# Parallelising QuantLib using Map-Reduce

Bojan Nikolic
BN Algorithms Limited
http://www.bnikolic.co.uk

June 6, 2012

**Abstract**

In this White Paper, I describe the approach developed recently at BN Algorithms Ltd for easy-to-use and very highly scalable parallelisation of QuantLib portfolio and scenario calculations using the Map-Reduce technique. Leveraging the industry standard Apache Hadoop[1] software framework means that commercial cloud computer providers (such as Amazon Elastic MapReduce[2]) can be efficiently and easily used, with scaling to 1000s of nodes without significant capital investments.

## 1 Introduction

QuantLib[3] is a popular, open source, C++ library for pricing of derivatives and other financial contracts. It is often used either by developing C++ application on top of the library or as an Microsoft Excel[4] Addin, where C++ library is wrapped into functions that can be called directly from the Excel worksheets. In particular when used through the Excel Addin, the library is straightforward to use in simple and complex business scenarios alike. These typical usage scenarios normally only utilise only one CPU core of a single computer.

For some application however, the time required to perform calculations using QuantLib on a single CPU core becomes prohibitively long. This is usually due to a *combination* of:

1. Complexity of calculation of individual prices for some instruments
2. Having to value a whole portfolio of instruments, consisting of 100s to 1000s of individual instruments
3. Necessity of doing *scenario analysis* where the whole valuation process is repeated after making multiple small adjustments to the input market data.

Distributing the computation of the price of a single instrument across multiple cores is in general a complex task and furthermore closely dependent on the actual algorithm used for pricing of this instrument. On the other hand, distribution of computation of different scenarios and instruments in a portfolio across different computers is a relatively simple task because it consists of *distribution of data* for computation without any further dependencies in the algorithm on the results of this computation.

Indeed distribution and parallelisation of items (2) and (3) above has been the primary means by which the acceleration of computation of prices of derivative contracts has been achieved in most commercial system, including those based on QuantLib. The parallelisation technique I describe here is also at the level of these two items but with a particular focus on the data-distribution aspect which in turn leads naturally the use of Map-Reduce techniques. I show that by leveraging existing Map-Reduce

[1] http://hadoop.apache.org/
[2] http://aws.amazon.com/elasticmapreduce/
[3] http://quantlib.org
[4] Trademark of the Microsoft corporation

