VIRTUAL STOCHASTIC SENSORS: HOW TO GAIN INSIGHT INTO PARTIALLY OBSERVABLE DISCRETE STOCHASTIC SYSTEMS

Claudia Krull, Robert Buchholz, Graham Horton
Department of Simulation and Graphics
Otto-von-Guericke-University Magdeburg
Universitätsplatz 2, 39106, Magdeburg, Germany
email: claudia.krull{robert.buchholz,graham.horton}@ovgu.de

ABSTRACT

This paper introduces the idea of a Virtual Stochastic Sensor. This paradigm enables the analysis of unobservable processes in discrete stochastic systems. Just like a virtual sensor, we use physical sensor readings to deduce the value of the quantity of interest. However, both the physical sensor readings and their relationship with the quantity of interest are stochastic. Therefore the measurement of our virtual stochastic sensor is a statistical estimate of the true value.

We describe a method to compute the result of the virtual stochastic sensor and show its validity and real-time capability for two example models. We also give system properties that must apply in order for the feasibility of virtual stochastic sensors, such as the sensitivity of the physical sensor output to changes in the quantity of interest.

The future potential of virtual stochastic sensors is their variability. They can be used to gain insight into hidden processes of partially observable systems, using readily available data. They enable online monitoring of production lines using already recorded data to ensure optimal control and maximum production efficiency.

KEY WORDS

virtual stochastic sensor, discrete stochastic model, state space-based simulation, time series analysis

1 Introduction

In industry today, highly complex production environments are prevalent. Precise measurement and control of the equipment and processes are needed in order to ensure high levels of efficiency. For optimal control of the production it is important to constantly monitor the current state of the equipment and the process. When deviations from prescribed performance parameters occur, they need to be detected quickly, in order for the operator to react and implement countermeasures.

However, sometimes important process parameters cannot be measured continuously, or only with a minimum delay, or they may not be directly measurable at all. In these cases, virtual sensors are often used instead of real sensory equipment. A virtual sensor is a model that takes the readings from physically existing sensors in the system and computes the value of the quantity of interest during runtime, so that this computed value can be used as if it were the result of a real sensor.

The challenge in using a virtual sensor is that a model must be available that can convert the physical sensor measurements into the value of interest. This model can have any form, but it must result in a definite sensor reading in a finite amount of time. Virtual sensors are often used in process and chemical engineering environments, where the cause and effect relationships are continuous and deterministic and are largely governed by closed mathematical equations.

On the other hand, many applications exist which are discrete, stochastic and non-continuous. Some examples are communication networks, assembly lines or logistics. In such cases, measurements (such as inter-arrival times of customers) are samples of random variables, and the quantities of interest are statistical in nature (such as waiting times or queue lengths). Here, standard virtual sensors cannot be applied.

We propose an approach that can be considered to implement a *virtual stochastic sensor*. Random inputs are used to gain insight into statistical properties of discrete stochastic systems which are not accessible directly. The paper introduces the concept of a virtual stochastic sensor and sketches the algorithm used for computation of the sensor results. We will also show the validity of our computed sensor reading characteristics and we will analyze the real-time capability of our virtual stochastic sensor using a small example.

Potential application areas include the monitoring and analysis of small production lines through temporarily installed external sensors and external expertise, instead of the expensive installation and maintenance of integrated sensors. One could also determine the characteristic of the arrival process by using information on the queue length and service process in a communication network or a real-world queuing system. The goal of both tasks would be to gain more understanding of the system based on readily available data and increase their efficiency without unnecessary intervention. An application outside of engineering would be the diagnosis of patients based on symptom information to give prognoses on the patients health status or on the efficiency of intervention methods.

We hope that virtual stochastic sensors can yield similar benefits to standard virtual sensors, which are well established in many application fields. For now, however, they are still a topic for further research.

2 Background and Previous Work

2.1 Virtual Sensors

Virtual sensors are very common in monitoring and control of process and chemical engineering. They are used to replace physical sensors that are too expensive, error prone, slow, high-maintenance, risky, or simply not possible to install. [10, 15, 18] Virtual sensors can be used differently, depending on their application:

- Virtual sensors can provide online monitoring of quantities, which could otherwise only be measured intermittently, or where a high latency between event and its detection would occur.
- Virtual sensors can be used in place of very expensive real sensors.
- Virtual sensors can provide measurements for quantities that are in fact not directly technically measurable, such as the friction of a tire on the street.

