SPIKE-Dx : A Low-Power High-Throughput Fault Diagnostics Tool using Spiking Neural Networks for Constrained Systems

Chetan Kulkarni[1], Johann Schumann[1], and Anupa Bajwa[2]

[1] *KBR Inc, NASA Ames Research Center, Moffett Field 94035 CA, USA*
chetan.s.kulkarni@nasa.gov, johann.schumann@nasa.gov
[2] *NASA Ames Research Center, Moffett Field 94035 CA, USA*
anupa.bajwa@nasa.gov

ABSTRACT

Diagnostic systems are important for many aerospace systems, which are severely limited in available power, like CubeSats or UAVs. Therefore, traditional diagnostics systems cannot be used due to their substantial footprint and constraints. In this paper, we present our very low power diagnostic tool SPIKE-DX to monitor critical systems with constrained computational and energy resources. This is made possible through spiking neural networks (SNNs), which are executable within optimized simulation environments and further implemented on cutting-edge neuromorphic hardware.

Based upon Failure Mode and Effect Analysis (FMEA) framework, Diagnostic Bayesian Networks (DBNs) can be constructed that provide powerful means for diagnostic reasoning. In this paper, we describe such DBNs and a method to automatically translate the DBN into highly structured networks of spiking neurons for execution in SPIKE-DX.

1. INTRODUCTION

Diagnostic tools are critical for many aerospace systems, which are severely limited in available power. Typical examples include CubeSats or battery-operated UAVs. They have numerous subsystems and sensors and need up-to-date system health information that must be provided by a diagnostics system. However, traditional diagnostics systems have a substantial computational and power footprint.

For this paper, we have set the following goals: Develop a powerful on-board diagnostic reasoning system, which consumes minimal electrical power. Use and evaluate Spiking Neural Networks (SNNs), executed on modern neuromorphic hardware for efficiency and low power.

Build a diagnostic system based on FMEA models featuring probabilistic reasoning and confidence metrics. Design the diagnostic system for certification and V&V.

In this paper, we propose to translate the diagnostic BN into a Spiking Neural Network (SNN), which consists of groups of neurons which are sparsely interconnected. Excitatory-type neurons are used to propagate information, inhibitory neurons weaken signals and form the basis for a principled translation, which implements Bayes Belief Reasoning. The translation of the BN is performed in a pattern-oriented manner: each small subgroup of connected Bayesian nodes (Markov Blanket) is translated into individual groups of neurons and weighted connection patterns. These groups of neurons are then composed into a bigger SNN, which models the original BN (Paulin & van Schaik, 2014; Rao, 2004; Yu, Huang, & Liu, 2018). The SNN is then compiled onto the neuromorphic hardware using existing, compiler-based techniques.

We will evaluate our approach using a NASA-relevant Case Study on monitoring the power system of an autonomous UAS at LaRC (Corbetta & Kulkarni, 2019; Hogge et al., 2018). Based on existing system and diagnostic models, we propose to develop the diagnostic Bayesian Network, and use it to test and validate our BN-to-SNN translation framework.

Application in safety-critical environments requires that diagnostic models and reasoning algorithms can be verified, validated, and certified. In this research work, we combine the power and expressiveness of Bayesian Belief Networks (BNs) with the ultra-low power requirements of Spiking Neural Networks (SNNs) on modern neuromorphic hardware (Christensen et al., 2022). We will develop a translation of BNs to SNNs, which is efficient and amenable to certification. This is in stark contrast to Machine-learning based approaches with Deep Neural Networks (DNNs) (Zuo, Zhang, Zhang, Luo, & Liu, 2021), which lack traceability and are not certifiable with state-of-the art technology

Developing an efficient diagnosis procedure involves two main steps (see Figure 1): (i) build the BN structure and (ii)

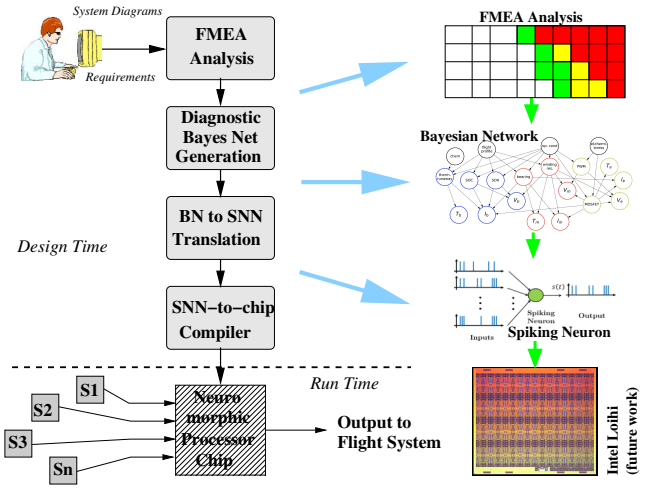

Figure 1. The Bayesian belief networks (BN) for diagnostics can be directly generated from the FMEA analysis and system diagrams and requirements in a model-based manner. Subsequently, the BN is translated into a spiking neural network (SNN) in the Nengo framework. Future work will use a translator to enable the execution on neuromorphic hardware, e.g., Intel Loihi (Davies et al., 2018)

establish the conditional probability tables (CPT) of each node. The relationships between each node are defined by pointing arcs from causes to failure modes, and from failure modes to effects, resulting in the construction of the BN structure (Moreno-Bote & Drugowitsch, 2015). Once the BN is completely defined, it can be used to detect and localize a fault in a complex system by turning observable nodes to True or False, based on the qualitative information provided by the FMEA. The diagnosis procedure updates the probabilities using Bayesian inference to determine the root cause with the highest failure probability when an evidence (observable) node is triggered (Davies et al., 2018). The dependency among elements in FMEAs do not have to be restricted to deterministic relationships in BNs.

