Design of HBase isolation and allocation

MOST OF THE BELOW HAS BEEN DONE BY HBASE-4120. OUTSTANDING IS REGION SERVERES ARE DIVIDED INTO GROUPS AND TABLES HAVE PRIORITIES.

Introduction

The HBase isolation and allocation tool is designed to help users manage cluster resource among different application and tables.
When we have a large scale of HBase cluster with many applications running on it, there will be lots of problems. In Taobao there is a cluster for many departments to test their applications performance, these applications are based on HBase. With one cluster which has 12 servers, there will be only one application running exclusively on this server, and many other applications must wait until the previous test finished.
After we add allocation manage function to the cluster, applications can share the cluster and run concurrently. Also if the Test Engineer wants to make sure there is no interference, he/she can move out other tables from this group.
In groups we use table priority to allocate resource, when system is busy; we can make sure high-priority tables are not affected lower-priority tables
Different groups can have different region server configurations, some groups optimized for reading can have large block cache size, and others optimized for writing can have large memstore size.
Tables and region servers can be moved easily between groups; after changing the configuration, a group can be restarted alone instead of restarting the whole cluster.
Though we put group and priority of tables together, users can use them separately, there is no inseparable link between this two functions.

Rules

Default group which holds “-ROOT-“and “.META.” regions can’t be deleted or restarted, the server holds “-ROOT-“and “.META.” regions can’t be move into other groups.
(Though when the servers in default groups are all down, system will find a random available server to hold root and meta, we suggest keep some servers in default group. Because when some brutal action like messed up the */{hbase_root}/groupinfomation.conf* file happens, if we find a table’s group have no server available, we will put this table into default group)
If you want to delete a group, move table and servers out of this group first, or the system will remind you “The group is not null”.
Group name must be a unique integer.
When a region server moves in or out a special configuration group, the server configuration will be updated.
If the table’s group attribute is “null”, its regions can be assign to all groups which don’t have special configuration tag.
Table priority must be an integer and value between 1 and 10, small number means high priority.
A table can use more than one groups, group attribute like “1,2” means this table’s region can be assign to group 1 and group 2.

Workflow of Background

In HBase isolation and allocation, *HMaster* will use *GroupAssignmentmanager* instead of *Assignmentmanager* to manage region assignment and *HRegionServer* will use *ScheduleHBaseServer* instead of *HBaseServer* to manage the RPC request.

Workflow of Group

Initiate configuration

After root and meta regions are assigned, system will start initiate the table-group information.
The information was stored in */{hbase_root}/groupinfomation.conf*, this file stored the region server to group information.
After the first time initiate, each time user change the group information like create a new group, move region servers between groups and so on, the web portal *jsp* will invoke *initValue()* to refresh the group information in memory.

Startup and Assignment

When start the cluster, user regions will be assigned by *GroupAssignmentManager*’s *assignAllUserRegons()* method. In this method, it will invoke *retainGroupAssignRegions()* or *groupAssignRegions()* method to assign regions to servers. *retainGroupAssignRegions()* will check the assignment in meta table first, if the region assignment is fit with the table’s group information, system will reuse the assignment, or system will use *getAvailableServer()* to get a list of available servers and choose a random one in this list.

Group Maintenance

There is a maintain thread which started in static block of *GroupAssignmentManager*. This thread will do two things, one is refresh the group information cached in memory; the second is to check region assignment, if there is some illegal assignments this thread will reassign them.
GroupAssignmentManager .balance(RegionPlan) will check the group information first and then descide whether put the plan in plan map.
GroupAssignmentManager .getRegionPlan(RegionState,HServerInfo,boolean) will get a random assignment in available servers which are choose according to group information.

Change table’s group attribute

When change the table’s group attribute, the table will be disable first then the group information will be write into table descriptor, after meta information of this table is changed system will refresh the table group information which cached in memory. Then the table will be enabled.

Workflow of Table Priority

We use a priority job queue and different priority job handlers to achieve this effect.

Initiate region server handlers

When *ScheduleHBaseServer* began to start its job handlers, we give different thread priorities to the handlers in *ScheduleHBaseServer.startThreads()*. Handler threads can only handle the jobs which have equal or higher priority.

Handle RPC calls

In *ScheduleHBaseServer.Connection.processData(),* the RPC calls will added into a *ScheduleQueue*, the priority is decided by the region’s priority cached in memory. If the cached priority is null, system will retrieve the region’s table descriptor to find the priority.
In *ScheduleQueue*, the calls are sorted by their priorities. So when the handler thread invokes the queue’s *get()*, it will get the highest priority RPC call.
When the times of one handler invoke the *ScheduleQueue .get()* reached a threshold value, the handler will interrupt the queue’s refresh thread and the refresh thread will reduce the priorities of the calls which in this queue by 1(small number means high priority), that will ensure low priority calls executed after a while of waiting.
When *ScheduleQueue*’s size reaches the capacity, the thread which adds calls to the queue will wait for a while and

check the size again, after the wait times exceed the threshold, the call will be added regardless of the capacity.

Workflow of Web Portal

There are many JSP pages; we use them to provide a GUI to manage the group division and table priorities. The detailed use cases of the GUI portal please refer to the user guide which you can download from https://github.com/ICT-Ope/HBase_allocation/raw/master/doc/HBase%20allocation.pdf .

Move server between groups

The *showgroup.jsp* page will read group configuration from HDFS first, Dead servers will be removed from group, new servers will be added to default group. This is done by functions *ServerWithGroup .readGroupInfo (HMaster master, final String confpath), ServerWithGroup.initGroupMap (HMaster master, String confline) and ServerWithGroup.initGroupPropertyMap (HMaster master, String confline).*

Before moving servers between groups, it will check if this server contains "-ROOT-"and ".META." region, then it will construct a *MoveGroupPlan* instance, it contains region server name, original group name and target group name. The move server action is processed by *processgroup.jsp.*

If there is a *MoveGroupPlan*, this page will set system busy and start a thread to move region servers to other group. This is done by Class *ProcessMove.* It will check available servers in this group by *GroupAssignmentManager.getAvailableServer (tablename) and move regions which belong to the MoveGroupPlan's region server to the available servers*. Then it will check if the source group or the target group is special configured. If so we must update the server's configuration to new one which includes hbase.jar, lib fold and conf fold. This is done by Class *MoveConfImpl.* This class contains 3 static methods.

public boolean ScpConf(String server, String command);
public boolean ImplRegionServer(String server, String command);
public boolean DistributeConf(String server, String command);

These methods will run local bash scripts to finish these operations. There are two bash script *moveremoteconf.sh* and *restartserver.sh.*

Load Balance

In this groupinfo.jsp page we provide an anchor linked to *dogroupbalacne.jsp; it* will start a thread to balance all regions in this group. This function is implemented by *GroupAssignmentManager.balanceGroup (group);*
Then this page will list detail information of all tables in this group. It contain region numbers of table, table priority, table group and an anchor to *dotablebalacne.jsp.* When the balance anchor clicked, a thread used to balance all regions of this table will be started, The balance table function is implemented by *GroupAssignmentManager.balanceTable (table);*

You can change table priority and table group by click the button at the end of each row. It will start a thread to invoke methods *GroupAssignmentManager.setGroup (groups, tablename) or GroupAssignmentManager.setPriority(priporty, tablename);*