One example of an application of virtual sensors is to monitor the key variables in an internal combustion engine. This is an important problem, since knowledge of these variables can enable control algorithms which improve fuel efficiency. At the same time, however, adding sensors to the engine is both expensive and also interferes with the combustion process. By monitoring the voltage and electrical current in the spark plug, the combustion pressure, air to fuel ratio and various fault conditions such as misfires can be derived [6, 16]. In this manner, the spark plug is used as a virtual sensor.

The difficult part in using virtual sensors is to provide a model that can accurately compute the value of interest from the measured quantities [18]. In some cases the models are trained using a temporarily installed sensor that will be replaced with the virtual one later on. If no such model exists, or if it is unknown, neural networks can be employed. However, these still result in a set of rules that deduces the virtual sensor measurement directly from the input data.

Virtual sensors are applied to physical quantities such as voltage, energy, temperature or flow rate. These quantities are continuous and are governed by deterministic laws such as algebraic or differential equations. These can be used to construct a continuous, deterministic model which can be solved in real time.

By contrast, there are many technical scenarios which are discrete by nature. Typical examples are assembly lines, traffic systems and logistics. In these cases, discrete entities are of interest, whose dynamics are described by events. In addition, these systems are stochastic, dealing with random quantities such as service times, probabilities of different states and machine failures. Such systems are modeled using discrete stochastic paradigms such as queuing systems or (stochastic) Petri nets.

In cases which involve stochastic processes or decisions, deducing scalar values is not easily possible or even meaningful. For this reason, standard virtual sensors cannot be employed, because they cannot appropriately represent random variables. Nevertheless, it seems clear such applications could equally benefit from virtual sensors.

We therefore propose the new *Virtual Stochastic Sensor* as an approach to filling this gap. A virtual stochastic sensor fulfills the same role as a standard virtual sensor, but with respect to discrete random variables as opposed to continuous deterministic variables.

2.2 Stochastic and Hidden Models

Discrete stochastic models (DSM) such as queuing systems [3], stochastic Petri nets [1] or Stochastic Activity Networks [17] can accurately model discrete stochastic systems such as a production environment with imperfect machines and random input from different sources. Discrete-event simulation methods can be used to predict the behavior of these systems by emulating their random behavior.

In the case that such a system only contains random variables with exponential distributions, its model may be reduced to a Markov chain without loss of accuracy. Markov chains have the advantage of possessing a simple mathematical representation which admits straightforward, deterministic solution methods [3]. However, in practice, stochastic systems will almost always contain generally distributed random variables, which cannot be represented in a straightforward way by a Markov chain.

Discrete-event simulation methods are restricted to predicting the future: they do not permit observations obtained from a real system to be integrated into the calculation. One method of achieving this is to use a Hidden Markov Model. Hidden Markov Models assume a Markovian model of a system whose behavior cannot, however, be observed. They can take a trace obtained from real system behavior and use it in conjunction with a Markov model of the system in order to deduce which sequences of state changes most likely took place to generate it. Hidden Markov Models are used, for example, in speech recognition, where a recorded sequence of spoken sounds is used to compute what words and phrases have most probably been said [7].

It can be shown that by using multiple states, any distribution function can be modeled by a Markov chain [8]. This is achieved using the principle of so-called supplementary variables, which represent the time the system has remained in a given state. We are thus able to model any stochastic system with a Markov chain at the cost of needing to store and compute a larger number of variables. As the size of the model increases, this cost can quickly become prohibitive, since the number of additional states needed increases suffers from a combinatorial explosion.

By combining the method of supplementary variables with Hidden Markov Models, we obtain a paradigm which allows us to model a general stochastic system whose behavior is partially hidden and to use a trace obtained from a real-life measurement in order to deduce the system's hidden behavior. We refer to such models as Hidden non-Markovian Models (HnMM) [12]. It is this combination of two essentially unrelated techniques which forms the key to virtual stochastic sensors.

HnMMs are solved by exploring all possible paths that the system may take using discretized time steps. The basic approach is known as the Proxel method and is an example of a state space-based analysis method [9, 14]. It has been shown [13] that Proxels can be used to relate given symbol traces to the behavior of discrete stochastic systems that might have produced it. Recent improvements and adaptations to special classes of HnMM made the performance of Proxels practicable for small realistic systems [5].