Traditional Bayesian Belief Reasoning algorithms (Zermani, Dezan, Chenini, Diguet, & Euler, 2015) are executed on CPU, GPU, or FPGA (Fang, Shi, Dong, Fan, & Ren, 2017) and exhibit substantial computational footprints, since numerous floating-point operations are needed. This prohibits their use for on-board diagnostics in many aerospace applications, such as autonomous UAS, distributed sensor networks, or CubeSats.

In recent years, tremendous progress has been made in development and deployment of neuromorphic hardware (e.g., Google TPU, Intel Loihi, Apple M1/2, Tesla, Brainchip, and others). They promise very high performance for AI tasks while reducing their power consumption by many orders of magnitude ((Blouw, Choo, Hunsberger, & Eliasmith, 2019), Fig 1). Of particular interest to us are hardware elements [8] that perform neural network execution using spikes traveling through the network and models of neurons to perform the computation (Stuijt, Sifalakis, Yousefzadeh, & Corradi, 2021). Structured like a highly simplified model of the neurons in the human brain, neurons collect information (spikes) that come in from other neurons through synapses, which provide a means to control the strength of the incoming signal. The neuron integrates the information and, when excited sufficiently, emits a spike towards its axons, which are linked to other neurons. Several chips have been developed using this paradigm. One of their key features is their extremely low power usage, making them our ideal target hardware.

The rest of this paper is structured as follows: Section 2 discusses related work and Section 3, we present the background of Bayesian Networks for diagnostics and how such networks can be generated based upon FMEA analysis. We also discuss several modeling approaches and issues. In Section 4, we first give a short introduction into the basics of Spiking Neurons and their capabilities, before discuss our method to translate BNs into networks of spiking neurons. We will show how these networks of neurons can be implemented using the Nengo framework. We demonstrate the translation with a small BN and show simulation results. Section 5 concludes and presents directions for future work.

2. RELATED WORK

The section discusses current state-of-art research being done in implementing the framework to different applications. Each subsections gives an overview of specific steps involved in developing this approach. The section ends on the approach being taken in this research for onboard constrained complex systems operation.

2.1. Translation BN to SNN

In prior works (Pecevski, Buesing, & Maass, 2011) demonstrated method for translation of BN to SNN, that networks of spiking neurons can perform probabilistic inference through sampling in general graphical models. Additionally, (Yu et al., 2018) present a proof-of-principle for implementing Bayesian inference using distributed neural networks, offering a roadmap for large-scale Bayesian network implementation based on spiking neural networks while, (Zhang, Gu, Zheng, De, & Pan, 2020) suggests a method to train spiking neural networks indirectly by first training an Artificial Neural Network (ANN) and then converting it into an equivalent SNN. This approach can facilitate the translation of Bayesian networks into spiking neural networks by leveraging existing ANN training methodologies.

Spiking network model that performs Bayesian inference through sampling on neuromorphic platforms is being discussed in (Kungl et al., 2019).

2.2. Spiking NN for Diagnosis

Different kinds of spiking neural networks have been implemented for diagnosis and fault detection, e.g., (Wang et al., 2020) demonstrates the application of Weighted Fuzzy Reasoning Spiking Neural P Systems (WFRSNPSs) for fault diagnosis in power systems (Huang et al., 2021) proposes a fault analysis method based on modified fuzzy reasoning spiking neural P systems for fault prediction and diagnosis in motors, demonstrating the practical application of SNNs in fault detection.

2.3. Translation of FMEA to BN

(Ma, Zhou, Jiang, & Ding, 2014) proposes a novel method that integrates FMEA with Fault Tree Analysis (FTA) to construct Bayesian networks, offering a structured approach to incorporating FMEA data into Bayesian network models. (Tarcsay, 2024) introduces a hybrid method for risk-based fault detection using Bayesian networks based on FMEA. In the context of fault diagnosis, (Xu et al., 2019; Yang & Yu, 2013) process the application of Bayesian networks for fault diagnosis in hydroelectric generation systems and industrial processes, respectively.

2.4. Neuromorphic Hardware for fault detection and diagnosis

Neuromorphic hardware presents promising opportunities for fault detection and diagnosis as discussed in (Xiao, Chen, & Wang, 2022) which facilitates the efficient implementation of SNN-based applications through optimal mapping of Spiking Neural Networks (SNNs) to neuromorphic hardware. (Zhao, Donati, & Indiveri, 2020) highlight the fault-tolerant and biologically plausible nature of neuromorphic hardware for motor control applications, while (Titirsha et al., 2022) stress the importance of endurance-aware mapping of SNNs to neuromorphic hardware for robust fault detection systems. (Yerima, 2023) proposes a fault-tolerant spiking neural network mapping algorithm and architecture for 3D-NoC-based neuromorphic systems, enhancing fault tolerance in novel hardware architectures. Similarly, (Wade, McDaid, Harkin, Crunelli, & Scott Kelso, 2012) propose a concept of self-repair in neuromorphic systems, such as the bidirectionally coupled astrocyte-neuron system to demonstrate the potential for autonomous fault detection and repair mechanisms inspired by biological systems. These techniques could be used to enhance the reliability and robustness of our generated SNNs on neuromorphic hardware with, for example magnetic or analog memory with potentially higher error rates.

