SANGE – Stochastic Automata Networks Generator
A tool to efficiently predict events through structured Markovian models

Joaquim Assunção *†, Paulo Fernandes *, Lucelene Lopes *, Angelika Studeny †, Jean-Marc Vincent †

*PUCRS University – Computer Science Department – Porto Alegre, Brazil
† Univ. Grenoble Alpes, LIG, F-38000 Grenoble, France
{joaquim.assuncao, angelika.studeny, jean-marc.vincent}@inria.fr
{joaquim.assuncao, paulo.fernandes, lucelene.lopes}@pucrs.br

Abstract

The use of stochastic formalisms, such as Stochastic Automata Networks (SAN), can be very useful for statistical prediction and behavior analysis. Once well fitted, such formalisms can generate probabilities about a target reality. These probabilities can be seen as a statistical approach of knowledge discovery. However, the building process of models for real world problems is time consuming even for experienced modelers. Furthermore, it is often necessary to be a domain specialist to create a model. This work illustrates a new method to automatically learn simple SAN models directly from a data source. This method is encapsulated in a tool called SAN GEnerator (SANGE). This new model fitting method is powerful and relatively easy to use; therefore this can grant access to a much broader community to such powerful modeling formalisms.

1 Introduction

Stochastic Automata Networks (SAN) is a powerful formalism to describe systems as stochastic models. Through these models we can derive probabilities concerning some event or set of events of a system. Our research group has a record of successful development of stochastic models for behavior prediction from several domains, *e.g.*, geological events [2], production lines [7] and distributed software development teams [8]. In all these examples, the model construction required domain specialists and a large amount of stochastic modeling knowledge. The resulting models are very accurate in predicting the behavior of the realities as could be verified by comparison with records of each reality behavior.

Typically, SAN model construction is a top-down driven approach, *i.e.*, first the target reality is analyzed, then its behavior is translated into a stochastic model. Once we have a complete SAN model, it is possible to use a collection of specialized algorithms that can solve it [5]. The problem of this approach is that it is specific to a given system. In other words, each new system must be carefully analyzed before the creation of the model. This analysis usually is performed via handmade steps such as data analysis and data selection.

DOI reference number: 10.18293/SEKE2015-087

In a previous work [1], we proposed a bottom-up process to forecast events using time series and stochastic models (Figure 1). This modeling approach also speeds up the time to develop a representative model from input data. However, we did not have a technique to automatically generate SAN code, but only plain Markov chains (MC). The extension of this previous work to generate SAN models increases our potential to handle more complex (multidimensional) data.

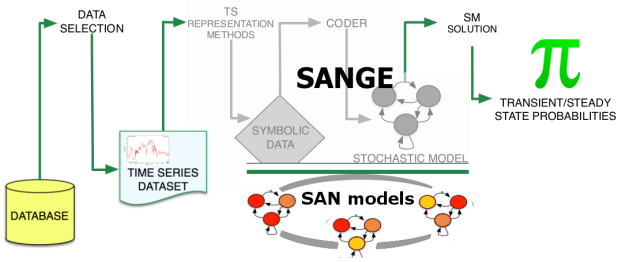

Figure 1: A Dimensionality Reduction Process to Forecast Events Through Stochastic Models [1] enhanced by the generation of SAN models (SANGE).

A bottom-up, MC-based, approach gives a solution for any generic model. However a plain (unstructured) MC is usually a limited, memory expensive model. Limited in the sense that you have a bulk representation for a system, regardless of how complex it may be. Memory expensive because a system with S states is represented by a transition probability matrix of S^2. In the best case, the number of non-zero entries will be in the order of $2S$.

SAN formalism is modular and its model representation is composed by a collection of sub-systems, which are usually much more compact benefiting from tensor representations. Thus, in this paper we show a new approach to fit SAN models directly to data, reducing the model size. This approach has been implemented in a tool called SANGE (SAN GEnerator) that automatically generates SAN models from a

dataset.

The excessive human effort to construct models can be avoided with a tool that handles the formal tasks to convert data into state transitions. Consequently, the user can focus on more interesting tasks, such as interpreting the results and applying the gain knowledge. Additionally, SANGE performs dimensionality reduction using time series representation methods [10]. Thus, SANGE is also capable of automatically fitting the model to input data, possibly achieving better models than those made by humans.

Our algorithm was inspired by a well known and broadly used formalism, Hidden Markovian Models (HMM), and its fitting algorithm (Baum-Welch, BW) [4]. Thus, the HMM user only needs to understand the modeling basics and how to interpret the generated HMM model.

