Comparison of Bayesian Networks and Data Mining for Coverage Directed Verification
Category Simulation-Based Verification*

Markus Braun†
STZ Softwaretechnik
Esslingen, Germany
email: braun@stz-softwaretechnik.de

Wolfgang Rosenstiel
University of Tübingen
Tübingen, Germany
email: rosenstiel@informatik.uni-tuebingen.de

Klaus-Dieter Schubert
IBM Germany Development
Böblingen, Germany
email: kdschube@de.ibm.com

1 Abstract

Today directed random simulation is one of the most commonly used verification techniques. Because this technique in no proof of correctness, it is important to test the design as complete as possible. But this is a hard to reach goal, that needs a lot of computing power and much human interaction. There had been a proposal for using Bayesian Networks to implement an automatic feedback loop [5]. In addition, this paper will introduce another implementation of an automatic feedback loop using Data Mining techniques. Both approaches are applied to the same design and the results will be compared.

2 Introduction

Today directed random simulation [2] is one of the most commonly used verification techniques. With directed random simulation, the test vectors and the sequence of test vectors for the device under test (DUT), are random. The test generation can be directed through various parameters. They change the weight for different aspects of the test generation.

To ensure simulation quality, it is essential to measure the progress of the verification during directed random simulation. For measuring the progress a coverage model is used. The coverage model is the Cartesian product of observed signals. Based on this Cartesian product, subsets can be defined, that represent valid signal combinations that may occur during the verification process or which are invalid and must not occur. Each of the valid signal combinations is called a coverage task.

During simulation the DUT is observed by a monitor, which feeds its observations to a coverage tool, that maintains the coverage of all simulations done so far. In combination with the defined coverage model, it is possible to identify yet uncovered tasks. After some time, directed random simulation covers a lot of tasks, but usually the hard to reach tasks remain uncovered or are covered very rarely. To hit those uncovered tasks, a verification engineer needs to modify the parameters of the test generator, so that it will generate the missing test vectors for the uncovered tasks. Here a permanent human interaction is necessary to tweak the parameters, run simulations, check the results and tweak the parameters again.

Another approach could be the introduction of an intelligent agent, that automates the tweak – simulate – check cycle. This intelligent agent includes a model of the random simulation process. The model encodes information about the coherences between the parameters for the test generator and the resulting coverage. This model is used to identify the sensitive parameters and their needed values to match a hard to reach coverage task. Using this model, previous uncovered tasks can be covered and previous covered tasks can be hit easier. So it is not only possible to hit all coverage tasks, it is also possible to compile an efficient regression suite to quickly verify design changes. Figure 1 shows the general structure of this approach.

There are different possibilities to build a model of the random simulation process needed by the intelligent agent:

- Using observed simulation data
  A set of simulations are done with varying parameter sets. This simulation data is used to build a behavior model of the random simulation process.

- Using domain knowledge
  Here the model is built using deep domain knowledge about the random simulation process.

- Using a combination of observed simulation data and domain knowledge
  This is a combination of the first and the second possibility. Here the model is built with some domain knowledge and then adapted with observed simulation data.

*Copyright © 2003 IEEE. Reprinted from proceedings of the HLDVT’03 conference. This material is posted here with permission of the IEEE. Such permission of the IEEE does not in any way imply IEEE endorsement of any of HLDVT products or services. Internal or personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution must be obtained from the IEEE by writing to pubs-permissions@ieee.org. By choosing to view this document, you agree to all provisions of the copyright laws protecting it.

†The work described in this paper had been done on behalf of IBM Development, Böblingen, Germany in cooperation with University of Tübingen, Germany.
The Data Mining approach described later in this paper implements the possibility of only using observed simulation data. With Bayesian Networks, it is possible to implement all of these possibilities. The Bayesian Networks experiment described in this paper, uses a combination of observed simulation data and domain knowledge.

3 The hardware design used for the experiments

For the experiments described in this paper, the NorthStar design had been used. It was designed by IBM Research, Haifa, Israel and was also used for another coverage directed verification experiment [5], that had the goal to generate short instruction sequences for coverage tasks. The goal of the experiments in this paper had been to increase the occurrence of a coverage task in a fixed length simulation of 10 cycles.

The design consists of a dispatcher and different pipelines. Figure 2 shows the block diagram.

For the following experiments only two pipelines of the NorthStar design had been used. The first one is a simple arithmetic pipeline the second one is a complex arithmetic pipeline. For these pipelines the dispatcher can handle five types of commands. The simple arithmetic pipeline is able to handle two commands, the complex arithmetic pipeline can handle all five commands. The commands operate on eight registers.