2.5. Approach

The current less computationally intensive onboard diagnostic approaches that rely on simple rule-based systems or threshold-based alarms. While these can be efficient for quick alerts, they often lack the depth and adaptability needed for comprehensive fault analysis.

Implementing the FMEA-SNN approach has both power, computational benefits, but it's crucial to highlight the conceptual advantages of root cause analysis (RCA) and model-based Bayesian analysis for onboard diagnostics. While simpler approaches may be sufficient for immediate fault detection, they often fall short of detecting and identifying the underlying causes of failures accurately.

While RCA delves deeper, systematically tracing the causality of events that lead to a malfunction, model-based Bayesian analysis, on the other hand, leverages a probabilistic framework to assess the likelihood of various failure modes. By incorporating prior knowledge and use real-time sensor data to adapt and refine its predictions, leading to more accurate diagnoses and prognoses. This approach is particularly valuable in complex systems with constrained power and computational resources.

Ultimately, the choice between approaches depends on the specific requirements and constraints of a given application. However, the conceptual advantages of RCA and model-based Bayesian analysis make them compelling options for systems where a deeper understanding of failures and more accurate predictions are critical as will be discussed in details in the next sections respectively.

3. BAYESIAN NETWORKS FOR DIAGNOSIS

Bayesian Networks (BNs) are directed acyclic graphs where nodes represent propositions or variables, the arcs represent the existence of direct causal influences between the linked propositions, and the strength of the causal relationships is represented through conditional probabilities (Pearl, 1985).

3.1. Bayesian Networks

An basic example of of BN as shown below, reproduced from (Pearl, 1985), is used as illustrative explanation of how BN works.

Figure 3 shows a representative BN, where the complete joint probability distribution $p\left(x_1, x_2, x_3, x_4, x_5, x_6\right)$ is the product of the conditional probabilities of each proposition given its ancestors, Eq. (1).

$$\begin{aligned} p\left(x_1, \ldots, x_6\right) =& p(x_6|x_5)\, p(x_5|x_2, x_3)\, p(x_4|x_1, x_2, x_3) \\ & p(x_3|x_1)\, p(x_2|x_1)\, p(x_1) \end{aligned} \tag{1}$$

The joint probability distribution could also be expressed with the short notation:

$$p\left(\boldsymbol{x}\right) = \prod_{j=1}^{n} p\left(x_j|\boldsymbol{a}_j\right) \ , \tag{2}$$

where $\boldsymbol{a}_j$ represents the set of ancestors of variable x_j, and $\boldsymbol{x}$ is the random vector containing all variables $x_1, \ldots, x_n$ (Pearl, 2000; Bobbio, Portinale, Minichino, & Ciancamerla, 2001). For example, the term $p(x_4|x_1, x_2, x_3)$ becomes $p(x_4|\boldsymbol{a}_4)$.

Dependencies among propositions are described through the definition of sets of ancestors (or parents) and descendants (or children). For example, the set $\{x_1, x_2, x_3\}$ contains the ancestors of x_4, while $\{x_2, x_3\}$ contains the children of x_1. This structural model allows analysis over interventions, i.e., enable the computation of the joint probability density function (pdf) conditioned on some specific assumptions over a specific variable in the network (Pearl, 2000). Starting from the example in Figure 3, it is possible to evaluate the joint pdf given, e.g., x_2 has been defined *True*:

$$\begin{aligned} p_{X_2=1}\left(x_1, x_3, \ldots, x_6\right) = & p(x_6|x_5)\, p(x_5|X_2 = 1, x_3) \\ & p(x_4|x_1, X_2 = 1, x_3)\, p(x_3|x_1) \\ & p(x_1)\,. \end{aligned} \tag{3}$$

The dependency of x_2 from x_1 has been removed in Eq. (3), since forcing $X_2 = 1$ does not depend on the value of x_1. Therefore, the edge connecting x_1 to x_2 should be removed to represent the proposed intervention. The challenge presented by BN is the assessment of all conditional probabilities of the system. Each node of the network requires a conditional probability table that defines the probability of the node being 1 (or 0) given all possible values of its ancestors. The dimension of the table increases in a combinatorial fashion with the number of ancestors.

3.2. Moving from FMEA to BN

In order to develop an efficient system level diagnosis procedure that takes uncertainty into account, both information from the FMEA and from the BN are combined. To this end, two main steps are involved: (i) build the BN structure and (ii) establish the conditional probability tables (CPT) of each node.

In order to construct the BN structure, the qualitative information about failure modes, causes and effects contained in the FMEA worksheet are transformed into nodes. As a starting point, observable nodes (simulating sensors), may represent simply events triggered by sensor signals, like, for example, a boolean variable for ‘‘temperature of the electronic speed controller too high”. Such information can help disambiguate among potential causes of the observed event. Then, the relationships between each node is defined by pointing arcs from causes to failure modes, and from failure modes to effects, resulting in the construction of the BN structure. A simple example of a BN structure built from FMEA is shown in Fig.2 for illustration.

Once the BN structure is built, the next step consists in assigning prior marginal probabilities to the root nodes (with no ancestors) and conditional probabilities to each of the other nodes, based on the qualitative information provided by the FMEA.

Probability values can be defined for example from historical failure data, expert knowledge about the probability of failure of each component, or by using maximum entropy theory as in (Gilabert, 2011).

Once the BN is completely defined, it can be used to detect and localize a fault within a complex system by turning observable nodes to *True* or *False*. The diagnosis procedure updates the probabilities using Bayesian inference in order to determine the root cause with the highest failure probability when an evidence (observable) node is triggered. The evidence nodes, also known as fault symptoms, are associated to observed variables such as sensor measurements, and can be for example triggered when the observed value exceeds a certain threshold.

