

# The Effects of Sampling Kurtosis on Anytime Algorithm Performance

**Scott A. Burgess**

Department of Computing Science  
Humboldt State University  
Arcata, CA 95521

## Abstract

*Agent-system and agent-simulator networks encompass complex interactions that are often difficult to model. We use an On-Line Maintenance Agent test bed for analyzing the behavior of Bayesian anytime inference algorithms for influence diagrams. This test bed is used to analyze the performance of a backward simulation algorithm used by the agent for decision making. We show that the kurtosis of sampling acts as a focusing mechanism which may be useful in designing better anytime algorithms.*

Keywords: POMDP, agent, anytime inference algorithm, sampling.

## 1. Introduction

*Resource-bounded* inference [7] confronts two major limitations to building better rational agents: limited information, which typically takes the form of incomplete information, and finite computational resources. Sometimes modeled as Partially Observable Markov Decision Processes (POMDPs), theoretical results suggest that it is unlikely we will discover computationally fast and complete solutions in these domains since current POMDP formulations are PSPACE-Complete [8]. And since agents are often embedded in a changing environment, exact solutions to POMDPs are often computationally impractical anyway.

My recent work in this area centers on the On-Line Maintenance Agent (OLMA) test bed, a repair scenario capable of analyzing performance of anytime inference algorithms used by an agent performing maintenance activities on electrical circuits. Bruce D'Ambrosio and I previously compared several Bayesian anytime inference algorithms via a test bed simulating a partially observable diagnostic task in an effort to understand what reasoning algorithms performed better and why [4]. Further analysis led to the use of Stochastic Automata Networks to model the agent-simulator interactions [1].

This short paper describes experiments on the OLMA with the backward simulation algorithm. While this algorithm's performance is significantly weaker than other algorithms we have studied, we have discovered that performance of the algorithm can be skewed by altering the sampling algorithm used in backward simulation.

## 2. The OLMA

Originally developed to explore compilation of reactive solutions and automatic construction of reactive controllers [7, 9], the OLMA has evolved into a laboratory for studying Incremental Probabilistic Inference [3], value-drive diagnosis [2], and comparisons of anytime decision algorithms. The OLMA can be broadly described as follows:

- The OLMA is an embedded diagnostic domain comprising an *agent* and a *simulator* that share information but otherwise are treated as separate processes. This allows the agent to solve different problems, and different kinds of agents to solve the same problem.
- The simulator, for purposes of this paper, mimics a half-adder with several gate components, each of which may independently fail during a time step with some small probability.
- The agent may perform both repair and sensing actions on the simulator, and actions have costs.
- The agent also incurs costs for each time cycle the simulator fails to function properly, and it therefore has incentive to reason quickly and act in order to minimize costs.
- The performance metric is the agent's long-term costs for maintaining the simulator; the gold standard is minimum long-term cost.

The challenge for the agent is to accurately diagnose and repair the circuit in the simulator while fighting time constraints on deliberation. This is in contrast to other formulations where diagnosis is a static activity in which an optimal or near-optimal action is chosen without regard to costs imposed by the passage of time.

Earlier papers [1, 4] describe the OLMA in much greater detail.

### 3. Backward Simulation

The backward simulation algorithm [5] was developed as a candidate reasoning algorithm for general influence diagrams. Since the number of samples used may be altered, the algorithm has great flexibility as an anytime algorithm.

Earlier experiments with backward simulation were disappointing. While flexibility is desirable, the quality of inference and time needed to gather sufficient samples were both inadequate in the OLMA. Thus the algorithm was discarded from our previous experiments at an early stage.

We did wish to explain the performance of algorithms that did perform well. We hypothesized that anytime algorithms that performed well tended to sample probability from only high probability mass regions of the influence diagram. The structure of the other algorithms prohibits analysis of performance along this dimension since it cannot be separated from other factors. Backward simulation allows us to alter where sampling is done without affecting other aspects of the algorithm. Hence we can isolate and study how a more focused sampling affects agent behavior.

### 4. Skewed Backward Simulation

Backward simulation provided a relatively simple algorithm, but how do we modify it to sample high probability mass zones of the distributions more heavily? Table 1 illustrates the effects of squaring and renormalizing a probability distribution. The variable A may be in states 1, 2, and 3 with the probabilities 0.6, 0.25 and 0.15 respectively. If we square each of these values, we obtain the values in the following row, which cannot constitute a probability distribution since they do not sum to one. But the values may easily be renormalized to generate the third row, which is a new probability distribution associating different values with each state of variable A.

