开源力量
Open Source Force
www.osforce.cn

HBase协处理器

主讲人： Louis

协处理器

HBase作为列数据库最经常被人诟病的特性包括：
无法轻易建立“二级索引”，难以执行求和、计数、排序等操作。

比如，在旧版本的(<0.92)Hbase中，统计数据表的总行数，需要使用Counter方法，执行一次MapReduce Job才能得到。虽然HBase在数据存储层中集成了MapReduce，能够有效用于数据表的分布式计算。

然而在很多情况下，做一些简单的相加或者聚合计算的时候，如果直接将计算过程放置在server端，能够减少通讯开销，从而获得很好的性能提升。于是，HBase在0.92之后引入了协处理器(coprocessors)，实现一些激动人心的新特性：能够轻易建立二次索引、复杂过滤器(谓词下推)以及访问控制等。

协处理器

HBase协处理器的灵感来自于Jeff Dean 09年的演讲(P66-67)。它根据该演讲实现了类似于bigtable的协处理器，包括以下特性:

1) 每个表服务器的任意子表都可以运行代码

2) 客户端的高层调用接口(客户端能够直接访问数据表的行地址，多行读写会自动分片成多个并行的RPC调用)

3) 提供一个非常灵活的、可用于建立分布式服务的数据模型

4) 能够自动化扩展、负载均衡、应用请求路由

HBase的协处理器灵感来自bigtable，但是实现细节不尽相同。HBase建立了一个框架，它为用户提供类库和运行时环境，使得他们的代码能够在HBase region server和master上处理。

协处理器

协处理器分两种类型，系统协处理器可以全局导入region server上的所有数据表，表协处理器即是用户可以指定一张表使用协处理器。

协处理器框架为了更好支持其行为的灵活性，提供了两个不同方面的插件。一个是观察者（observer），类似于关系数据库的触发器。另一个是终端(endpoint)，动态的终端有点像存储过程。

