2008-01-1221

Using Multiple Processors for Monte Carlo Analysis of System Models

Amory Wakefield
The MathWorks, Inc.

ABSTRACT

Model-Based Design has become a standard in the automotive industry. In addition to the well-documented advantages that come from modeling control algorithms, [1,2,3,4] modeling plants can lead to more robust designs. Plant modeling enables engineers to test a controller with multiple plant parameters, and to simulate nominal or ideal values. Modeling variable physical parameters provides a better representation of what can be expected in production. Monte Carlo analysis is a standard method of simulating variability that occurs in real physical parameters. Automotive companies use Monte Carlo testing to ensure high quality, robust designs. Due to time and resource constraints, engineers often examine only a limited number of key parameters rather than an entire set. This leaves the design vulnerable to problems caused by missing the full potential impact of parameters that were unvaried during testing. New high-performance computing tools and multiprocessor machines have eliminated the time and resource limitations in many cases by providing the processing power needed to vary large numbers of parameters in complex dynamic models. This paper presents new methods for distributing Monte Carlo analyses of system models across multiple machines. These methods reduce testing time and enable more complete analyses, ensuring better quality when designs go into production.

INTRODUCTION

Model-Based Design has become a standard for control design. In the automotive industry, using Model-Based Design offers many benefits, including improving the resulting design, shortening development time, reducing costs, and reducing design errors [1,2,3,4]. Model-Based Design enables engineers to gain insight into system behavior without building physical prototypes of the system, which allows this to happen early in the development process.

One common technique for evaluating designs is Monte Carlo analysis. Monte Carlo techniques require simulating the model many times using randomized inputs and parameters. Running the model many times on different test cases provides engineers with insight they can use to refine and improve their design, as well as to verify the design meets the system requirements.

Each of these runs, however, may take hours to complete. To meet the increasing demands of faster product development timelines, speeding up this analysis is critical. The high quality expectations of the automotive market make it impossible to reduce testing time by simply reducing the number of testing scenarios. Finding ways to reduce testing time for all required scenarios is the best alternative.

This paper discusses new techniques to reduce the time needed to complete Monte Carlo testing using software tools for Model-Based Design, high-performance computing (HPC) clusters, and multiprocessor machines. Examples and benchmarks will be discussed that demonstrate the typical increase in performance that can be achieved.

SYSTEM MODELING

The creation of a system model is at the heart of Model-Based Design. A detailed control algorithm model built using commercial-off-the-shelf (COTS) software [5,6] can be simulated in an open-loop manner that provides many, well-documented benefits [2,3]. To further increase the value of the model, engineers can build a plant model, and test the control algorithm in a closed-loop manner.

System models are often created and analyzed under the assumption that the model inputs, defining parameters (constants), and operating environment are known precisely. Real-world systems, however, operate under conditions that are uncertain, and failure to account for that uncertainty can lead to inaccurate predictions of system behavior.

By modeling the whole system, engineers can test multiple scenarios in a simulation environment, including some that may be difficult to reproduce on real systems without damaging the system being tested. For example,

engineers can force system sensors into failure states to ensure the algorithm handles the errors correctly, or they can simulate a system at extreme temperatures. For this testing to be representative of the real system, an accurate plant model is needed.

Another advantage of testing system models, instead of prototypes in a lab, is the elimination of the effort and cost needed to build prototypes, when they are even available. Automotive suppliers often develop plant components and sensors in parallel with the controller design. Parts may not always be available for testing when the control designer is ready to test them. In these cases, developing a plant model that can be simulated with the controller model is necessary to achieve the benefits of system testing in simulation.

PLANT MODELS

Evaluating plant model accuracy and accounting for uncertain model parameters is an important part of the system modeling process. It provides insight into the system's operation and helps ensure that the model is accurate across a range of operating values [7]. One way to determine the accuracy of a model is by comparing test data to simulation data. Automating this comparison using COTS software [8] can simplify this step of system modeling.