**Table 1: An Example of Peaking a Distribution with Logic Sampling**

	P(A=1)	P(A=2)	P(A=3)	SP(A=i)
<b>Original probability distribution:</b>	0.6	0.25	0.15	1.0
<b>Elements above squared:</b>	0.36	0.0625	0.0225	0.445
<b>Previous row renormalized:</b>	0.8+	0.14+	0.05+	1.0

We notice immediately that if we sample according to the renormalized squaring instead of the original probability distribution, more samples will be allocated to state 1 at the expense of states 2 and 3. Such a sample does not give a true picture of the probability distribution, but this can be fixed: if after sampling 100 times we note that 81 samples were allocated to state 1, then we can recover an estimate of the true distribution for state 1 by multiplying by 0.6/0.8, the ratio of the original distribution to the peaked distribution for that state. This yields an estimate of 60.75 samples for state 1, or an estimated probability of 0.6075 for that state.

By applying this peaking technique to backward simulation, we still obtain an accurate estimate of the probability distributions but with more samples than usual allocated to the high probability mass zones. My hope was that this would give a more accurate picture of these critical zones and thus improve the performance of stochastic simulation when only a small number of samples can be allocated. Of course, one is not limited to taking the square of the probability values, and I have termed the power that the values are raised to the *kurtosis value* for that sampling.

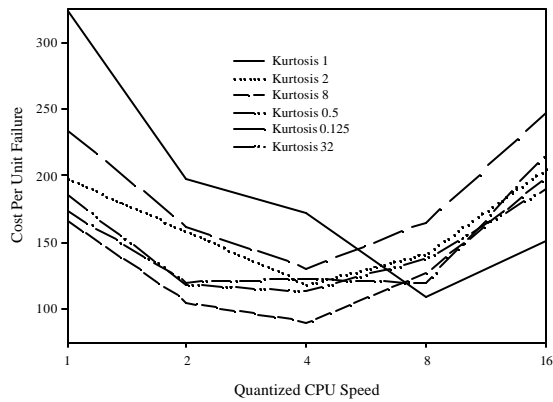
I decided to try a number of different kurtosis values to see if “more is better.” I felt that in the limit all samples would be allocated to just the highest probability mass(es) in the distribution, which gives a poor estimate of the complete distribution. Thus if peaking improves stochastic simulation with a small number of samples, it was also likely that it did so only within a range of kurtosis values that depended somehow on the total number of samples. I decided also to test fractional kurtosis values, since those should exhibit the opposite effect of peaking with values greater than one.

Note that I am taking liberty in calling the sample skewing factor a “kurtosis value,” as this is not standard statistical terminology except in spirit.

### 5. Results

It is clearly impossible to model the reasoning processes of the agents through SANs. However, there are two practical uses we can put our OLMA models to. First, the Random agent can be analyzed because it represents equiprobable selection of any agent action. Second, it might be possible to construct an ideal agent, one which always selects a correct repair action if one is available.

The results are summarized in Figures 1 and 2: Figure 1 presents results using 1000 samples, and Figure 2 presents results using 5000 samples. We expect higher quality decisions under the latter regimen, though the added time may increase rather than decrease costs.



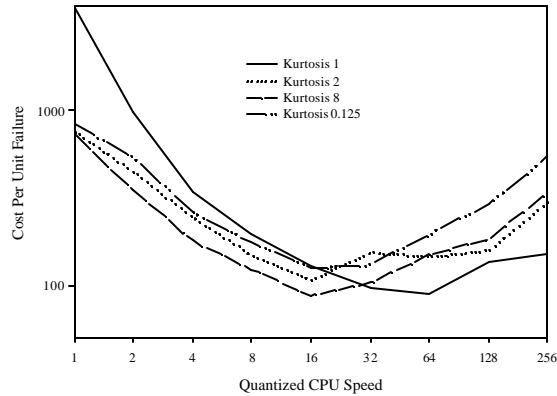
**Figure 1: Critical Region of the Averaged Performance Data for the Backward Simulation Algorithm with 1000 Samples at Different Kurtosis Values**

Studying these graphs carefully, I concluded there were two regions of interest. The first is the downward slope on the left where cost per unit failure is improving. The second is a rise in the cost per unit failure beginning in the middle of the graph. Beyond a certain quantized CPU speed the graph levels off due to synchronizing behaviors in the simulation that were revealed by my stochastic automata network model of the OLMA [1].

