方式，系统恢复时间可控，但系统复杂度增加，资源使用率也较低，因为需要一些机器来当备机；而Storm选择了更简单可行的上游回放方式，资源使用率高，就是恢复时间可能稍长些；Puma和S4都支持状态持久化，但S4目前不支持数据去重，未来可能会实现；三个系统都做不到精确恢复，即恢复后的执行结果和无故障发生时保持一致，因为即使是Primary Standby方式，也只是定期Checkpoint，并没有跟踪每条消息的执行。商用的StreamBase支持精确恢复，这主要应用于金融领域。