Possible external sources that can affect the diagnosis procedure, such as environmental conditions or false alarm, are discussed in the next section.

3.3. Modeling approach and issues

The dependency among elements in FMEAs do not have to be restricted to deterministic relationships in BNs (Bobbio et al., 2001), and this property intrinsically enhances the modeling of the diagnostic system. Let us consider, for simplicity, a fault event f with two root causes, its ancestors, x_1 and x_2. Table 1 is the conditional probability table of the model, where probabilities are defined through three binary subscripts $i,\ j,\ k\ \in \{0, 1\}$. The term $p_{k|i\,j}$ defines the probability of the outcome k given values $i,\ j$, with k referring to the fault event f and i, j referring to its ancestors x_1 and x_2. For example, $p_{1|0\,0}$ is the probability that $f = 1$ given both ancestors $x_1,\ x_2$

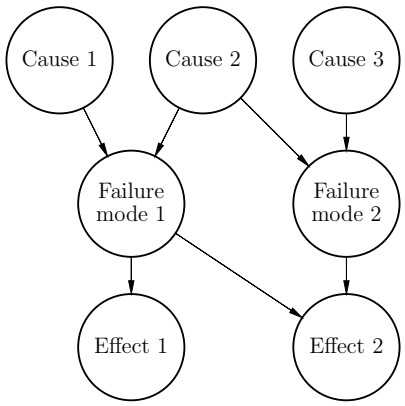

Figure 2. A basic Bayesian network structure derived from Failure Mode and Effects Analysis (FMEA).

Table 1. Example of conditional probability table for a fault event with two known root causes.

x_1	x_2	f 0	f 1
0	0	$p_{0\|0\,0}$	$p_{1\|0\,0}$
0	1	$p_{0\|0\,1}$	$p_{1\|0\,1}$
1	0	$p_{0\|1\,0}$	$p_{1\|1\,0}$
1	1	$p_{0\|1\,1}$	$p_{1\|1\,1}$

are 0 (or *False*).

The fault event may happen, with low probability, because of external causes or unknown events not described by its ancestors. Such external forcing are called *Common Cause Failures* (Bobbio et al., 2001), and following that idea, $p_{1|0\,0} \geq 0$, and so $p_{0|0\,0} = 1 - p_{1|0\,0}$. On the other side of the spectrum, the fault event may not happen even if both ancestors are activated (true). This option describes the ability of a system to work partially or reconfigure, (Bobbio et al., 2001), or describes a statistical relationship between the three elements, suggesting that root causes do not deterministically trigger the failure, so $p_{1|1\,1} < 1$. As a result, the two ancestors may occur without triggering the fault event, so $p_{0|1\,1} \geq 0$ and $p_{1|1\,1} = 1 - p_{0|1\,1}$. Different ancestors may influence the fault event in different ways, e.g. based on the severity of the root cause. This properties can be easily embedded in the network by assigning different values to the probabilities conditioned over $\{x_1 = 1, x_2 = 0\}$ and $\{x_1 = 0, x_2 = 1\}$.

In addition to the cases of failures induced by external variables or prevented system reconfiguration, the BN should also account for the performance of the measuring and/or detection system. In this architecture, the evidence used to perform inference over the network is collected through sensors that measure variables connected (directly or indirectly) to the fault event we aim to detect. The sensor performance or, similarly, the ability of the detection system to identify anomalous sensor data, should be embedded in the estimation of the CPT values. Reconnecting to the previous example, therefore, the element $p_{0|0\,0}$ in Table 1 should account for false alarm rates, and $p_{1|1\,1}$ should include, on top of any statistical relationship between the elements, the probability of mis-detection.

3.4. BayesNet Diagnostics

Developing an efficient diagnosis procedure involves two main steps: (i) build the BN structure and (ii) establish the conditional probability tables (CPT) of each node. The relationships between each node are defined by pointing arcs from causes to failure modes, and from failure modes to effects, resulting in the construction of the BN structure (Moreno-Bote & Drugowitsch, 2015). Once the BN is completely defined, it can be used to detect and localize a fault in a complex system by turning observable nodes to True or False, based on the qualitative information provided by the FMEA. The diagnosis procedure updates the probabilities using Bayesian inference to determine the root cause with the highest failure probability when an evidence (observable) node is triggered (Davies et al., 2018). The dependency among elements in FMEAs do not have to be restricted to deterministic relationships in BNs [4].

4. SPIKING NEURONS AND BAYESIAN NETWORKS

Traditional Bayesian Belief Reasoning algorithms as discussed above are typically executed on CPU, GPU, or FPGA. Since numerous floating-point operations are needed for each reasoning step, their computational footprint is substantial, which is prohibitive in a power-stripped environment like a typical space asset.

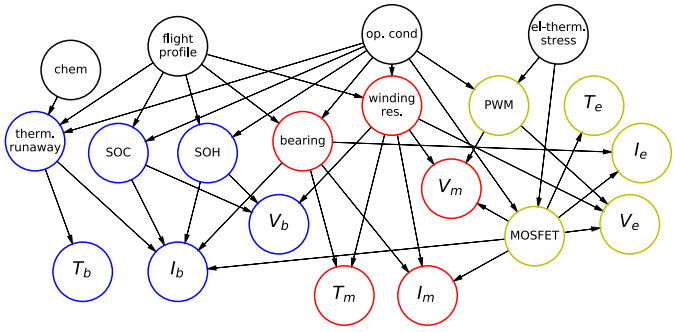

Figure 3. Case study showcasing the development and application of a Bayesian network for the electric powertrain of a UAV.