The BW algorithm is a special case of the Expectation-Maximization algorithm, using forward-backward probabilities to estimate the model parameters. Our approach can be seen as an adaptation of this algorithm implementing only the forward procedures. However, our solution works with a structured formalism, which naturally can provide higher accuracy and can be more flexible and user-friendly to describe a system.

2 Computational Kernel

SANGE's main objective is to reduce the time spent on generating SAN models, thus, opening the use of SAN models to non-specialists. However, our solution needs records of a system behavior in the form of time series that will be assigned to the variables of interest for the system.

SANGE's basic operation consists in the composition of a set of time series describing how system variables behave [9]. Considering a system with n interest variables ($v^{(i)}$ with $i = 1,\ldots,n,$) we need a behavior sample of the system in the form of n time series with the successive values of the variables through time. With these time series, the first step is to identify the points of interest in time as the time ticks where at least one of the variables changes its value. Once the time ticks of interest are identified, the second step is to determine the possible values for the variables in order to define the stochastic model. This process is summarized in an example with three time series in Figure 2 and Figure 3 showing the identification of 11 time ticks (a) and the three succession of values for variables $v^{(1)}$, $v^{(2)}$ and $v^{(3)}$.

The examples of Figure 2 and Figure 3 results in three automata where local states are given by the observed values for each variable. Transitions events refer to the possible changes in states. These can happen locally within one automaton or simultaneously for several automata (synchronizing events). Figure 4 presents the SAN model for this example.

To compute the rates of local events we must count how many transitions take place starting in each local state. For

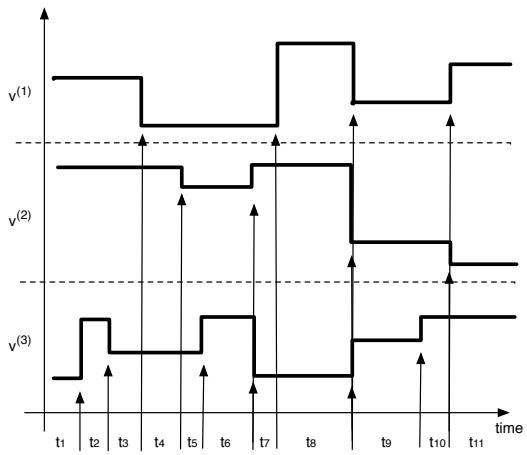

Figure 2: Example of SANGE basic process to three time series - identifying time ticks of interest.

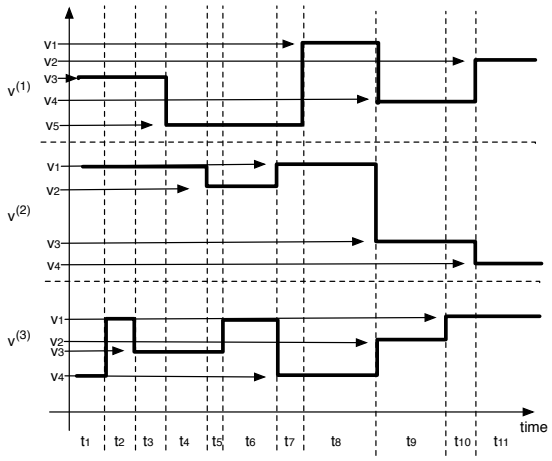

Figure 3: Example of SANGE basic process to three time series - identifying transitions between variable states.

example, consider the transition from state $v_3^{(1)}$ to state $v_5^{(1)}$. Observing all time ticks, we see state $v_3^{(1)}$ at the end of ticks t_1, t_2 and t_3. In t_1 and t_2 automaton $V^{(1)}$ does not change state, before going to $v_5^{(1)}$ in t_3. Therefore, one in three times the event $v_3^{(1)} \rightarrow v_5^{(1)}$ occurs, and the rate of this event is $1/3$.

For synchronizing events the computation is similar, but since more than one automaton is concerned, we now have to count the number of times that a certain number of combination of states occurs. For example, we look at automata $V^{(1)}$ and $V^{(2)}$ and the combination of states $v_4^{(1)}$ and $v_3^{(2)}$. This combination occurs at the end of time ticks t_9 and t_{10}, and the synchronizing event happens at one of these two time points (after t_{10}), hence, its rate is $1/2$.