Observer

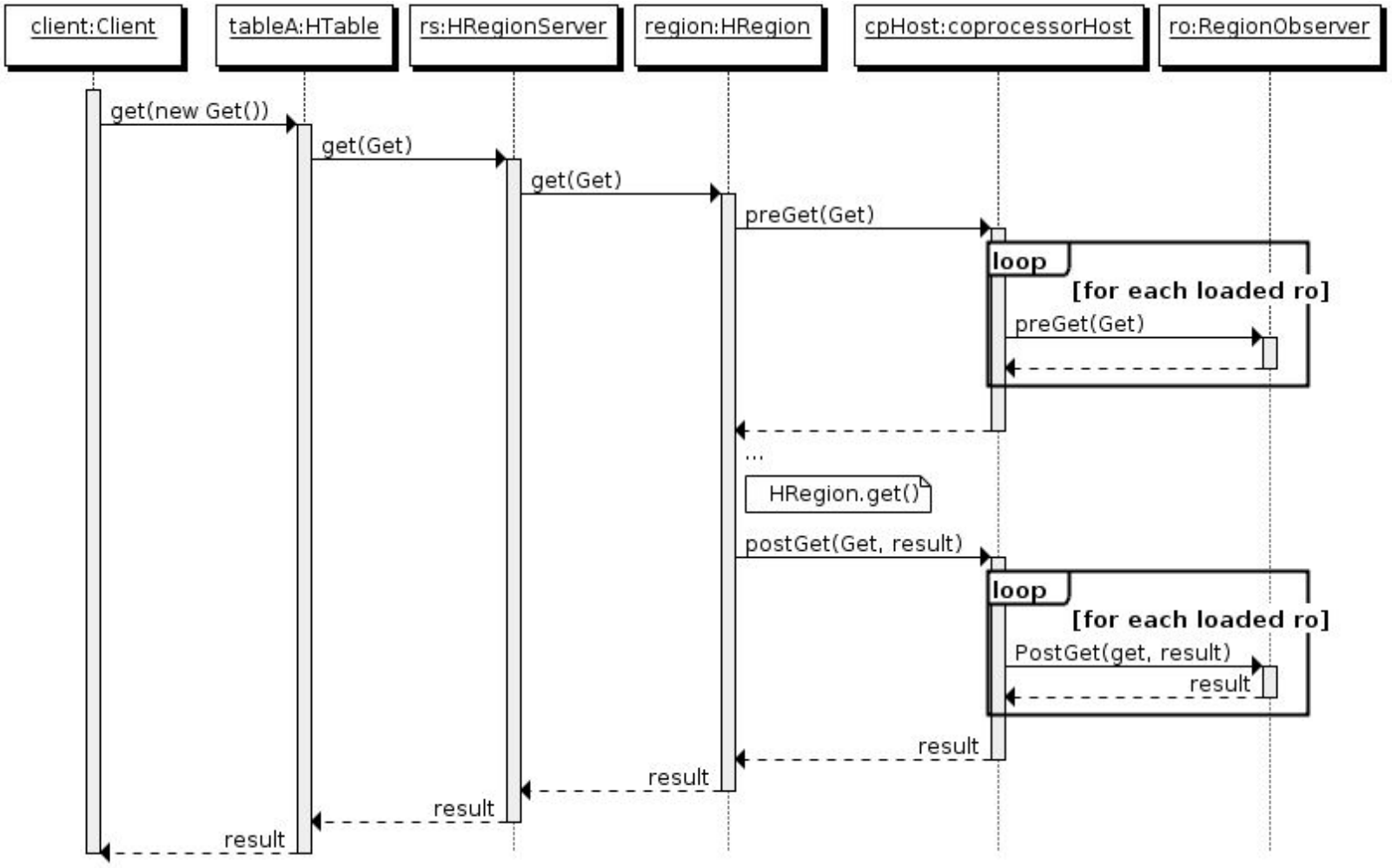
观察者的设计意图是允许用户通过插入代码来重载协处理器框架的upcall方法，而具体的事件触发的callback方法由HBase的核心代码来执行。协处理器框架处理所有的callback调用细节，协处理器自身只需要插入添加或者改变的功能。

以HBase0.92版本为例，它提供了三种观察者接口：

RegionObserver：提供客户端的数据操纵事件钩子：Get、Put、Delete、Scan等。
WALObserver：提供WAL相关操作钩子。
MasterObserver：提供DDL-类型的操作钩子。如创建、删除、修改数据表等。
这些接口可以同时使用在同一个地方，按照不同优先级顺序执行.用户可以任意基于协处理器实现复杂的HBase功能层。HBase有很多种事件可以触发观察者方法，这些事件与方法从HBase0.92版本起，都会集成在HBase API中。不过这些API可能会由于各种原因有所改动，不同版本的接口改动比较大。