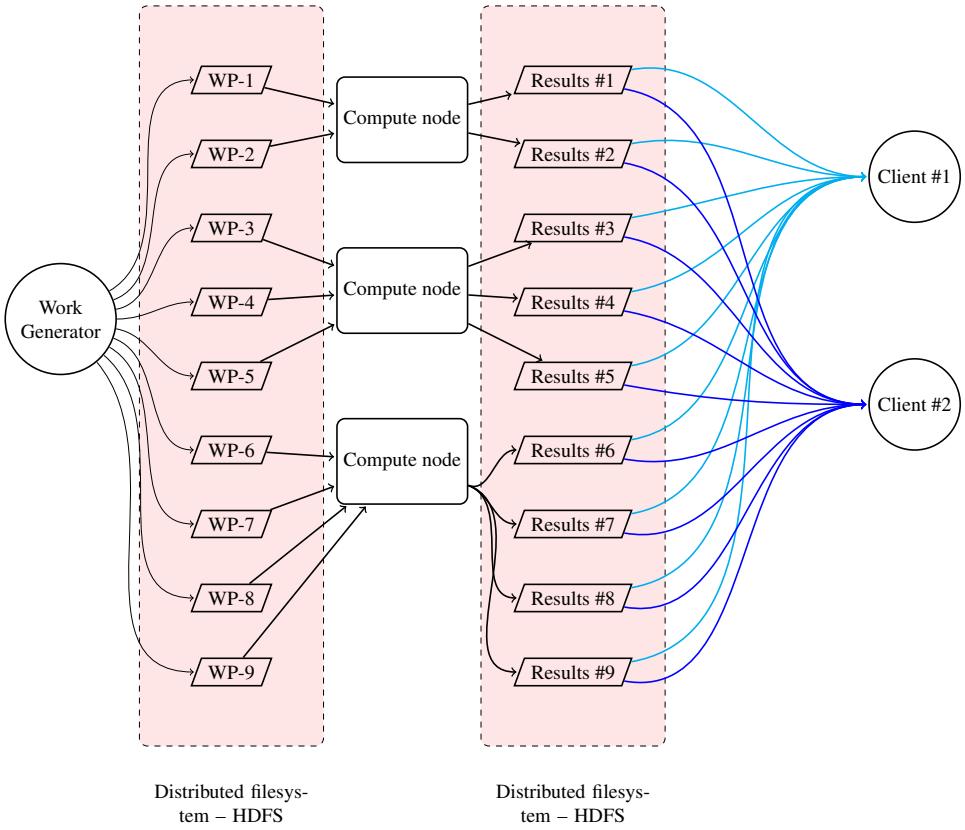


Figure 1: Schematic diagram of parallelisation via the Map-Reduce technique. Note the work packages (WPs) and results are stored (semi-)permanently on a distributed file system while the computation is controlled by the map-reduce framework. In practice a much larger number of nodes and work packages can be used.

software frameworks and clusters it is possible make extremely reliable and easy to use parallel QuantLib systems which are immediately usable from a wide range of business applications.

## 2 How is Map-Reduce applicable to parallelisation of QuantLib (and other derivative pricing systems)?

As noted above, most practical systems for pricing of derivatives implement parallelisation on level of an instrument or a more coarse level. That is, a price of a single instrument is computed on one core (or sometimes node) only but the there are many such nodes pricing different instruments and/or under different market condition assumptions. In such a system several logical stages of processing can be identified:

1. **Work generation**, where the combination of different market conditions and instruments is made to generate specification of prices that need to be computed
2. **Work packaging**, where the individual pricing tasks are combined to form units which are large enough that the overheads of distribution over network are not significant
3. **Work distribution**, i.e., distribution of tasks to the compute nodes

4. **Computation**, i.e., the computation of prices from the specification in recorded in work package
5. **Collection of results**, i.e., centralisation and summarising of results of computation from all of the nodes

The principle of applying Map-Reduce to this system is to identify the output of stage (2), i.e., work packaging, not just as an ephemeral request sent over the network but as a permanent (or semi-permanent) *file* on a distributed file system. Once this idea is adopted, the actual computation of prices can be modelled as the *map* process, the collection of results can be done partially by the *reduce* or simply by making use of the distributed file system underlying the Map-Reduce concept (illustrated in Figure 1). The whole job of work distribution, fail over, control, etc, can of course then be taken over by the adopted Map-Reduce framework.

# 3 The advantages of the Map-Reduce approach

What are the advantages of using the Map-Reduce approach over more conventional approaches which would be based on a *grid engine?* Conceptually, the advantages stem from the decomposition of the overall system into completely independent components which communicate via simple files on a distributed file system. This is augmented with the practical advantages of access to many, highly developed, Map-Reduce technologies and ready-to-use deployments.

The resultant advantages to organisations implementing this approach are:

1. A simple, easy to understand system – due to separated components and ability to inspect intermediate files
2. Robust and reliable operation – due to use of well developed existing technologies (such as Apache Hadoop) and the availability of intermediate files to restart jobs
3. Very high scalability – due to re-use of technologies developed at leading Internet companies for generic processing of large data sets
4. Fast implementation and incorporation into production – can use existing infrastructures or cloud-based providers
5. Small capital costs – the standardised nature of Map-Reduce jobs means that use of cloud providers (which charge by time only) is particularly easy
6. Reuse of results of risk and scenario analysis – the results are stored by default on a networked, distributed, highly reliable file system
7. Complete trail of operations is automatically available if required for internal controlling purposes

# 4 Parallelisation Services offered by BN Algorithms

BN Algorithms has a complete implementation of QuantLib Parallel pricing system based on following technologies:

- Our *QLW*[5] Java-based wrapper for QuantLib
- Apache Hadoop Map-Reduce framework
- Deployment Amazon Elastic Map Reduce cloud servers

We are happy to provide you with complete solutions starting from spreadsheets that you supply or to provide advice and/or software on individual parts of the system. For more information or to discuss your requirements contact us at mailto:webs@bnikolic.co.uk.

[5] http://www.bnikolic.co.uk/ql/qlw.html