Comparing test and simulation data can validate a plant model for a set of ideal or nominal parameters. Plants, however, are built on production lines and almost always contain physical variability. They are also operated in variable conditions, such as temperature, humidity, and electrical noise. The ability to simulate a system across multiple scenarios can greatly increase confidence in a model.

MONTE CARLO TECHNIQUES

Monte Carlo methods are useful techniques for simulating physical variability and electrical noise, because these variables can be modeled using Gaussian and other probability distributions. The inputs to these statistical models can be obtained from manufacturers' specifications of physical parts, lab measurements, test specifications, or historical data.

Whether conducting manual random testing or applying sophisticated Monte Carlo techniques, the more random values tested, the more valuable the testing is at predicting real performance. In practice, this can mean testing thousands of scenarios to cover a design space. One technique used to reduce the number of test cases needed is Design of Experiments. While this can greatly reduce the number of tests needed, running 1000 instead of 5000 scenarios can still take a long time, particularly as the fidelity of a model increases.

Reducing the time needed for Monte Carlo analysis can enable engineers to fully test designs in a modeling environment without sacrificing product schedules. That, in turn, leads to higher quality designs and fewer problems when the design goes into production.

HIGH PERFORMANCE COMPUTING

With the increased availability of multicore and multiprocessor computers, and the growth of computing clusters, many organizations already have vast CPU processing power installed on site. Though the processing power is available, running an individual engineer's simulations on the cluster or on multiple processors is not always straightforward, particularly from within a COTS simulation software tool. In fact, the difficulty in setting up distributed applications is still a barrier to tapping the processing power in one computer, let alone a cluster of perhaps hundreds. Yet, taking advantage of these computing resources is a critical step to increasing the number of Monte Carlo simulations that can be reasonably run.

Running simulations in a HPC environment is one approach that was once prohibitively expensive due to the cost of installing and maintaining sufficient computing power. HPC was available mainly to government agencies and large research laboratories, because only these groups had the means to purchase supercomputers. Today, most supercomputers have been replaced by COTS computer clusters that provide affordable, high-performance, distributed computing environments. The number of available clusters is growing rapidly. Last year, HPC hardware revenue hit an all-time high of $10 billion and revenue has been growing more than 20% over the last four years [9].

DISTRIBUTING SIMULATIONS

Certain computing problems are classified as distributed or coarse-grained because they can be segmented easily to run on several nodes without communication, shared data, or synchronization points between the nodes. Monte Carlo simulations fall into this category, and thus are an obvious candidate to run on computing clusters.

Some organizations have found that the time needed to setup and maintain their COTS simulation software on a cluster substantially reduces the time savings provided by the additional computing power [10]. In these situations, integration between the simulation software and the HPC cluster is needed to maximize the investment in computing power.

EXECUTABLES

One way to run simulations on a cluster without installing COTS software on every node is to compile the model into an executable [11]. For example, one can generate ANSI/ISO C/C++ code from the model, and then compile this code into an executable. The separate executable can be executed from a command line, thus requiring scripts to provide input data to the executable and handle its output data.

While this technique is very useful for accelerating simulations, to use it to run Monte Carlo analyses, the engineer must write scripts that set up parameter variations and gather results from the multiple executables into a usable format. If the engineer is using a remote cluster, additional scripting may be needed for communication with a scheduler and job creation. This can be tedious and often requires help from a cluster administrator to customize a particular simulation setup. The technique described below and used throughout the remainder of the paper addresses this barrier to using HPC resources for Monte Carlo simulations.

DISTRIBUTED COMPUTING TOOLS

An alternative to using executables is to use COTS tools that provide an interface to the computer clusters. For example, with SystemTest™ [8] and Distributed Computing Toolbox™ [13] engineers can use a Graphical User Interface (GUI) to vary a set of parameters in a Simulink® model for Monte Carlo testing and then define it as a job to be run on multiple processors. With these COTS solutions, engineers can launch the job by simply checking a box within the GUI. An engineer using SystemTest can distribute complete Simulink models for execution in a cluster or in a multicore or multiprocessor computer without writing any lines of code [8,12].