The following informations are available during the simulation for building a coverage model:

- The command type in stage 0 of the simple and complex arithmetic pipeline.
- The usage of stage 1 of the simple and complex arithmetic pipeline.

4 Coverage directed verification using Data Mining techniques

In this experiment the model in the feedback loop had been generated with Data Mining techniques only by using observed simulation data. To build the model the Tree Classification algorithm [4] of the IBM Intelligent Miner for data [1] had been used. With this algorithm, rules can be extracted out of the data for each coverage task. These rules restrict simulation parameters to subranges and predict the number of occurrences of this coverage task, when a parameter set matches a rule.

The behavior model, that describes the random simulation process, is not built using knowledge about internals of the process. The only information used for building the model is the observed behavior during simulations done with various parameter sets. Hence the quality of the model depends on the quality of the observed behaviour.

The Data Mining approach for building the behavior model and the feedback loop consists of three general steps:

- Run several simulations with different parameters to gather statistical data about the coherences between parameters and coverage tasks. Later on, this will be referenced as sample data.
- Run the Data Mining algorithm on the sample data to extract the coherences.
- Apply the resulting rules to the parameters.
- Run the simulation with the modified parameters.
There are different ways to generate the sample data for the Data Mining process. The simplest method is to run simulations by random and collect the used parameters and the occurred coverage tasks for each run. For this, the simulator must be capable of supplying the information about the used parameters. An alternate possibility is, to generate different parameter files offline and use these parameter files for generating the sample data. This option had been used in this experiment. The same sample data had also been used for the later described Bayesian Network experiment to get comparable results.

The sample data consist of 1144 simulations each of an instruction sequence length of 10. Each simulation run consists of the parameter set used for that simulation run and a histogram of occurred coverage tasks. For each of the 54 coverage tasks the behavior model is built, the rule with the highest number of occurrences is extracted. After the rule generation, for each coverage task, the rule is applied to the parameters and a simulation run is done.

If the predicted parameter sets don’t generate the expected coverage task, there are different possibilities that may lead to better predictions and better parameter sets.

One possibility is to use the simulation data of wrong predictions as additional sample data for the classification algorithm. This hadn’t worked for this experiment.

Another possibility to improve the results is, to cluster different similar coverage tasks to get a better predictions. Similar coverage tasks can be tasks that are caused by similar test vectors or tasks that differ only in less important portions of the signal combinations, that define those tasks. But even with clustered coverage tasks, it had not been possible to improve results on this design.

Figure 3 shows the resulting changes of the coverage tasks compared to random simulation with unified distribution over the parameters. The dotted line at zero marks the average value of about 5000 random simulations, each 10 cycle long. The histogram bars show the change of the coverage task as factor.

The results show, that the Data Mining approach is capable of predicting parameter sets, that increase the occurrence of different coverage tasks in a significant way. It increased the occurrence of coverage tasks in average by a factor of 3.1. The first coverage tasks shows a extreme improvement. This could be explained with the simplicity of this task. It occurs, if the dispatcher only dispatches NOP commands. The coverage task at the left-most position is very easy to hit and the Data Mining approach predicts a very efficient parameter set for this task. The figure also shows, that in some cases the occurrence of coverage tasks gets lower, than in the random simulation. But for this hardware design, the classification algorithm could not predict parameter sets for rare coverage tasks. Even the mentioned methods to improve the quality of the prediction had no significant influence. Nevertheless this approach didn’t used domain knowledge, beside of the clustering of the coverage tasks.

5 Coverage directed verification using Bayesian Networks

5.1 What are Bayesian Networks

Bayesian networks [3], also called belief networks, are directed acyclic graphs. The nodes of this graph represent the random variables of the problem domain. The arcs between the nodes represent the direct dependencies between the corresponding variables. Each node of this graph needs a conditional probabilistic distribution, where the variable in question is conditioned by its predecessors in the network.

The Bayesian Network approach for coverage directed verification had been developed by IBM Haifa research lab, Israel [5].

A Bayesian Network can be built in different ways. One possibility is to define the structure of the network as well as the probabilities by hand. Another possibility is to learn either the probabilities or structure and probabilities from observed data. In the following experiment the structure of the Bayesian Network is built by hand and the probabilities are learned from observed data.

5.2 Implementation of the feedback loop

The Bayesian Network used in this approach encodes a behaviour model of the random simulation process. This model had been build using observed simulation data and some domain knowledge about the process. In concrete that means, that the structure of the Bayesian Network had been build using domain knowledge. Figure 4 shows the network. The top layer of the network consists of the simulation parameters, the bottom layer of the coverage events. Both layers are connected through two hidden nodes. Hidden nodes are nodes, that are not observable during simulation, but that make sense for the structure of the network. These two nodes encode both modes of the dispatcher. Meaning, dispatching an instruction to the simple arithmetic pipeline or to the complex arithmetic pipeline. To learn the probabilities of the network, the same sample data as for the Data Mining experiment have been used to get comparable results between both approaches.

