(IJRST) 2018, Vol. No. 8, Issue No. IV, Oct-Dec **e-ISSN: 2249-0604, p-ISSN: 2454-180X**

LEVERAGING THE NEST, GENN, EDLUT, AND BRIAN TOOLS IN MITIGATING THE COST AND ENHANCING OF THE COMPUTATIONAL SPEED

ROMHARSH MITTAL

ABSTRACT

Objectives: To review various tools available for simulating Spiking Neural Networks using heterogeneous parallel processing platforms that help to reduce cost, increase the computational speed, and also to document/archive lessons learned.

Methods/Statistical Analysis: The computational speed is a continuing challenge for simulating good spiking neural network models. Understanding of the spiking neural networks is significantly simplified by computer simulators like NEST, GeNN, EDLUT, and BRIAN.

Findings: Simulation is a handy toolkit of scientists and engineers of all disciplines. NEST, GeNN, EDLUT, and BRIAN simulators help in achieving better performance not in terms of the same kind of processing but with additional particular tasks that require more computational power. BRIAN and EDLUT, which are hybrid simulators, supports both time-driven and event-driven techniques and outperform when compared to other simulators.

Application/Improvements: Using BRIAN and EDLUT simulation techniques, we can achieve high performance when compared to other spiking neural simulation techniques.

INTRODUCTION

The simulation of Spiking Neural Networks (SNN) usually is a difficult task that involves a large number of calculations, particularly while handling large scales SNNs. The limitations emanate primarily from the demands on computer resources such as the Modern Graphics Processing Unit (CPU) and storage. As SNNs become bigger with added complexity, the required processing power increases exponentially. The designs of SNN simulators facilitate the decomposition of computational load into several parts that can be processed on many processors. The systematic mapping of the computational load to the processing elements reduces the complexity of implementing simulation techniques and improves the efficiency of the simulator with accelerated computational processes. GPUs are low-cost supercomputers devoted to handling massively parallel computations that can be used as computational accelerators .Of late, GPU enabled massively parallel processing has attracted significant interest of the researchers given the possibilities of dramatically accelerating computations using off-the-shelf GPU modules. The models or simulations built on GPUs using parallel programming paradigms result in accomplishing quicker results in scientific and engineering research. Using Open Computing Language (OpenCL) or

Compute Unified Device Architecture (CUDA) solutions to various problems of the real world can be implemented quickly and run faster compared with multi-core or multiprocessor systems. Building clusters of heterogeneous computing cores is another promising approach for data processing and parallel computation in enterprise computing.

SPIKING NEURAL NETWORKS MODELLING

The spiking neural network simulation is close to realism in the third generation of neural networks. In addition to the neuronal and synaptic states, SNNs also integrate the idea of time into the operating model. The idea is that neurons in SNN do not fire at every propagation cycle but rather fire only when the membrane's charge touches a specific neuron. When the neuron fires, it produces the signal which passes to another neuron, which in turn, rises or reduces its potentials concerning this signal.

The simulation of SNNs is a most inspiring computational job because of its numerical calculations and the simulation time. For a real-time simulation, a large number of numerical calculations need to be massively parallelized. A basic neural network is shown in Figure 1, in which the neurons are specified by the label Nj, and the presynaptic relations Xi of the neuron are signified as small circles with weight

Wi. The synaptic weight calculation formula is as follows:

Synaptic weight[j] = $\sum_{1}^{n} X_i W_i$

The present activation state in SNN simulation is usually measured as neuron's state, with incoming spikes passing spike values to the next level. Different coding approaches are available for understanding the departing spike chain as a real significance number, either trusting on the regularity of spikes or timing between spikes and to encode data. This type of neural network is used.

