Combiner

Combiners

Combiners are used for performing message aggregation to reduce communication overhead in cases when messages can be summarized arithmetically e.g., min, max, sum, and average at the sender side. Suppose that you want to send the integer messages to a specific processor from 0 to 1000 and sum all received the integer messages from all processors.

```
  public void bsp(BSPPeer<NullWritable, NullWritable, NullWritable, NullWritable> peer) throws IOException,
      SyncException, InterruptedException {

    for (int i = 0; i < 1000; i++) {
          peer.send(masterTask, new IntegerMessage(peer.getPeerName(), i));
    }
    peer.sync();

    if (peer.getPeerName().equals(masterTask)) {
      IntegerMessage received;
      while ((received = (IntegerMessage) peer.getCurrentMessage()) != null) {
        sum += received.getData();
      }
    }
  }
```

If you follow the previous example, Each bsp processor will send a bundle of thousand Integer messages to a masterTask. Instead, you could use a Combiners in your BSP program to perform a sum Integer messages and to write more concise and maintainable as below, that is why you use Combiners.

```
public static class SumCombiner extends Combiner {

  @Override
  public BSPMessageBundle combine(Iterable<BSPMessage> messages) {
    BSPMessageBundle bundle = new BSPMessageBundle();
    int sum = 0;

    Iterator<BSPMessage> it = messages.iterator();
    while (it.hasNext()) {
      sum += ((IntegerMessage) it.next()).getData();
    }

    bundle.addMessage(new IntegerMessage("Sum", sum));
    return bundle;
  }

}
```