**Indexing Larger Spatial Data using Hadoop, a closer look at spatialhadoop and its complementary QuadTree implementation**

Donghan Miao

**Building Index**

Generically, database storage consists of persisted index structure and data storage, but neither part should infinitely scale up. In big data environment, we leverage the advantage of distribution and parallelism to achieve scalability.

Indexing process is partitions large data set into even-sized data chunks (the number of chunks is determined by the volume of spatial dataset). Each data chunk is a small but fully-fledged database storage.

Data

Index

Data chunk, size limit 64MB

By default, the block size of Hadoop distributed file system (HDFS for short hereafter) is 64MB. Therefore our default chunks should be close to but no larger than that size, to guarantee i/o efficiency.

**Partitioning phase one**

In this phase, partitioning is done by mapreduce framework, the level of parallelism is largely determined by the volume of data set. Mapreduce framework will split the files into smaller chunks before submitting them to mappers. Each mapper instance has its own unique id.

**Partitioning phase two**

Reducers receiving the spatial objects with same id will aggregate and partition them into chunks.

**Executing a Query**