Bringing Together Millions of Publishers and Thousands of Advertisers by Zanox Reporting Systems built on top of Hadoop and Lucene Technologies

Dr. Dragan Milosevic, Senior Hadoop Architect, dragan.milosevic@zanox.com

Zanox is a company which deals with huge amount of data. Currently our tracking gets more than 2 million sales, 30 million clicks and almost 1 billion views every day. A further huge amount of data is coming from search-engines that provide information about related costs. The challenge is to join and summarise those huge sets of data and provide valuable tracking and cost statistics to more than 1 million publishers and 10 thousands advertisers. They will use our analyses to successfully drive their business by knowing which launched campaigns are well performing and how others can be changed to earn even more money. This talk will have two parts. The first part shows how Hadoop helps analysing available data and saving valuable results in Lucene indexes. The second part describes Lucene search infrastructure which is able to efficiently retrieve reports in real time for publishers and advertisers.

Challenges to be solved during Hadoop processing can be summarised as follows:

(1) **Joining several huge sets of data**, namely the data which is tracked by Zanox, costs-data coming from search-engines and master-data about publishers and advertisers. We have developed an unique approach which combines map-side and reduce-side joins, trying to use the best of both techniques. It does sampling to identify records which have frequent join-keys. Those records will be joined on a map-side and directly written to output without loading map-reduce pipeline. Only records whose join-key is infrequent will be propagated to reducers in order to be joined there. Provided experiments have shown significant gains compared to pure reduce-side join mainly due to the fact that more than 80% of records can be joined on a map-side, resulting that only one fifth of records have to be sorted and joined on a reduce-side.

(2) **Several aggregation jobs** turned out to be very important for us, mainly due to the huge number of records that have to be summarised. We have made many experiments in order to speed-up map-reduce engine, which in our particular case was slow mainly due to the fact that sorting on a reduce-side has to wait until all map-tasks have been completed. In our scenario where thousands of map-tasks have to be executed, reduce tasks are waiting a significant amount of time. Our solution is very simple, and it is based on replacing one huge job with several smaller ones which are responsible only for selected parts of the input data. Those smaller jobs have much fewer map-tasks, which have to be finished before reduce-tasks can start with expensive sorting and aggregation operations. Consequently, we have managed to activate resources earlier that are responsible for reduce tasks, and to speed-up the complete aggregation for more than 30%.

(3) **Lucene indexes save every aggregated result** produced by Hadoop jobs to make them real-time available to publishers and advertisers. Created indexes represent different views to aggregated data, being responsible to speed-up the processing of queries. We have optimised indexing performance by (a) building several levels of aggregations of every record simultaneously, (b) using intelligent partitioners that send the semantically same aggregated-levels of records to same reducers, and (c) taking care of producing medium-size indexes that can be generated completely in memory on a reduce-side.

Lucene search infrastructure is dealing with the real-time generation of reports on a request basis. Because the response time plays very important role while serving online requests, the architecture of our search-backend is optimised by following actions:

(1) **Indexing data with different aggregation levels** to optimise the execution of various types of queries. Ideally the requested information is already directly indexed in which case it is only necessary to find needed record and to simply forward it to publisher or advertiser. For example, one publisher might be interested in knowing the number of views on a particular day. If such an aggregation is already pre-computed by Hadoop and directly available in Lucene index, reporting is simply searching for a record that corresponds to given publisher and day. Those reports can be generated in a matter of only several milliseconds.

(2) **Estimating costs of processing every received query** to select the best available index and to optimise response-time for report-generation. Due to the great flexibility that publishers and advertisers have in building their own customised queries, it is not possible to pre-compute all possible aggregation-levels for every single query that can be received. Our approach is capable of estimating costs of processing each received query by every available index type. The index that has the closest aggregation-level will be chosen, because it guarantees that the amount of aggregation that has to be performed on-the-fly is the smallest. For example, if one advertiser wants to know the number of clicks in the particular time-period, the given time-period will be first split on days, months and years. Indexes with year-aggregation are preferred over month and day-aggregations, due to having smaller on-the-fly aggregation requirements. Consequently the most expensive day-aggregation will be used only for the parts of the requested time-period where month and year aggregations are not applicable.

(3) **Profit-aware combination of memory and file-system indexes** provides nice business models by saving more important data in memory-indexes and others in file-system ones. In practice this means that the important publishers and advertisers are getting more space in memory-indexes and therefore they will get their reports faster. The implemented solution can nicely ensure that the amount of memory-indexes used by given publisher or advertiser is proportional to the paid price margins.

Those two parts of our reporting-system have already a long history behind. System is productive since 2009, and one of its first versions has been already presented on Hadoop-Get-Together in March 2010 (http://vimeo.com/10201534). Its enhancements made in the last two years are here briefly summarised. If both parts of the system are to be presented 40 minute slot will be more than welcome. Never the less, we can always focus presentation either on Hadoop or Lucene part, in which case it will be possible to fit everything in 20 minutes.