Additional time can be saved if the COTS simulation tools are integrated with HPC schedules. For example, a system or cluster administrator can setup a configuration file that contains information about the particular scheduler in the cluster and the shared directories that may be needed. After the file is made available to anyone who wants to use the cluster, it can be used by programs such as the MATLAB® Distributed Computing Engine [12] to schedule and execute jobs on the available workers. This technique is examined in the following DC motor model example.

DC MOTOR MODEL EXAMPLE

A DC motor model was previously described by Kozola and Doherty [14]. We use a similar model, shown in Figure 1, as an example with a Monte Carlo test built around it.

The DC motor model has two inputs - the supply voltage (Vs) and the inertial load (Jd) - and six adjustable model parameters, as shown in Figure 2. These parameters, which were defined based upon manufacturer specifications for the motor, are assumed to be constant in the model. The output characteristics of interest are the rise time and the steady-state angular velocity.

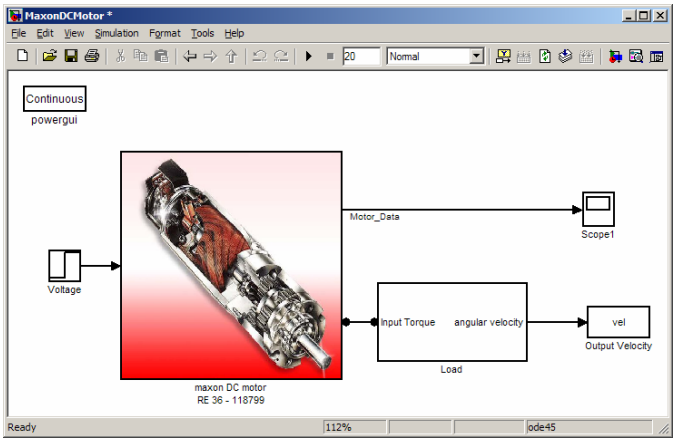

Figure 1. DC motor model.

MONTE CARLO SIMULATIONS

We set up a test to evaluate performance that engineers might expect to see in a real DC motor due to manufacturing variations. Iterations of the model were run with 1000 different sets of system parameters. These parameters were generated using normal (Gaussian) and uniform probability distributions. Additionally, the input voltage signal was varied across a range of expected operational values.

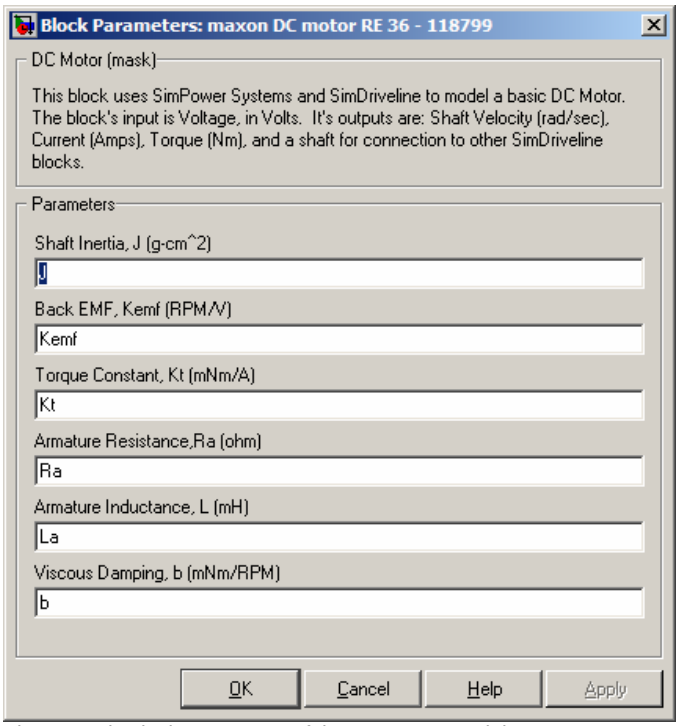

Figure 2. Physical parameters of the DC motor model.