3 The Virtual Stochastic Sensor

Standard virtual sensors use a mathematical or informatical model to represent the relationship between the measured quantities and the variables of interest. The input-output relationship of these models is continuous and deterministic, and results in scalar values in real time. By contrast, a Virtual Stochastic Sensor measures random quantities and uses a stochastic model to derive statistical output measures.

The Virtual Stochastic Sensor will allow us to deduce statistical properties of a system on the basis of samples of random variables taken at other locations in the system. In this sense, they are analogous to standard virtual sensors. However, since they are based on random samples taken over a period of time, most virtual sensor outputs will only be reliable after a sufficient number of measurements have been made. In this, they contrast with the standard case.

Virtual stochastic sensors can be applied when the following properties hold for the system of interest:

- It is possible to build a discrete stochastic model of the system or system part to be analyzed, e.g. we have enough information on the system structure, processes and physically installed sensors to build the model.

- The physical sensor readings (protocol entries) are discrete, time stamped, and stochastic.

In addition, there needs to be sufficient information in the physical sensor protocol to estimate the value of our virtual sensor reading. There needs to be a connection or correlation between the quantity of interest and the protocol entries, such that the following conditions apply:

- The value of the quantity of interest needs to have influence on the physical sensor readings from the system.

- The physical sensor readings are sensitive to changes in the quantity we are interested in, the quantity measured using the virtual stochastic sensor. Meaning that a change in the quantity of interest has sufficient impact on our physical sensor readings to be detected and characterized.

- There are not too many stochastic influences interfering with the influence of the quantity of interest.

Otherwise the protocol does not hold any or not sufficient information to deduce statistical estimates for the quantity of interest.

When these properties apply, a Virtual Stochastic Sensor can yield useful results. The following section will outline the method for computing the sensor results based on the system model and given protocols.

3.1 Computing the Sensor Output

We propose using a Hidden non-Markovian Model (HnMM) to describe the discrete stochastic system and the relationship between the recorded physical sensor results and the parameter of interest. The embedded discrete stochastic model (DSM) actually describes the complete system part of interest including all relevant processes and the quantity of interest. The HnMM extends the DSM by incorporating the physical sensor readings being produced. We can then use this HnMM of the system to evaluate measurements from the real system and generate statistical estimates for the measure of interest.

Solving the Hidden non-Markovian model enables us to determine possible system behavior that could have produced the given output. The procedure is given in detail in [12, 13], therfore we will only roughly outline the procedure here.

We are using the Proxel-based simulation method [9, 14] as a basis for our solution algorithm. This state space-based simulation method generates all possible development paths the system can take in discrete time steps. All possible system developments can be tested against the given protocol of physical sensor readings. Only the paths that could have produced the given time series are kept, all the other ones can be eliminated.

The Proxel method generates state changes on-the-fly, in order to reduce both computation time and memory consumption. Therefore we can also evaluate the validity of a given state change with regard to the given protocol. Whenever an input for our virtual sensor is available, of all the possible state transitions at that time step, only those which produce that particular input are relevant. For this reason, all other paths in the computation may be discarded. Overall, only those paths that could have produced the observed time series are considered.

The result of the computation is a set of all paths which are able to generate the given time series together with the probability that the system followed each of these paths. We can then analyze the value of the measure of interest in each path and deduce statistical estimates, such as the expected value, the development of the measure over time, or its probability distribution at any point in time.

Using Proxel-based simulation to compute the value of the Virtual Stochastic Sensor imposes another restriction on the system to be analyzed. Regardless of recent improvements Proxels are still a state space-based method, which suffers from state space explosion to some extend, also due to the use of supplementary variables to incorporate non-Markovian distributions. Therefore the Virtual Stochastic Sensors are only feasible to analyze relatively smalls systems or system parts. Our current estimates is that the system should not contain more than four parallel processes. Furthermore, the value of quantity of interest should be used as indicator of the path, and not the whole history of state changes needed.

In [12, 13] we have shown that the approach is valid for the small tester model also used here. The experiments described there assumed that the analysis was done offline using a time series that had been previously recorded. There it was shown that the approach was able to reconstruct system behavior that had not been recorded (the source of a larger than usual number of defective items).