4.1. Spiking Neurons

We therefore take inspiration by looking at the human brain: here, information is processed and complex sense-making takes place, based upon a highly structured network of *spiking* neurons. The input from all synapses are processed by the cell body of the neuron, and if enough input has been received and the neuron is in the right state, it produces an output *spike* on its axon, which is then fed into synapses of other neurons.

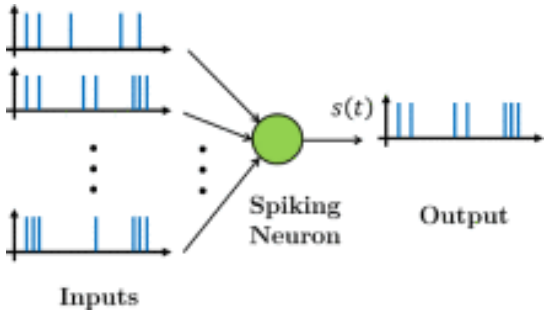

Figure 4. Spiking Neuron (on abstract level): input spike sequences are integrated in the neuron to produce an output spike train

The firing mechanism of real biological neurons is highly com-

plex (Gerstner, Kistler, Naud, & Paninski, 2014) but even a strong simplification, which we are using here has the following characteristics:

- information between neurons is not passed as floating-point values, but rather as a sequence of spiking events (see Figure 4).
- computational characteristics of the network is defined by the strength of the connections between neurons (''synaptic weights'') and internal neuron parameters
- typical brain structures are highly structured, in contrast to artificial neural networks, which consist of few (10s to 100s) of highly interconnected layers, requiring large memory and a huge amount of floating-point operations

Due to these characteristics, reasoning and computations can be carried out with extremely low power consumption. Tool sets exist to map Deep Neural Networks into Spiking Neural Networks.

In this work, our SPIKE-DX reasoning and monitoring component is, as described above, a Bayesian Network. It contains all the necessary information to perform its diagnosis and monitoring task.Further develop a direct translation of a Bayesian network into a Spiking Neuron architecture. Probabilities are represented as spiking rates: a low probability corresponds to low firing rate (low frequency of spiking events), whereas a high probability causes frequent firing.

As shown in Figure 1, we generate a network structure that, when provided with a rate-coding of inputs at the observable nodes, produces spike sequences at the neurons of the unobservable diagnosis and sense-making nodes, which correspond to the posterior probability according to a Bayesian inference.

In order to accomplish Bayesian Reasoning steps, we are using two different kinds of neurons: excitatory neurons (E), which summarize and integrate their synaptic input, and inhibitory neurons (I), which produce spikes that weaken or even disable output spikes of their target neurons. A balanced interplay of E and I neurons is necessary to realize the Bayesian inference steps.

We will explain the translation method using a simple Bayesian network with only four nodes as shown in Figure 5. The nodes, labeled V3 and V4 are observable (input) nodes. Our desired output node sits on the top of this network (V1). The information on the V2 node can ''explain away'' the reasoning link between V3 and V1.

4.2. Translation

For the actual translation of the BN, we use a method inspired by (Moreno-Bote & Drugowitsch, 2015; Paulin & van Schaik, 2014). Each of the motifs of the BN are translated individually in a compositional way. Figure 6 shows the translation of the small BN in Figure 5. White triangles are the excitatory

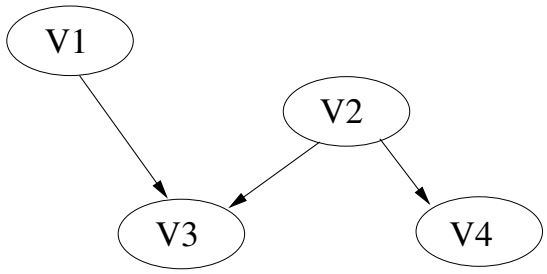

Figure 5. A basic Bayesian network demonstrating the phenomenon of ''explaining away'' motif

neuron groups for the input and outputs (V3, V4, V1); other excitatory neurons are marked in green. Inhibitory neurons are represented by red triangles; their output connections (dashed) dampen or inhibit the spiking behavior of their target neurons.

Since each of the Bayesian nodes have discrete states and a discrete conditional probability table (see Table 1), we need to represent each state by a group of neurons. In this case with two root causes V3 and V4, we would need $2^2 = 4$ groups of neurons to represent the state. The nodes a00,a01,a10,a11 in Figure 6 correspond to these states. Since only one of these states can be active, lateral inhibition, which is realized by the lateral inhibitory neurons (located to the right of the 'a__' neurons). For example, the activity of a00 activates the inhibitory neuron lat00 (top right), which in turn inhibits the firing of the other 'a__' neurons a01,a10,a11 using inhibitory (dashed line) connections.

The different probabilities are realized by setting the (excitatory) connection strengths accordingly. This will lead to different firing rates as averaged over all neurons in the corresponding group.

The translation of a motif (e.g., the one shown in Figure 5) also requires ''mixing'' connections that model the merging of the different inputs. Here, the outputs of node V_3 and V_4 are merged to activate the corresponding state a00,a01,a10,a11. Here again, excitatory and inhibitory connections play a key role. The generated networks for each individual motif can be connected together to form the network for the evaluation of the Bayesian network.