We used SystemTest to read experimental test data from an Excel® file. Gathered during tests of real DC motors across a representative set of operating conditions, these experimental test data provide real-world response data. These data were used to determine the model's accuracy when predicting performance variability of physical motors. A Monte Carlo simulation was then performed, running the model repeatedly with 1000 random combinations of parameter values.

We performed a curve fit on the experimental test data, to determine the output characteristics for the operating conditions used in the Monte Carlo testing. We evaluated the model's accuracy for predicting performance variability by comparing the expected output characteristics derived from the experimental test data to the output characteristics produced from the Monte Carlo simulations. All of this was automated in a SystemTest TEST-file.

In this paper, we are not as concerned with the accuracy of our model as we are with the time it takes to perform the Monte Carlo tests, compare each output to experimental data, and report the results. We first ran the model on a single processor, which took almost two hours (6600 seconds). We then ran the same Monte Carlo test using a computing cluster.

BENCHMARKING ON A COMPUTING CLUSTER

The number of workers used for a specific test run was selected using the graphical configuration tool provided to manage worker configurations. Each 1000-iteration test was identical; only the number of workers used varied from run to run. Each scenario was run four times and the times were averaged to obtain the results shown in Figure 3.

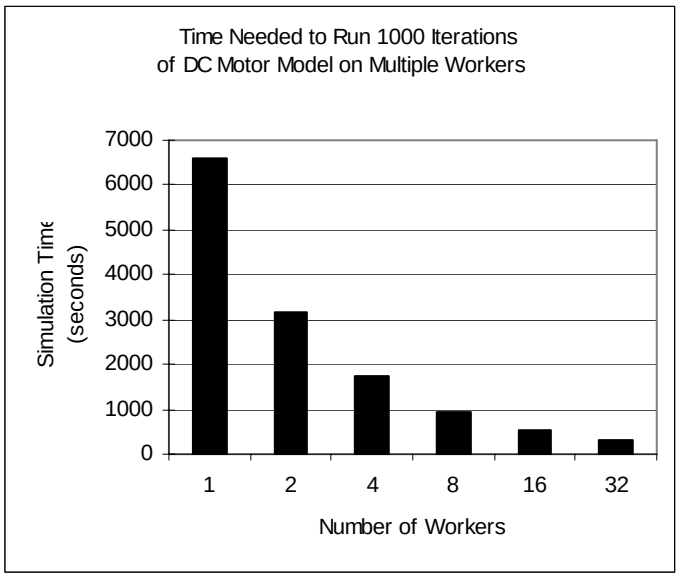

Figure 3. Graph showing the time savings when running Monte Carlo example on multiple processors.

The dedicated cluster of machines used for this example consists of 8 machines with Quad-Core AMD Opteron™ processors running at 2.4 GHz. The cluster is homogenous, meaning that each of the quad-core machines ran the Debian 4.0 Linux distribution, shared 4GB of RAM and communicated with the other machines using a Gigabit Ethernet network.

As shown in Figure 3, the speed improvement is not precisely linearly dependent on the number of workers. This is because distributing the tasks adds its own overhead, which includes copying over the files from the client machine to each machine on the cluster and transmitting input and output data between the workers and the job manager and the network latency. In general, total simulation time should be significantly longer than the time needed for these overhead tasks to achieve speed benefits from distributing tasks.

CONCLUSION

An accurate, production-representative plant model that can be simulated in the same environment as an embedded controller model can significantly increase the quality of automotive systems. This paper has presented some of the hurdles that must be overcome to create and validate of accurate system models. We presented a concrete example of one technique, Monte Carlo testing, to validate plant models.

One drawback to Monte Carlo testing is the number of simulations needed to create meaningful data and the time needed to complete those simulations. Often, product delivery schedules do not allow enough time to perform more complete testing in simulation before building hardware.