However, the approach could also be used to determine the development of the measure of interest while the system is operating. This could be done by considering single sensor readings as they are recored, or a certain time window of the last n sensor readings to determine the current value of the measure of interest. This corresponds to the real-time mode that most virtual sensors are used in. We will test the real-time capability in the experiments section using the quality tester model described in detail in Section 4.2.

3.2 Potential Application Areas

Potential application areas of virtual stochastic sensors are manifold. One idea is to monitor an existing production line by temporarily installed sensors (cameras or photoelectric barriers) and using external expertize to for the installation and analysis of the sensor readings. The operator of such a production line could thus save the effort of integrating expensive sensory equipment in the production process. The goal of the analysis could be to determine the current bottleneck in the system, identify optimization potential or detect deviations of machine behavior from the manufacturers specifications. The operator could thus gain more insight into his production line, better control the process and increase production efficiency. [5]

Another possible application is in queuing systems in communication networks or real world queuing problems. One could determine the characteristic or behavior of the arrival process by using information on the queue length and service process. The task could be to locate the source of a clogged up communications channel or lost packets, or a too long queue in a service environment. This would again lead to more understanding of the system and one could devise an optimal intervention strategy without unnecessary interference with the system.

An application outside of engineering would be the diagnosis of patients based on symptom information and models of different diseases. The symptom records of a patient should be readily available already today, thus not imposing further work on medical personnel. These records could be used to determine the possible illnesses the patient is suffering from. From that a support system could devise a prognosis of the patients health status or even of success of intervention methods. [11]

4 Application Example

This section describes two simple scenarios, where virtual stochastic sensors could be used to gain insight into partially observable discrete stochastic systems. These are used in the experiment section to test the performance and the validity of virtual stochastic sensors.

4.1 Fast Food Restaurant

The first example model is academic and demonstrates how we can deduce interesting measures from seemingly unimportant data. The setting is a fast food restaurant with one employee and two types of customers, customers that buy a meal without a drink and customers buying a meal including a drink. Preparing a meal without a drink takes a randomly distributes amount of time ($\sim Normal(50, 10)\ s$), and the employee has to walk $2\ m$ during the process. Preparing a meal that includes a drink takes on average $10\ s$ longer and the employee has to walk $5\ m$ during the process. This can add up to a considerable distance covered over a whole shift.

One of our employees who is working alone at the counter on his shift complains that the distance he has to walk is too large, and demands to be supported by another colleague. To support his claim, he shows us his estimate of the distance traveled during the last week, every day exceeding $1.8\ km$. However, we cannot validate his claim, since we cannot monitor the employee due to privacy concerns. However, we can monitor the door of the restaurant with a sensor already installed there.

The Hidden non-Markovian model of the system including the customers, the employee and the sensor at the door is shown in Figure 1. The discrete stochastic system representing the arrival and service processes is again represented using a stochastic Petri net (SPN) [2]. The arrival process is represented by a timed transition with an exponentially distributed firing time mean $\mu = 60\ s$. The customers then have to wait until the single service person is available and can then advance to the decision, whether

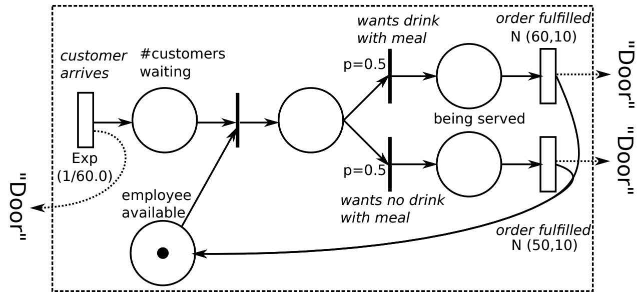

Figure 1. Hidden non-Markovian Model of Fast Food Restaurant

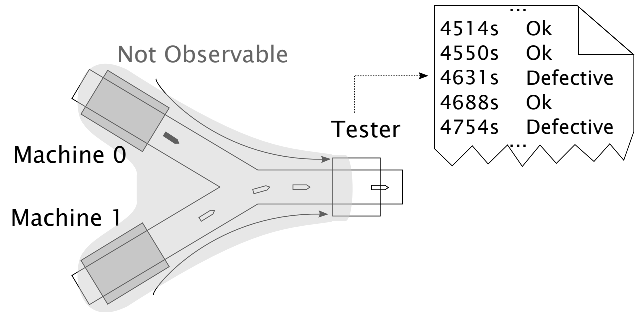

Figure 2. Schema of Quality Tester Model with Incoming Item Streams and Example Protocol