The most important point for each kurtosis value is where the cost per unit failure is minimum. I note two interesting phenomena in the graph. First, some data minima show performance differences with respect to the baseline, Kurtosis 1. The minimum value for Kurtosis 8 is lower than the minimum for Kurtosis 1. Applying the Wilcoxon rank-sum test for the data establishing these averages I found that the difference is statistically significant at the level 0.025 (Table A.17 in [10] was used in making this determination). The minimum value for Kurtosis 0.125 is higher than the minimum for Kurtosis 1 at the same level of significance. The minima for Kurtosis 2 and Kurtosis 0.5 could not be statistically separated from the Kurtosis 1 minimum with the given data. The minimum for Kurtosis 32 also could not be separated statistically from that of Kurtosis 1, but one data point in the sample set for the minimum of Kurtosis 32 is an extreme outlier that I elected not to discard since the number of simulator faults appeared to be sufficient to permit it. Ignoring that outlier, Kurtosis 32 would also give performance superior to Kurtosis 1.

The second phenomenon worth noting in the graph is that all peaked samples appear to have their minima shifted toward lower quantized CPU values.

This is useful in one sense: minima that occur at lower quantized CPU speeds yield better performance precisely where the agent is pressed hardest under the time/quality tradeoff.



**Figure 2: Average Performances of Backward Simulation with 5000 Samples at Different Kurtosis Values**

The data for samples of size 5000 are presented in Figure 2. Since collecting a complete dataset for 5000 samples could have taken considerable time, I chose to concentrate on values that appeared important in the 1000 sample case. Parallels exist between the results. Again I observed that all peaked runs appeared to have their minima at lower quantized CPU speeds than Kurtosis 1. And again the Kurtosis 0.125 had a minimum considerably higher than the minimum of Kurtosis 1. Unlike the 1000 sample runs none of the different kurtosis values tried yielded statistically significant performance improvements in an absolute sense, though again it is worth noting that the leftward shift could constitute a performance improvement.

## 6. Conclusions

Clearly our initial hypothesis appears to be correct: sampling that emphasizes high probability outcomes using backward simulation improves performance. But this performance improvement has limitations. It appears that the performance improvements are greater when the sampling regimen is most sparse. This suggests that under tight time constraints, the agent should compare its current belief state with the most likely outcomes, choosing accordingly. As time to reason increases, the payoff for this strategy decreases. Further research is needed to explore the boundaries and limitations of this approach.

With this in mind, we can design other anytime algorithms with better performance by emphasizing high probability outcomes.

## References

- [1] Burgess, Scott. "Stochastic Automata Network Models of Agents," in *IC-AI '04, Proceedings of the International Conference on Artificial Intelligence*, edited by Hamid R. Arabnia *et al.* Las Vegas, NV : CSREA Press, 2004.
- [2] D'Ambrosio, Bruce. "Real-time Value-driven Diagnosis," in *Proceedings of the 3<sup>rd</sup> International Workshop on the Principles of Diagnosis*, 1992.
- [3] D'Ambrosio, Bruce. "Incremental Probabilistic Inference," in *Uncertainty in Artificial Intelligence: Proceedings of the Ninth Conference*. San Mateo, CA: Morgan Kaufmann Publishers, Inc., 1993, pp 301-308.
- [4] D'Ambrosio, Bruce and Scott Burgess. "Some Experiments with Real-time Decision Algorithms," in *Uncertainty in Artificial Intelligence: Proceedings of the Twelfth Conference*, pp 194-202.
- [5] [Fung 94] Fung, Robert and Brendan Del Favaro. "Backward Simulation in Bayesian Networks," in *Uncertainty in Artificial Intelligence: Proceedings of the Tenth Conference*. San Mateo, CA: Morgan Kaufmann Publishers, Inc., 1994, pp 227-234.
- [6] Horvitz, Eric J. "Reasoning Under Varying and Uncertain Resource Constraints," in *Proceedings AAAI-88: Seventh National Conference on Artificial Intelligence*, pp 111-116.
- [7] Kaul, Lothar. *A Feasibility Study on Compiling Reactive Problem Solution Methods for an AI Domain*. Master of Science Thesis, Department of Computer Science, Oregon State University, 1991.
- [8] Papadimitriou, Christos H. and John N. Tsitsiklis. "The Complexity of Markov Decision Processes," in *Mathematics of Operations Research* **12** (3): 441-450.
- [9] Westerberg, Caryl J. *Investigation of Automatic Construction of Reactive Controllers*. Master of Science Thesis, Department of Computer Science, Oregon State University, 1991.
- [10] Walpole, Ronald E. and Raymond H. Myers. *Probability and Statistics for Engineers and Scientists*. New York: Macmillan Publishing Company, 1985.