One solution to this issue is to reduce the time needed to run multiple simulation iterations. With the rapid growth of the HPC market and increasing availability of computing resources to individual engineers, an attractive option is to take advantage of the multicore machines and computing clusters that are installed in many organizations today. We presented tools from The MathWorks that enable engineers to make use of these resources without needing to redesign tests or write additional scripts.

The benchmarking clearly demonstrated that significant time savings is achievable using existing tests, even with a modestly sized computing cluster. Engineers are no longer forced to choose between meeting a production schedule or fully testing a design. Using a combination of distributed computing tools, SystemTest and HPC resources, engineers can simultaneously address two opposing demands that face the automotive industry today: time-to-market and quality.

REFERENCES

1. Smith, Paul F., Sameer M. Prabhu, and Jonathan H. Friedman, "Best Practices for Establishing a Model-Based Design Culture," *Systems Engineering*, 2007 SAE World Congress, Detroit, Michigan, April 2007.
2. Tung, Jim, "Using model-based design to test auto embedded software," *EE Times*, 09/24/2007, http://www.eetimes.com/showArticle.jhtml?articleID=202100792 (accessed Sept. 24, 2007).
3. Thate, Jeffrey M., Larry E. Kendrick, and Siva Nadarajah, "Caterpillar Automatic Code Generation," *Electronic Engine Controls*, 2004 SAE World Congress, Detroit, Michigan, March 8-11, 2004.

4. Hodge, Grantley, Jian Ye, and Walt Stuart, “Multi-Target Modelling for Embedded Software Development for Automotive Applications,” *In-Vehicle Networks and Software, Electrical Wiring Harnesses, and Electronics and Systems Reliability,* 2004 SAE World Congress, Detroit, Michigan, March 8-11, 2004.
5. “MATLAB® User’s Guide,” The MathWorks, Natick, MA, September 2007.
6. “Simulink® User’s Guide,” The MathWorks, Natick, MA, September 2007.
7. Wood, G. D., and D. C. Kennedy, “Simulating Mechanical Systems in Simulink® and SimMechanics,” Technical Report 91124v00, The MathWorks, Inc., Natick, MA, 2003.
8. “SystemTest User’s Guide”, The MathWorks, Natick, MA, September 2007.
9. IDC HPC Briefing, International Supercomputing Conference, Dresden, Germany, June 26-29, 2007.
10. Hutton, Clifford, “HPC in the Kitchen and Laundry Room: Optimizing Everyday Appliances for Customer Satisfaction and Market Share,” SC07, Reno, Nevada, November 11-16, 2007.
11. “Real-Time Workshop® User’s Guide”, The MathWorks, Natick, MA, September 2007.
12. Ghidella, J., A. Wakefield, S. Grad-Freilich, J. Friedman, and V. Cherian, “The Use of Computing Clusters and Automatic Code Generation to Speed Up Simulation Tasks,” AIAA Modeling and Simulation Technologies Conference and Exhibit, Hilton Head, South Carolina, Aug. 20-23, 2007.
13. “Distributed Computing Toolbox User’s Guide”, The MathWorks, Natick, MA, September 2007.
14. “MATLAB® Distributed Computing Engine System Administrator's Guide”, The MathWorks, Natick, MA, September 2007.
15. Kozola, Stuart and Dan Doherty, "Using Statistics to Analyze Uncertainty in System Models," *MATLAB Digest*, May 2007, http://www.mathworks.com/company/newsletters/digest/2007/may/uncertainity.html.

ADDITIONAL SOURCES

Automotive Engineering International, March 2005.

Martinez, Wendy L., and Angel R. Martinez. Computational Statistics Handbook with MATLAB®. Chapman & Hall/CRC, 2002.

Morgan, Byron J. T. Applied Stochastic Modelling. Arnold, 2000.

Robert, Christian P., and George Casella. Monte Carlo Statistical Methods. Springer, 2004.