

Structured Product Pricing Simulation Using PC Grid

Ong Guan Sin, SCS; Wang Junhong, SVU; Zong Jianping, CFE

Introduction

Financial modeling is one application that benefits substantially from Grid's multi-processing capabilities. Together with the Centre for Financial Engineering, TCG@NUS's project team worked on identifying suitable applications and enabling them on the TCG@NUS. The first application identified was Structured Product Pricing (SPP) - a Monte Carlo-style simulation application which uses the historical performance data of a fixed set of stocks to derive the best option price of structured financial products. Two option prices are produced, based on the American and European models respectively. In enabling the application, the challenge was to maximise the accuracy of the prediction result through higher number of "paths" and "time steps".

Changes to program design and supported platforms

SPP was originally implemented by Jian Ping of CFE using Visual C++. It was designed for interactive use on a Windows PC. In order to grid-enable it, we had to ask the author to adjust the design so that it can be invoked on the command line and its input supplied through command line arguments. This was achieved fairly easily without significant change to the original code.

As TCG@NUS consists of resources available in two OS platforms, namely Windows and Linux, we attempted a build for Linux platform. We successfully expanded the supported platform of SPP to the Linux platform. With this in place, the grid server of TCG@NUS will be able to schedule SPP jobs across grid nodes of different operating systems by automatically selecting the appropriate executable module. This is completely transparently to the user.

Performance

With the executable modules successfully built for the grid, we started testing the performance of the application. In its standalone mode, the maximum number of simulations that a PC can perform is no more than a few hundred thousands. There are two factors causing this restriction: the memory consumption of the SPP process would easily exceed 512MB, and the length of execution would be beyond the acceptable limits.

However, in our tests on TCG@NUS, we pushed the number of simulations to the extreme range of 100 and 300 million. We did this by partitioning the jobs into 2,000 to 10,000 work units. The following test cases were conducted:

Job#	No of Simulations	No of Work units	Nodes Involved	Total CPU Time (day:hr:min:sec)	Elapsed Time
839	300 million	6,000	143	3:05:57:23	1 hr 16 min
840	300 million	6,000	143	3:08:02:29	1 hr 38 min
841*	300 million	6,000	141	3:08:34:20	58 min
842	300 million	6,000	218	3:11:25:34	1 hr 49 min
847	300 million	10,000	174	3:09:16:04	1 hr 49 min

- Job 841 was configured with Concurrent-Dispatches-per-Workunit = 1, whereas the other jobs were configured with Concurrent-Dispatches-per-Workunit = 2. This difference explains the shorter elapsed time. See the Tuning section below for details.
- Job 842 was executed during the day, coinciding with the *dynamic* period of TCG@NUS when user activities on the grid nodes are likely to interfere with resource availability. This explains the longer elapsed time.
- Job 847 was partitioned into smaller work units. There was no gain in elapsed time, probably due to the much larger number of work units and the associated overhead.

Just by measuring total CPU time, it would be impossible to perform the test simulations *reliably* on a PC within the acceptable turnaround time as specified by the user, due to the scale involved (this is without considering the total memory required). The grid resources of TCG@NUS, on the other hand, which is not of guaranteed service level, is proven to be reliable enough for the SPP jobs to be completed within a certain acceptable range.

To test the suitability of running SPP interactively, we ran the following tests:

Job #	No of Simulations	No of Work units	Nodes involved	Total CPU Time (m:s)	Memory consumption	Elapsed Time
Standalone	180,000	N/A	1	N/A	825MB	155 sec
Standalone	190,000	N/A	1	N/A	870MB	166 sec
Standalone	200,000	N/A	1	N/A	910MB	--
848	180,000	6	5	3:30	N/A	92 sec
849	180,000	6	6	2:47	N/A	75 sec
850	200,000	6	6	2:24	N/A	120 sec
851	200,000	6	6	2:52	N/A	155 sec
852	300,000	10	10	4:34	N/A	108 sec

- The tests on the standalone PC were performed on a laptop with a Pentium-M 1.7GHz CPU and 1GB of RAM.
- At 200,000 simulations, the standalone test could not be completed due to excessive process memory requirement.

As can be seen, the grid version of SPP is still able to respond fairly interactively for small jobs. At 300,000 simulations, which are beyond the capability of the standalone PC, the grid version consistently turned in the results, taking about the same length of time as other tests.

Tuning

One of the tuning decisions that we faced was the issue of reliability versus efficiency. Some grid nodes may not be reliable, by the fact that they can be switched off, loaded with user activities or simply malfunction due to local failures. As a result, the reliable completion of a job may be over-dependent on any single grid node. There is one tuning parameter that we used to overcome this: Concurrent-Dispatches-per-Workunit, meaning the number of times a work unit gets dispatched to (therefore get worked on) the grid. In other words, a value of more than 1 provides redundancy and increases reliability. While this makes utilisation of the grid resources less efficient, it was deemed as a worthwhile compromise as the resources are harnessed from idle CPU cycles and are in abundance.

Another tuning decision that we faced was the work unit size. This basically determines the number of simulations each grid node should perform. Should the size be too big and utilise larger memory consumption by the SPP process, some of the grid nodes would not be able to participate in the execution due to the memory constraints. If the size is too small, resulting in too short a turnaround time per work unit, the efficiency will suffer due to overheads involved in work unit dispatches. We eventually concluded that the range of 30,000 and 50,000 simulations per work unit was optimal to cover most of the grid nodes without compromising efficiency.

Conclusion

The results of the enabling exercise show that SPP gains considerably from being grid-enabled and is able to out-perform any other standalone financial modeling applications. With ample resources harnessed on the TCG@NUS, performance gains for any application that can be enabled are almost certain to be considerable.