# Overall

In the current coprocessor service invocation, for each region the client starts a thread to send a call, shown in the figuer1. If there’s one region server, and 100 regions reside in this server, each coprocessor service will send 100 calls, each call uses a single thread in the client side. The threads will run out soon when there’re too many invocations on the coprocessor service.

Figure1

In this design, all the calls to the same region server will be batched into one single coprocessor invocation. This call will be spread into each region in the server side, and the result will be merged ahead in the server side before being returned to the client. It’s shown in the figure2.

Figure2

This implementation could significantly reduce the thread cost for the coprocessor service invocations in the client side. In the above case, each coprocessor service invocation only needs one thread in the client side whereas the non-batch coprocessor service invocation needs 100 threads.

# Implementation

## Server Side

Add a new API *execBatchService* to the *HRegionServer*. In this method, the call to this region server are spread into the regions, the results are merged in the server side and returned to the client.

1. Parse the *BatchCoprocessorServiceRequest* into multiple *CoprocessorServiceCall* which could be used to invoke the Region directly.
2. Merge the results come from regions with the user defined function. The server side callback function (merge function) and its initial data could be found in the *BatchCoprocessorServiceRequest.*
3. Return the merged result to the client.

## Client Side

In the client side, we’ll firstly find all the necessary regions according the argument of the key ranges. Then we’ll group these regions by server names, and send these invocations server by server.

We don’t add APIs to the *HTable* directly, instead we add a new class *BatchCoprocessorServiceClient* to communicate with the *HRegionServer*.

In this class, we define several APIs. Besides the same parameters with the ones in the *coprocessorService* of *HTable*, we define some more arguments.

1. HTable: The table that the coprocessor service is executed against.
2. ExecuteService: The pool used in the *BatchCoprocessorServiceClient.*
3. ServerCallbackClassName: The results come from regions will be merged in the server side ahead. The merge logic is defined in the ServerCallbackClassName.
4. InitialData: The initial data used by the ServerCallbackClassName.
5. keyRanges: The key ranges that the coprocessor service is executed against.