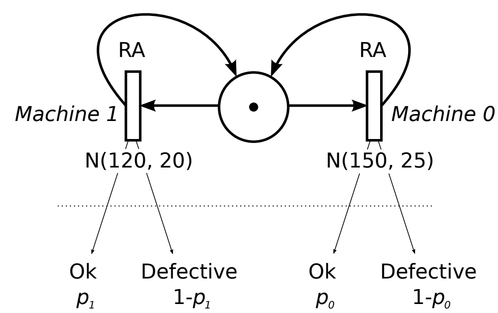

Figure 3. Hidden non-Markovian Model of Quality Tester

they want a drink with their meal, or not. About half of the customers want a drink with their meal, which is represented by the $p = 0.5$ at the immediate transitions of the process fork. The service of a customer takes a normally distributed amount of time, while a meal including a drink needs on average $10\ s$ longer. After completing the service of a customer, the service person is again available.

The dashed line represents the boundary of the unobservable system. The arrival of a customer and the fulfillment of an order both generate a $Door$ signal, which can be detected from the outside. From the protocol of the door opening and closing, we will try to deduce the individual customer's service time, from these their menu choice and finally the distance the employee covered on that day. Thus we will try to validate or disprove his complaint in the experiments section.

The difference to the tester model described in the next section, which we have also used in previous publications [5], is that the fast food model involves three independent processes, a shared resource and a fork or decision in the process. This makes it more complex and thus more realistic than the models we have tested before.

4.2 Item Tester in a Production Line

The second application example has been taken from [5] and models a tester in a production line. Two machines with different characteristics produce streams of items that are merged prior to testing the items. Our physical sensor reading is the testing protocol with the test results and timestamps of the tests. The source of each single item is not recorded and cannot be determined trivially, since both input streams are stochastic with overlapping process time distributions. The schematic of the tester production line is shown in Figure 2. The grey overlay symbolizes that this part of the prodcution line is not observable and therefore one does not know the source of a particular item. On the right side of the schematic one can see a small excerpt from such a tester output protocol with the timestamps and the test results.

Figure 3 shows the HnMM representation of the quality tester model. The discrete stochastic system itself is represented by a Stochastic Petri Net (SPN) [2] with only one place, representing the working state of the tester. The two independent input streams are represented by two timed transitions, which have a race age policy (RA), meaning that the firing of one transition does not influence the firing time of the other. Machine 0 is the slower of the two with a normally distributed production interval with mean $\mu = 150\ s$ and a standard deviation of $\sigma = 25\ s$. The observable output symbols being produced are the test results of each item, which extend the SPN to form a HnMM. The probabilities for items from machine 0 are $P_0(OK) = p_0$ and $P_0(Defective) = 1 - p_0$, analogously for machine 1 $P_1(OK) = p_1$ and $P_1(Defective) = 1 - p_1$. The SPN is the hidden part of the model, represented by the dashed line, while only the tester results are observable from the outside.

In our example we assume that both machines work equally well and therefore have comparably low defect probabilities. However, recent tester protocols suggest, that at least one of the two machines is not working according to specifications, since the number of defective items in the whole production is unusually large. The task is to determine if one of the machines is the cause for this problem, and which one does produce a larger than usual number of defective items.

In [4] and [5] we have already shown that we can reliably deduce the machine that produces more defective items when analyzing a complete production protocol. Here we will view the problem another way, we will continuously analyze the result protocol and give online estimates for the source of a defective item and the current defect rate of each of the machines. By monitoring these while the machines are running, the production manager can intervene, should anything unusual happen to the error rates of the respective machines. In the experiments section we will test the whether the machine having produced a particular defective item can be determined reliably, and whether the Virtual Stochastic Sensor is fast enough for online monitoring.

5 Validation and Performance Experiments

This section contains the results of different experiments. First we will test the validity of virtual stochastic sensors on the fast food model. The validity of the tester output model has been shown already in [5]. The second set of experiments is aimed at determining the real-time capability of the sensor using the tester model. Here we will analyze a time series of the last items and estimate the source of a defective item and the defect rate of each machine.