Each of the 54 coverage tasks is put into the Bayesian Network as evidence. Then the marginals for the simulation parameter nodes are calculated and are used as parameter sets for the simulator.

If the parameter sets don’t fulfill the expected result, the Bayesian Network can be improved in different ways.
One possibility is to enlarge the sample data and use this for retraining and adaptation. This can be done for example by rethinking the samples generation and add some new samples that haven’t been included in the first set. Another way to add new samples is, to use some of the samples generated with the predicted weights during the evaluation of the network. The predictions can be improved by clustering different similar coverage events together. This enlarges the number of sample data sets on which this prediction is based, thus the prediction is more confident.

A third possibility is to rework the layout of the Bayesian Network by adding or removing hidden nodes or by changing the connections between the nodes. This can be necessary, if the initial Bayesian Network missed a design aspect or it had been to detailed.

Figure 5 shows the results of the simulations done with predicted parameter sets. The coverage tasks in this figure are in the same order as in figure 4. Here also it is clearly visible, that the Bayesian Network approach is capable to direct the simulation. It is even capable to generate coverage tasks, that didn’t occur in the sample data. For this experiment the Bayesian Network approach results in a factor 6.3 increase of the coverage tasks in average.

6 Comparison

An advantage of the Data Mining approach is, that it doesn’t need much domain knowledge about the design. It extracts all its information from the observed behaviour of the random simulation process. Thus this approach can be implemented as a fully automated system, that observes random simulation for some time, builds the behaviour model and closes the feedback loop. The drawback of this approach is, that the predicted parameter sets for rare coverage tasks don’t generate these tasks in all cases. If a specific coverage task occurs only a few times in the sample data, the predicted parameter set is likely not capable of producing this task. That means, that this approach has the tendency to not improve the situation in an interesting part of the verification process.

The Bayesian Network provides a great freedom in building the behaviour model for the feedback loop. It is possible to build the Bayesian Network only using observed simulation data, only using domain knowledge or any combination of both. Because domain knowledge is already available in the verification team, it is a major advantage of the Bayesian Network, that it possible to use this knowledge and to improve the quality of the model this way. A problem could be, that there are very few verification engineers, that are able to construct the Bayesian Networks themselves. Thus there must be either some education effort for verification engineers or a Bayesian Network expert, who creates the networks based on the information he gets from the verification team. One of the most interesting aspects of the Bayesian Network approach is the capability of predicting parameter sets for rare coverage tasks and even for coverage tasks, that haven’t occurred in the sample data.

As the experiments show, with both approaches it is possible to tune the parameters to increase the number of each coverage task. But the Bayesian Network approach has a more than twice as good performance compared to the Data Mining approach. The Bayesian Network is capable of predicting parameter sets for infrequent coverage tasks and even for coverage tasks that didn’t occur in the sample data at all. The Data Mining approach has here its weakness.

As second comparison of the approaches, both had been used to generate 100% coverage as fast as possible. To achieve this, each model had been used to predict simulation parameters for one uncovered task. These parameters had been used for a simulation and all occurred coverage tasks had been added to the coverage. After that, each model had been used to predict simulation parameters for the next uncovered task. This procedure had been repeated until either 100% coverage had been generated or simulation parameters for all uncovered tasks had been predicted.

Figure 5 shows, that the Bayesian Network reaches 100% coverage very fast. Each simulation covered the coverage task it was predicted for and in most cases other, yet uncovered, tasks. Because of that it was possible to cover all task after 22 simulations.

With the Data Mining approach it was not possible to cover the whole space. There had been 7 coverage tasks left, for with the Data Mining approach was not able to predict simulation parameters that cover these tasks. Never the less, the progress of the Data Mining was faster than the progress of pure random simulation. At least this could be used to gain some speed in the coverage process. When the Data Mining approach get stuck,
the verification could be continued using random simulations. With Data Mining a coverage level of 47 tasks was reached after 26 simulations. Random simulations reach this level after 140 simulations.

Figure 6: Results of a regression run with Bayesian Networks and Data Mining approach NorthStar design

7 Conclusion

This paper compares the both approaches only by using a small hardware design. To get more confidence in the comparison of the Data Mining and the Bayesian Network approach, a Storage Controller experiment [5], that had been used to prove the Bayesian Network approach, is conducted at the moment using the Data Mining approach. This is work in progress, but first results indicate, that the comparison in the previous section is also valid for the Storage Controller experiment.

References