4.3. SNN Implementation

There are numerous simulation systems and dynamical models for spiking neurons. For our approach, we do not require elaborate neuron models, rather, we base out model on simple ''integrate & fire'' (IF) neurons. We therefore use the Nengo[1] development environment. This open-source environment is based upon the NEF (Neuro Engineering Framework) (Stewart, 2012) paradigm and allows parameterized and modular definitions of groups of neurons. The tool is imple-

[1] https://www.nengo.ai

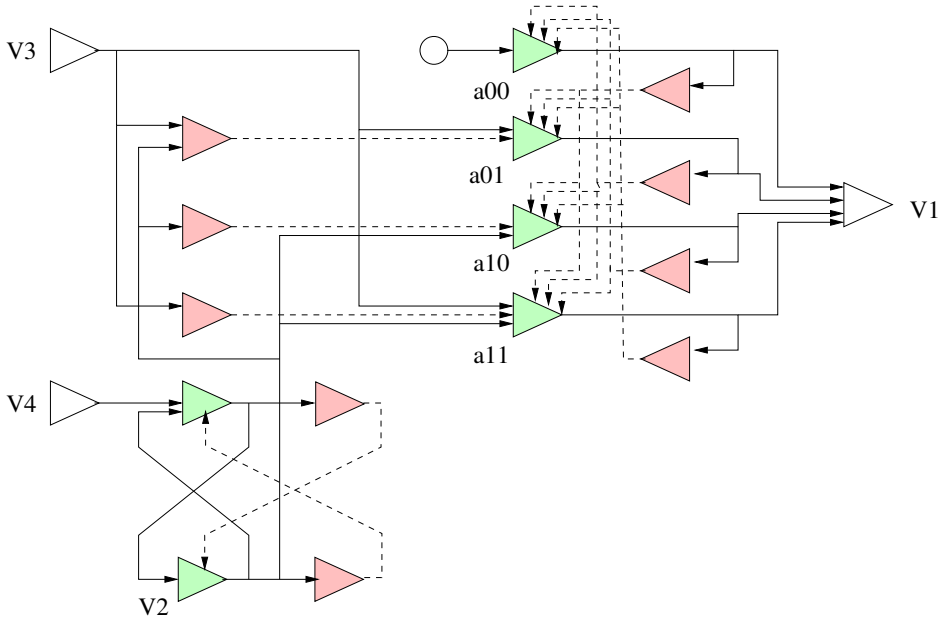

Figure 6. Translation of BN in Figure 5 into a network of spiking neurons

mented in Python. The generated networks can be simulated in software, or compiled to the Loihi neuromorphic processor (Davies et al., 2018), which has an extremely low power consumption.

Within the Nengo environment, we define our resulting SNN as a set of interconnected neurons (''Node'') or groups of identical neurons (''Ensemble''). In our experiments, we used groups with $N = 30$ neurons for each functionality. There are excitatory and inhibitory neurons. In our simplified model, excitatory and inhibitory neuron are of the same type; only their axional output has a negative sign.

An excerpt of our generated network (Figure 6 is shown in Listing 1.

Connection between individual neurons or groups of neurons are defined using the nengo.Connection (..) method. Connection strengths, connection topology, and transmission functions can be varied[2].

The nodes a00,a01,a10,a11 (in Figure 6) correspond to the internal neurons. Since only one of these can be active, lateral inhibition, which is realized by the lateral inhibitory neurons lat00 ,..., lat11. Listing 1 shows, how the activity of a00 activates the inhibitory neuron lat00, which in turn inhibits the firing of all neurons representing the other nodes a01,a10,a11. Note the negative value of the connection strength S_{inh}.

Listing 1. BN tanslated into SNN in Nengo framework (excerpt)

```
import nengo
model = nengo.Network(label="Full■4-node■BN")
N=10; S= -5.0
with model:
    I_v3 = nengo.Node(0)  # "input" neurons
    I_v4 = nengo.Node(0)  # "input" neurons

        # auxiliary neurons
    a00 = nengo.Ensemble(N, dimensions=1)
    a01 = nengo.Ensemble(N, dimensions=1)
    a10 = nengo.Ensemble(N, dimensions=1)
    a11 = nengo.Ensemble(N, dimensions=1)

    l00 = nengo.Ensemble(N, dimensions=1)
...
        # lateral inhibition
    lat00 = nengo.Ensemble(N, dimensions=1)
    lat01 = nengo.Ensemble(N, dimensions=1)
    lat10 = nengo.Ensemble(N, dimensions=1)
    lat11 = nengo.Ensemble(N, dimensions=1)

        # connections: input to auxiliary
    nengo.Connection(I_v3, a11)
    nengo.Connection(I_v3, l10)
    nengo.Connection(I_v3, a01)
    nengo.Connection(I_v3, l00)
        # lateral inhibition between a00,...,a11
        #  via inhibitory neurons
    nengo.Connection(a00, lat00)
    nengo.Connection(lat00, a01.neurons,
          transform=[[S_inh]] * N)
    nengo.Connection(lat00, a10.neurons,
          transform=[[S_inh]] * N)
    nengo.Connection(lat00, a11.neurons,
          transform=[[S_inh]] * N)
...
```

Figure 7 shows the signal traces over time during a simulation within the Nengo environment. The network is provided with input signals V3 and V3 (top panel), which change over