5.1 The Fast Food Restaurant

The first experiment is performed using the HnMM for the fast food restaurant shown in Figure 1. The experiments were performed on an Intel Core2 Duo processor with 3 GHz and 4 GB RAM. We have created the test output sequences (protocols of door sensor readings) using a discrete event-based simulation of the HnMM in the software tool AnyLogic 6 from XJTechnologies. The actual distance covered by the employee in the run producing the a particular output sequence was also recorded. The runtime of the analysis of a protocol for an 8-hour working shift was around ten seconds.

Figure 4 shows the result of the analysis of an example trace of length 500 symbols, corresponding to $10,000$ seconds of system time. The actual distance covered in the example run was 550 m. The analysis result was an estimated distance of 545.45 m. Figure 4 shows the histogram of the possible distances covered and their respective probability. The estimated distance is very close to the actual one and the histogram shows a bell shape around the maximum.

Figure 5 shows the result of ten such test runs which were performed with protocols corresponding to 8-hour shifts at the restaurant. We performed ten statistically independent replications of the discrete event-based simulation to generate the output traces. The graph shows the actual distance covered in each replication (grey bars), and the 95% confidence intervals of the analysis result. In nine out of ten cases, the actual distance lies in this confidence interval. We can deduce the distance covered by the employee quite accurately, which supports the validity of the Virtual Stochastic Sensor.

If the traces analyzed represented ten typical working

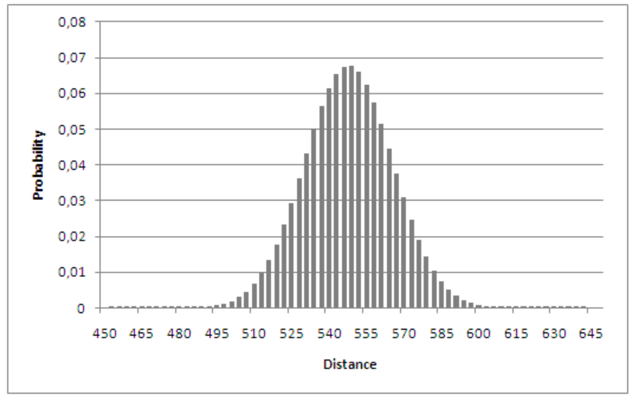

Figure 4. Histogram of Probability for Possible Walking Distances Extracted from one Example Trace of Fast Food Restaurant Model

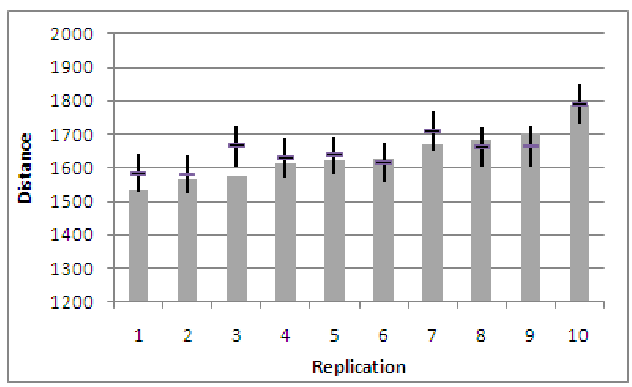

Figure 5. Real Distance Covered by the Employee for Ten Independent Replications and 95% Confidence Intervals of the Analysis Results

Table 1. Table Showing the Comparison of True Defect Source and Estimated Defect Source for a small Example Trace of the Tester Model

timestamp	true source	estimated source	probability M0	probability M1
602.88	M0	M0	0.882	0.118
2788.18	M1	M1	0.453	0.547
4285.68	M1	M1	0.215	0.785
5149.68	M0	M0	0.917	0.083
5442.24	M0	M0	0.962	0.038
5583.75	*M1*	*M0*	0.688	0.312
6906.17	M0	M0	0.547	0.453
8167.13	M1	M1	0.373	0.627
8606.95	M0	M0	0.693	0.307

days in the restaurant, we could conclude that only on one of the ten days does the confidence interval of the distance covered include the $1.8\ km$ claimed by the employee. We could thus decline his request for support and in effect save a lot of money.