From a practical point of view the current version of

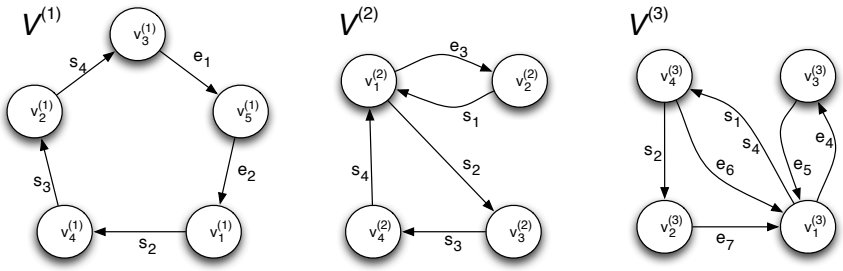

Figure 4: Equivalent SAN model for the three time series example of Figure 3.

SANGE can generate either a SAN model or a Markov chain output (which is basically a SAN model with a single automaton and only local events). The output is in the format of a `.san` file, that can be directly fed into a SAN solver, such as PEPS [5] that performs state of art Kronecker solutions [6].

The presented example consists of quite short time series, which limits the validity of the generated model. In real cases, much longer time series, where events are represented by a large number of state successions, must be considered in order to see statistically relevant patterns. Here we limited the amount of data for the sake of clarity, yet an extended version of this work is available in [3].

3 Example: Weather in Gothan city

To facilitate the understanding, we use a classical example for Markov models, *i.e.*, forecasting weather events [11]. The most basic example is a Markov chain with 3 states, representing **R**aining, **S**unny and **C**loudy (Figure 5). Probabilities are assigned to the transitions between states as well as staying in the same state.

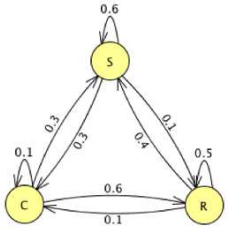

Figure 5: Classical example Markov chain model.

By solving this model we can achieve the transient and stationary probabilities of being in one of the three states. As the only representation of such a model is by a probability matrix, It can be computationally hard to handle for real world applications with many states. Furthermore, the best known formalisms, Markov chains and HMMs, can not integrate models with multiple automata in a unique system, *i.e.* they do not have a structure for this. By using SAN with SANGE it is possible to generate a structured version, which allows us to assemble more elements to our model and solve those as one.

SANGE encapsulates the techniques described in the Section 2; thus, it provides an interface for this basic and more advanced statistical tools. As pointed out before, the algorithm merges the SAN and TS characteristics.

The following example does consider nor real data neither the adequacy of the model; Our goal here, is to illustrate how SANGE works with multiple variables and how easy is to create a model with it. Although SANGE is a prototype, the basic functions are implemented and some basic models can be generated.

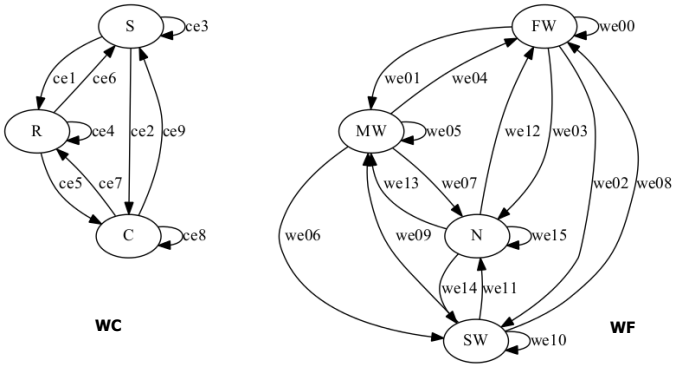

Figure 6: Generic SAN model for weather conditions and wind force. **“ce"** means climate event and **“we"** wind event.

For ease of understanding, Figure 6 shows a generic model with 2 automata, WC for weather conditions and WF for wind force. The states's labels WC are **R**aining, **S**unny and **C**loudy. In WF, the states correspond to the wind velocity, **F**ast, **M**edium, **S**low and **N**one.

Assuming that this model is accurate to predict the wind and the climate of a city called *Gothan*, we want to know the probability of Gothan facing rain and fast wind at the same time. We do not have many records, so we need to learn this probability by a small sample which is formatted as Table 1.

In this case, the probability to have rain and fast wind is 4.5%. In this example a basic combination of two automata

Table 1: Sample for input data

Raw data		Symbolic data	
weather	**wind speed**	**weather**	**wind speed**
cloudy	48	b	d
raining	16	c	b
sunny	26	a	c
...	...	...	...
sunny	9	a	a