[2] https://nengo.ai/getting-started

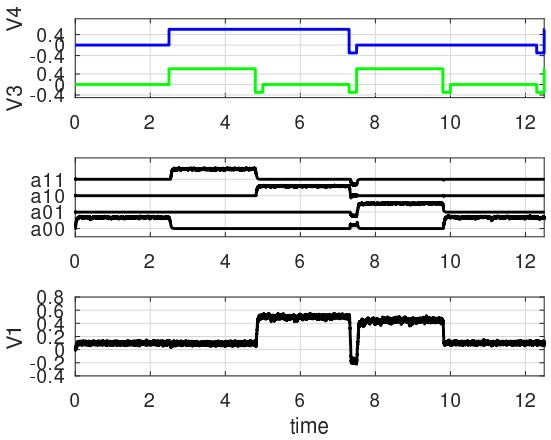

Figure 7. Simulation of network in Figure 6B in the Nengo framework

time. The middle panel shows the activity of the intermediate neurons a00,a01,a10,a11 as the mean of the firing rates of all neurons in the corresponding group. Due to the lateral inhibition, only one of these 4 neuron groups show activity. Finally, the bottom panel shows the activity at the output of the SNN. Only input combinations of V3=0, V4=1, or V3=1, V4=0, cause high output activity. Due to different example settings of the weights, (0.8 vs 0.75), corresponding to different probabilities, cause different activity amounts.

5. CONCLUSION

In this paper, we presented how a diagnostic Bayesian network, produced out of a FMEA analysis can be translated into a Spiking Neural Network, which performs Bayesian Diagnostic reasoning. Our translation of a Bayesian network into a Spiking Network exhibits a number of unique advantages:

- The SN can be executed on a highly power-restricted platform. All information is presented as rates (or sequences of spikes). Therefore, the number of operations necessary and thus the power consumption can be substantially reduced. This is even more evident when neuromorphic spiking hardware is used (see below) and the fact that only integer (fixed-point) operations are needed. Often, a low reasoning rate (e.g., 10Hz) is needed, which can reduce the computational footprint even further
- some sensors might be able to directly produce spike trains or events without the need of (power-intensive) pre-processing by the CPU.
- Spike trains can be exchanged affectively between different members of a distributed system of space assets, e.g., a swarm of micro satellites
- Our BN is highly modular. Therefore it can be (a) easily constructed, (b) is human understandable, and (c) its corresponding SN can be executed efficiently, since the modularity is preserved.
- Our SN has fixed synaptic weights. No training is needed. Our approach starts from a given Bayesian network, that is being constructed using proven engineering approaches. Individual probabilities can be considered in isolation and can often be provided by the component vendor.
- Our BN is therefore human understandable and easy to verify and validate. Its translation into an SNN is deterministic, which means, that no additional complications for validation and certification are introduced, as no training or adaptation is taking place.

Future work will include the execution of the spiking network on a power-saving neuromorphic hardware, optimization of the translation process, study on robustness and scalability, as well as the development of neural networks that can produce the spiking sequences directly from sensor inputs, thus eliminating power-consuming CPU-based signal preprocessing.

ACKNOWLEDGMENT

This work was authored by employees of KBR Wyle Services, LLC under Contract No. 80ARC020D0010 with the National Aeronautics and Space Administration. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, worldwide license to reproduce, prepare derivative works, distribute copies to the public, and perform publicly and display publicly, or allow others to do so, for United States Government purposes. All other rights are reserved by the copyright owner.

REFERENCES

Blouw, P., Choo, X., Hunsberger, E., & Eliasmith, C. (2019). Benchmarking keyword spotting efficiency on neuromorphic hardware. In *Proceedings of the 7th annual neuro-inspired computational elements workshop*. New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3320288.3320304 doi: 10.1145/3320288.3320304

Bobbio, A., Portinale, L., Minichino, M., & Ciancamerla, E. (2001). Improving the analysis of dependable systems by mappping fault trees into bayesian networks. In *Reliability engineering and systems safety* (Vol. 71, p. 249-260).

Christensen, D. V., Dittmann, R., Linares-Barranco, B., Sebastian, A., Gallo, M. L., Redaelli, A., ... Pryds, N. (2022, may). 2022 roadmap on neuromorphic computing and engineering. *Neuromorphic Computing and Engineering*, 2(2), 022501. Retrieved from https://dx

.doi.org/10.1088/2634-4386/ac4a83 doi: 10.1088/2634-4386/ac4a83

Corbetta, M., & Kulkarni, C. S. (2019, September). An approach towards uncertainty quantification and management of unmanned aerial vehicle health. In S. Clements (Ed.), *Annual conference of the prognostic and health management society.* Retrieved from https://doi.org/10.36001/phmconf.2019.v11i1.847

Davies, M., Srinivasa, N., Lin, T.-H., Chinya, G., Cao, Y., Choday, S. H., ... Wang, H. (2018). Loihi: A neuromorphic manycore processor with on-chip learning. *IEEE Micro*, *38*(1), 82-99. doi: 10.1109/MM.2018.112130359

Fang, H., Shi, H., Dong, Y., Fan, H., & Ren, S. (2017). Spacecraft power system fault diagnosis based on dnn. In *2017 prognostics and system health management conference (phm-harbin)* (p. 1-5). doi: 10.1109/PHM.2017.8079271

Gerstner, W., Kistler, W. M., Naud, R., & Paninski, L. (2014). *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press.

Gilabert, A. G. E. (2011). Mapping fmea into bayesian networks. *International Journal of Performability Engineering*, *7*(6), 525--537.

Hogge, E., Bole, B., Vazquez, S., Kulkarni, C., Strom, T., Hill, B., ... 8, C. Q. (2018). Verification of prognostic algorithms to predict remaining flying time for electric unmanned vehicles. In *International journal of prognostics and health management, issn 2153-2648, 2018 021*.