5.2 Online Analysis of Defect Probability

The second experiment is aimed at testing the real-time capability of the Virtual Stochastic Sensor. We are using the tester model with the HnMM shown in Figure 3. Both input streams are normally distributed in their inter-arrival times, but with different mean and standard deviation. The example traces were again generated using a discrete event-base simulation of the tester using the tool AnyLogic 6. In this simulation we increased the defect probability of machine 0 to 0.1 compared to machine 1 where the defect rate is 0.05, thus making machine 0 the cause of a higher than usual defect rate. We will analyze the protocol up until the current item and try to determine the current defect rate of the machines and the source of the last defective items.

The test trace containing 1500 items corresponds to approximately 10.000 seconds of system time. The computation of the source of the current item and the current defect rate of each machine takes at most 10 seconds of computation time. With an average gap of 60 seconds between two items, that can be regarded as suitable for real-time.

Table 1 shows the analysis results for the first ten defective items. For each defective item the true source and the estimated source is shown, as well as the probability with which it was produced by either machine. Only in one out of these ten cases the real source is not detected correctly. The overall recognition rate of the experiment including 106 defective items or $10,000$ seconds of production time was 71%. Most recognition errors occurred, when both machines had approximately the same probability to produce a particular defective item. We can thus conclude that an online analysis of the source of a particular defect is possible and feasible.

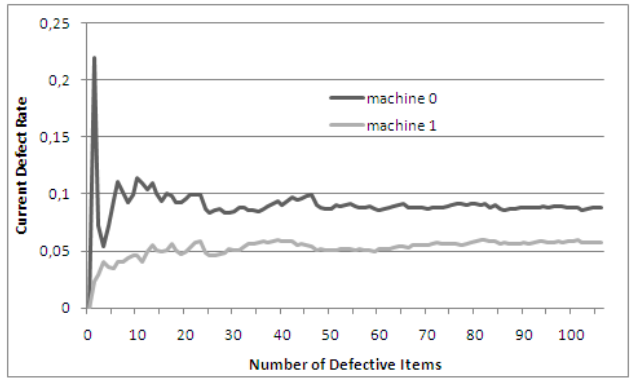

Figure 6. Development of Defect Rate of Each Machine, Values Shown at the Each Detection of a Defective Item

Figure 6 shows the development of the defect rate of each of the machines over time with continuing analysis of the protocol. One can see, that with each defective item tested, the defect rate becomes more smooth. The higher defect rate of machine 0 is detected reliably, even after only a short protocol length and thus small absolute number of defects. This shows that we do not need to analyze the complete protocol to gain reliable results for the cause of an unnaturally high defect rate, but we can detect that using only a small amount of information. The more information we have, the longer the protocol is, the more reliable the result of the analysis.

5.3 Discussion of Experiment Result

The experiments showed the validity of the Virtual Stochastic Sensor paradigm for a more model with three independent processes, a fork and a shared resource. In the same example we could deduce interesting data (distance covered by an employee during an 8-hour shift) from seemingly unrelated measurements (door of the restaurant opening and closing). The runtime of one analysis was still within a few seconds, which makes us confident, that we will be able to apply virtual stochastic sensors in realistic settings with more complex models.

We also showed the real-time capability of our paradigm in a production line tester example. With a runtime around $10\ s$ The analysis was fast enough to be performed in the average gap between two sensor readings, thus making it possible to compute real time results for the Virtual Stochastic Sensor.

The virtual stochastic sensor paradigm is applicable in these cases presented, because we have discrete stochastic models of the systems of interest. The the quantities of interest have an influence on the output of the physical sensors in the system. The meal choice of a customer influences his service time, and therefore the time stamp of the door movement in the protocol. The arrival time of an item in the quality tester corresponds with the time stamp in the tester protocol and the machines defect probability influences the quality of the item and thereby the type of protocol entry. Furthermore, there are not too many other influences interfering with the quantity of interest in their influence on the protocol. Finally, the systems are relatively small, so that computation time of realistic trace lengths does not exceed one minute.

6 Conclusion and Outlook

In this paper we introduced the Virtual Stochastic Sensor. The idea is similar to virtual sensors, which are applied to continuous systems governed by deterministic laws. However, virtual stochastic sensors can analyze discrete stochastic systems on the basis of stochastic, time-stamped and discrete readings of physical sensors. Thus they can cope with systems containing uncertainties such as production environments or communication systems.

We can model the system of interest using Hidden non-Markovian models. By solving these using a state space-based simulation method, we can compute the value of the Virtual Stochastic Sensor. Since we are dealing with stochastic systems, the output of a virtual stochastic sensor is not a scalar value, but always a statistical estimate: an expected value, a distribution or a range.