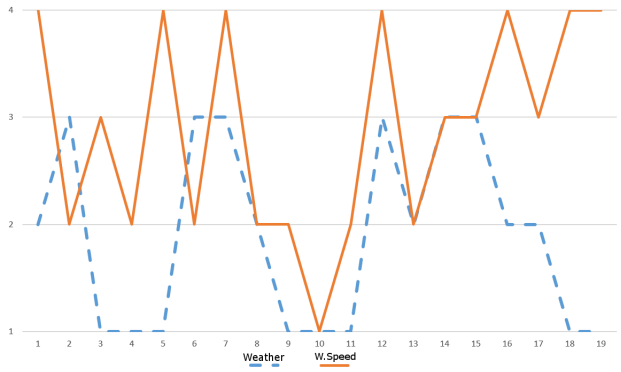

Figure 7: Line plot after the symbolic representation. Each letter is assigned to a value, $a = 1$, $b = 2$, $c = 3$, $d = 4$.

was used with a maximum number of 80 states. However, the number of automata and states could be much larger. This is a very compact representation and through SANGE; it scales better and demands less effort than the equivalent Markov model, allowing an easier derivation of probabilities in large models.

4 Final Remarks

As the core technique to data mining, statistics are important to knowledge discovery. It allows us to infer probabilities through samples instead of considering the complete behavior data. Stochastic formalisms are heavily based on probabilistic techniques. The use of stochastic modeling tools is promising to forecast the behavior of systems which can be described as sets of time series.

Our main achievement was to introduce SAN formalism to the process (illustrated in Figure 1) in an automated way, allowing non-specialist users to take advantage of SAN's structure and solutions. Through the automatic fitting, we create a bottom-up approach that can be broadly used, once that the learning process avoids the human effort to manually create such models. Compared to the traditional modeling approach, our implemented solution, SANGE, represents an interesting option that simplifies the effort to construct SAN models. As SANGE is a first attempt to automatically generate SAN models that can be useful to real world datasets, we have introduced a new method for knowledge discovery through stochastic modeling.

For future work, we will improve our algorithm by adapting the forward-backward procedures from BW algorithm, improving SANGE capacity to handle more complex SAN models for real datasets.

5 Acknowledgments

"This work was conducted during a scholarship supported by the International Cooperation Program CAPES/COFECUB at the University Joseph Fourier. Financed by CAPES - Brazilian Federal Agency for Support and Evaluation of Graduate Education within the Ministry of Education of Brazil."

References

[1] J. Assunção, P. Fernandes, L. Lopes, and S. Normey, *A dimensionality reduction process to forecast events through stochastic models*, in SEKE 2014, Jul 2014, pp. 534–539. ISBN-13: 978-1-891706-35-7.

[2] J. Assunção, L. Espindola, P. Fernandes, M. Pivel, and A. Sales, *A structured stochastic model for prediction of geological stratal stacking patterns*, Electronic Notes in Theoretical Computer Science, 296 (2013), pp. 27 – 42.

[3] J. AssunÇï£¡o, P. Fernandes, L. Lopes, A. Studeny, and J.-M. Vincent, *SANGE -Stochastic Automata Networks Generator. A tool to efficiently predict events through structured Markovian models (extended version)*, research report, Inria Rhône-Alpes, Mar. 2015. https://hal.inria.fr/hal-01149604.

[4] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, *A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains*, The Annals of Mathematical Statistics, 41 (1970), pp. 164–171.

[5] L. Brenner, P. Fernandes, B. Plateau, and I. Sbeity, *PEPS2007 - Stochastic Automata Networks Software Tool*, in Proceedings QEST, 2007, pp. 163–164.

[6] R. M. Czekster, P. Fernandes, and T. Webber, *Efficient vector-descriptor product exploiting time-memory trade-offs*, ACM SIGMETRICS Performance Evaluation Review, 39 (2011), pp. 2–9. doi: 10.1145/2160803.2160805.

[7] P. Fernandes, M. O'Kelly, C. Papadopoulos, and A. Sales, *Analysis of exponential reliable production lines using kronecker descriptors*, Int. Journal of Production Research, 51 (2013), pp. 2511–2528.

[8] P. Fernandes, A. Sales, A. R. Santos, and T. Webber, *Performance evaluation of software development teams: a practical case study*, Electronic Notes in Theoretical Computer Science, 275 (2011), pp. 73 – 92.

[9] J. Lin, E. Keogh, S. Lonardi, and B. Chiu, *A symbolic representation of time series, with implications for streaming algorithms*, in Proceedings of the 8th ACM SIGMOD, DMKD '03, New York, NY, USA, 2003, ACM, pp. 2–11.

[10] J. LIN, E. KEOGH, L. WEI, AND S. LONARDI, *Experiencing sax: a novel symbolic representation of time series*, Data Mining and Knowledge Discovery, 15 (2007), pp. 107–144.

[11] W. J. STEWART, *Probability, Markov Chains, Queues, and Simulation*, Princeton University Press, USA, 2009.