Observer模型

开源力量 Open Source Force
www.osforce.cn



EndPoint

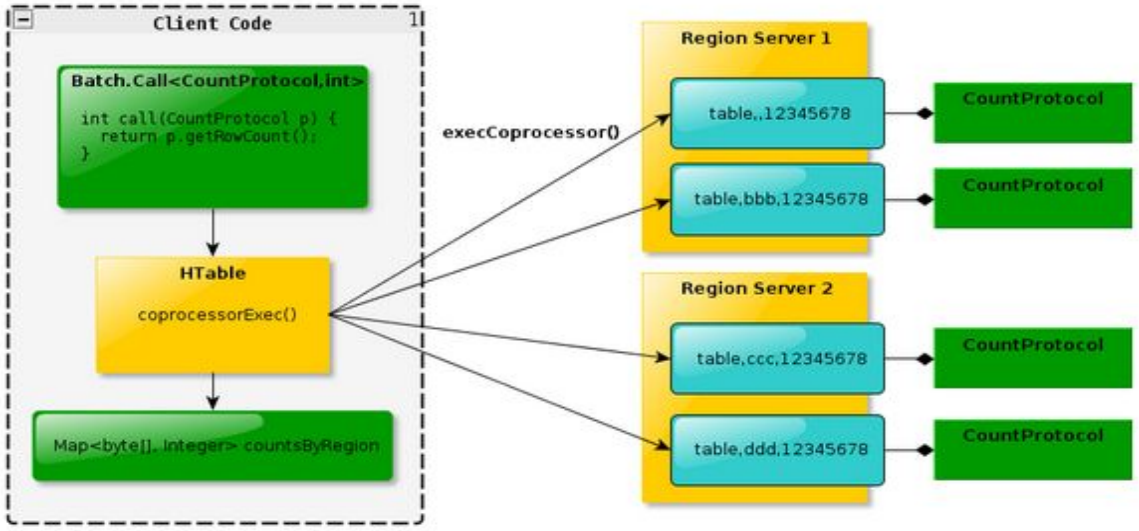
终端是动态RPC插件的接口，它的实现代码被安装在服务器端，从而能够通过HBase RPC唤醒。客户端类库提供了非常方便的方法来调用这些动态接口，它们可以在任意时候调用一个终端，它们的实现代码会被目标region远程执行，结果会返回到终端。用户可以结合使用这些强大的插件接口，为HBase添加全新的特性。终端的使用，如下面流程所示：

定义一个新的protocol接口，必须继承CoprocessorProtocol.

实现终端接口，该实现会被导入region环境执行。

继承抽象类BaseEndpointCoprocessor.

在客户端，终端可以被两个新的HBase Client API调用 。单个region：HTableInterface.coprocessorProxy(Class<T> protocol, byte[] row) 。regions区域：HTableInterface.coprocessorExec(Class<T> protocol, byte[] startKey, byte[] endKey, Batch.Call<T,R> callable)



EndPoint

有三个方法对Endpoint进行设置：
A. 启动全局aggregation，能过操纵所有的表上的数据。通过修改hbase-site.xml这个文件来实现，只需要添加如下代码：

```
<property>
  <name>hbase.coprocessor.user.region.classes</name>
  <value>org.apache.hadoop.hbase.coprocessor.RowCountEndpoint
</value>
 </property>
```

B. 启用表aggregation，只对特定的表生效。通过HBase Shell 来实现。
(1)disable指定表。hbase> disable 'mytable'
(2)添加aggregation hbase> alter 'mytable', METHOD =>
'table_att','coprocessor'=>'|org.apache.hadoop.hbase.coprocessor.RowCountEndpoint ||'
(3)重启指定表 hbase> enable 'mytable'

C. API调用

```
   HTableDescriptor htd=new HTableDescriptor("testTable");
   htd.setValue("CORPROCESSOR$1",path.toString+"|"+ RowCountEndpoint
.class.getCanonicalName()+"|"+Coprocessor.Priority.USER);
```

几点说明

1. 协处理器配置的加载顺序：先加载配置文件中定义的协处理器，后加载表描述符中的协处理器

2. COPROCESSOR$<number>中的number定义了加载的顺序

几点说明

3. 协处理器配置格式

Coprocessors can also be configured to load on a per table basis, via a shell command ``alter''.

```
hbase> alter 't1', METHOD => 'table_att',
 'coprocessor1'=>'hdfs:///foo.jar|com.foo.FooRegionObserver|1001|arg1=1,arg2=2'
```

The coprocessor framework will try to read the class information from the coprocessor table attribute value. The value contains four
pieces of information which are separated by ``|'':

- File path: The jar file containing the coprocessor implementation must located somewhere where all region servers can read it. The file could be copied somewhere onto
  the local disk of all region servers, but we recommended storing the file into HDFS instead. If no file path is given, the framework will attempt to load the class from the server classpath using the default class loader.
- Class name: The full class name of the coprocessor.
- Priority: An integer. The framework will determine the execution sequence of all configured observers registered at the same hook using priorities. This field can be left
  blank. In that case the framework will assign a default priority value.
- Arguments: This field is passed to the coprocessor implementation.