The experiments showed that for a small model the method is real-time capable. We also showed that virtual stochastic sensors can deduce values of interest from seemingly meaningless data. The experiment with the fast food restaurant model also showed that the paradigm is feasible for more complex systems. We will yet have to gradually increase the complexity of our models and adapt the analysis method to be able to analyze systems of realistic size in the future.

Potential application areas include the analysis and optimization of small production lines by using temporarily installed external sensor equipment. Furthermore one could analyze the optimize queuing systems based on intermittent readings of the queue length and some further process information. Diagnosing a patient and devising a prognosis based on the record of his symptoms is also conceivable.

References

[1] F. Bause and P. S. Kritzinger. *Stochastic Petri Nets*. Vieweg, 2002.

[2] A. Bobbio, A. Puliafito, M. Telek, and K. S. Trivedi. Recent developments in non-markovian stochastic petri nets. *Journal of Systems Circuits and Computers*, 8(1):119–158, 1998.

[3] G. Bolch, S. Greiner, H. de Meer, and K. S. Trivedi. *Queuing Networks and Markov Chains*. John Wiley & Sons, New York, 2006.

[4] R. Buchholz, C. Krull, and G. Horton. Efficient event-driven proxel simulation of a subclass of hidden non-markovian models. In *7th EUROSIM Congress on Modelling and Simulation*, August 2010.

[5] R. Buchholz, C. Krull, T. Strigl, and G. Horton. Using hidden non-markovian models to reconstruct system behavior in partially-observable systems. In *3rd International ICST Conference on Simulation Tools and Techniques*, March 2010.

[6] L. Eriksson and L. Nielsen. Ionization current interpretation for ignition control in internal combustion engines. *Control Engineering Practice*, 5(8):1107–1113, 1997.

[7] G. A. Fink. *Markov Models for Pattern Recognition*. Springer, Berlin, Heidelberg, 2008.

[8] R. German and C. Lindemann. Analysis of stochastic petri nets by the method of supplementary variables. *Performance Evaluation*, 20:317–335, 1994.

[9] G. Horton. A new paradigm for the numerical simulation of stochastic petri nets with general firing times. In *Proceedings of the European Simulation Symposium 2002*, pages 129–136. SCS European Publishing House, 2002.

[10] P. H. Ibargüengoytia and A. Reyes. Constructing virtual sensors using probabilistic reasoning. In *MICAI*, pages 218–226, 2006.

[11] C. Krull, R. Buchholz, and G. Horton. Matching hidden non-markovian models: Diagnosing illnesses based on recorded symptoms, 2010. to be published in Proceedings of the 24th annual European Simulation and Modelling Conference ESM'2010, October, Hasselt, Belgium.

[12] C. Krull and G. Horton. Hidden non-markovian models: Formalization and solution approaches. In *Proceedings of 6th Vienna Conference on Mathematical Modelling, Vienna, Austria*, pages 682–693, February 2009.

[13] C. Krull and G. Horton. Solving hidden non-markovian models: How to compute conditional state change probabilities. In *21st European Modeling and Simulation Symposium*, Santa Cruz de Teneriffe, Spain, September 2009.

[14] S. Lazarova-Molnar. *The Proxel-Based Method: Formalisation, Analysis and Applications*. PhD thesis, Otto-von-Guericke-University Magdeburg, November 2005.

[15] N. Oza, A. Srivastava, and J. Stroeve. Improvements in virtual sensors: using spatial information to estimate remote sensing spectra. *Geoscience and Remote Sensing Symposium, 2005. IGARSS '05. Proceedings. 2005 IEEE International*, 8:5606 – 5609, jul. 2005.

[16] R.J.Howlett, M. M. de Zoysaa, S. D. Walters, and P.A.Howson. Neural network techniques for monitoring and control of internal combustion engines. In *Int. Symposium on Intelligent Industrial Automation, Genova, Italy*, 1999.

[17] W. H. Sanders and J. F. Meyer. Stochastic activity networks: Formal definitions and concepts. In *Lecture Notes in Computer Science no. 2090*, pages 315–343. Berlin: Springer, 2001.

[18] E. Wilson. Virtual sensor technology for process optimization, 1997. Presentation at the ISSCAC 1997.