Huang, Z., Wang, T., Liu, W., Valencia-Cabrera, L., Pérez-Jiménez, M. J., & Li, P. (2021). A fault analysis method for three-phase induction motors based on spiking neural p systems. *Complexity*. doi: 10.1155/2021/2087027

Kungl, A. F., Schmitt, S., Klähn, J., Müller, P., Baumbach, A., Dold, D., ... Petrovici, M. A. (2019). Accelerated physical emulation of bayesian inference in spiking neural networks. *Frontiers in Neuroscience*. doi: 10.3389/fnins.2019.01201

Ma, D., Zhou, Z., Jiang, Y., & Ding, W. (2014). Constructing bayesian network by integrating fmea with fta. doi: 10.1109/imccc.2014.148

Moreno-Bote, R., & Drugowitsch, J. (2015, december). Causal inference and explaining away in a spiking network. *Sci Repo*, *5*(17531).

Paulin, M. G., & van Schaik, A. (2014). *Bayesian inference with spiking neurons.*

Pearl, J. (1985). Bayesian networks: A model cf self-activated memory for evidential reasoning. In *7th conference of the cognitive science society.*

Pearl, J. (2000). *Causality: models, reasoning and inference*. MIT Press Cambridge, MA.

Pecevski, D., Buesing, L., & Maass, W. (2011). Probabilistic inference in general graphical models through sampling in stochastic networks of spiking neurons. *Plos Computational Biology*. doi: 10.1371/journal.pcbi.1002294

Rao, R. P. (2004). Hierarchical bayesian inference in networks of spiking neurons. In L. Saul, Y. Weiss, & L. Bottou (Eds.), *Advances in neural information processing systems* (Vol. 17). MIT Press. Retrieved from https://proceedings.neurips.cc/paper_files/paper/2004/file/38181d991caac98be8fb2ecb8bd0f166-Paper.pdf

Stewart, T. C. (2012). *A technical overview of the neural engineering framework* (Tech. Rep.). Centre for Theoretical Neuroscience.

Stuijt, J., Sifalakis, M., Yousefzadeh, A., & Corradi, F. (2021). μbrain: An event-driven and fully synthesizable architecture for spiking neural networks. *Frontiers in Neuroscience*, *15*. Retrieved from https://www.frontiersin.org/journals/neuroscience/articles/10.3389/fnins.2021.664208 doi: 10.3389/fnins.2021.664208

Tarcsay, B. L. (2024). Risk-based fault detection using bayesian networks based on failure mode and effect analysis. *Sensors*. doi: 10.3390/s24113511

Titirsha, T., Song, S., Das, A., Krichmar, J. L., Dutt, N., Kandasamy, N., & Catthoor, F. (2022). Endurance-aware mapping of spiking neural networks to neuromorphic hardware. *Ieee Transactions on Parallel and Distributed Systems*. doi: 10.1109/tpds.2021.3065591

Wade, J., McDaid, L., Harkin, J., Crunelli, V., & Scott Kelso, J. A. (2012). Self-repair in a bidirectionally coupled astrocyte-neuron (an) system based on retrograde signaling. *Frontiers in Computational Neuroscience*. doi: 10.3389/fncom.2012.00076

Wang, T., Wei, X., Wang, J., Huang, T., Peng, H., Song, X., ... Pérez–Jiménez, M. J. (2020). A weighted corrective fuzzy reasoning spiking neural p system for fault diagnosis in power systems with variable topologies. *Engineering Applications of Artificial Intelligence*. doi: 10.1016/j.engappai.2020.103680

Xiao, C., Chen, J., & Wang, L. (2022). Optimal mapping of spiking neural network to neuromorphic hardware for edge-ai. *Sensors*. doi: 10.3390/s22197248

Xu, B., Li, H., Pang, W., Chen, D., Tian, Y., Lei, X., ... Patelli, E. (2019). Bayesian network approach to fault diagnosis of a hydroelectric generation system. *Energy Science & Engineering*. doi: 10.1002/ese3.383

Yang, F., & Yu, W. (2013). Multi-fault diagnosis for industrial processes based on hybrid dynamic bayesian network. doi: 10.2316/p.2013.794-053

Yerima, W. Y. (2023). Fault-tolerant spiking neural network mapping algorithm and architecture to 3d-noc-based neuromorphic systems. *Ieee Access*. doi: 10.1109/

access.2023.3278802

Yu, Z., Huang, T., & Liu, J. K. (2018). Implementation of bayesian inference in distributed neural networks. In *2018 26th euromicro international conference on parallel, distributed and network-based processing (pdp)* (p. 666-673). doi: 10.1109/PDP2018.2018.00111

Zermani, S., Dezan, C., Chenini, H., Diguet, J.-P., & Euler, R. (2015). Fpga implementation of bayesian network inference for an embedded diagnosis. In *2015 ieee conference on prognostics and health management (phm)* (p. 1-10). doi: 10.1109/ICPHM.2015.7245057

Zhang, M., Gu, Z., Zheng, N., De, M., & Pan, G. (2020). Efficient spiking neural networks with logarithmic temporal coding. *Ieee Access*. doi: 10.1109/access.2020.2994360

Zhao, J., Donati, E., & Indiveri, G. (2020). Neuromorphic implementation of spiking relational neural network for motor control. doi: 10.1109/aicas48895.2020.9073829

Zuo, L., Zhang, L., Zhang, Z.-H., Luo, X.-L., & Liu, Y. (2021). A spiking neural network-based approach to bearing fault diagnosis. *Journal of Manufacturing Systems*, *61*, 714-724. Retrieved from `https://www.sciencedirect.com/science/article/pii/S0278612520301138` doi: https://doi.org/10.1016/j.jmsy.2020.07.003