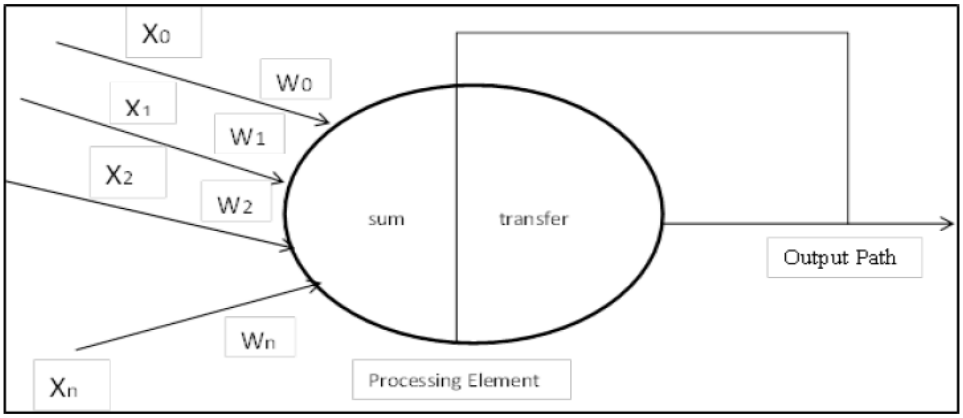

Figure 1. Simplified spiking neural network.

For information processing requirements as the technique similar to the customary artificial neural networks. The SNNs can model a computer-generated central nervous structure. They can also be used to learn the process of biological neural networks. The real-world use of large scale SNNs is limited due to the exponential increase of computational price, which is related to the accurate simulation of neural models with computational power. As a result, there has been limited use of large scale SNNs to resolve computational jobs dealt with by second-generation neural networks. It is challenging to adjust the second generation, neural models, to real-time SNNs. However, it is easy to create and observe its dynamics but difficult to improve with steady performance. Spiking neural networks acquire biological patterns; the nervous structure can train processes that usually occur in the brain, but they are not entirely understood.

Mainly, an SNN simulator permits the study of the theory that helps in the review of procedures such as:

• Validation of how a theoretical model mimics a biological system,

• Research of mental sicknesses by imitating brain sicknesses, and

• Analysis of how medicine disturbs the brain.

Research involving SNNs aims to learn how the human brain functions. Since the beginning of computer science, computers have constantly spent a high level of computational power for simulating biological networks. Still, they are not close to mimic the nature they created. The computational principles of the human brain are modeled in artificial neural networks, which permit there modeling of human capabilities. Thus, networks are used for various machines learning jobs such as pattern recognition and process estimations. The neurons essentially transfer spikes with other neurons by electrical pulses. The spikes transferred by neurons are very accurate and, they are known as Spike Timing Dependent Plasticity (STDP). There are three different types of spiking neuron models, namely, **(i)** Integrate and Fire Neuron Model, **(ii)** The Hodgkin Huxley (H&H) Neuron Model, and **(iii)** The Izhikevich Neuron (I&N) Model.

The Hodgkin Huxley(H&H) Neuron Model

The Hodgkin Huxley model or conductance-based model is a mathematical model that describes action potentials in neurons that are initiated and propagated in the neurons. It is a set of nonlinear differential equations that approximates the characteristics of excitable cells such as neurons, which is a continuous-time model. The neuron layer is signified as a conductance represented by Cm. The motivating gradients initiate the movement of neurons that are signified by the DVM (Membrane Voltage source), whose dvm value is identified using the ratio of the intra and extracellular absorptions of neurons. Current source Ic is represented by:

$Ic = Cm(DVM/dt)$

The Izhikevich Neuron (I&N) Model

The Izhikevich neuron model simulation is costly and complex to simulate SNNs because it involves various complex differential equations. Equations that are capable of duplicating various rich firing

patterns attainable with the Hodgkin Huxley neuron model are used to stimulate neurons. This simulation model exhibits high performance, similar to the I&F model.

POPULAR SPIKING NEURAL NETWORK SIMULATORS

The SNNs Simulating on CPU

A study of SNNs simulation platforms on CPU is reported in the literature. Also, the NEURON simulator is widely used because of its support for the design and evaluation of different SNN models. The NEURON simulator combines both event-driven and time-driven simulation modes. Furthermore, this simulator is suitable for performing parallel processing on both multiprocessor and multi-core systems by distributed processes and threads in system clusters as per MPI standards. This is available for UNIX, MS Windows, and Linux environments. It is also available for IBM Blue Gene and Cray supercomputers. The Neural Network Simulation Tool (NEST) is developed as the outcome of collective advanced technology projects for the simulation of spiking neural networks. It is intended to simulate the large scale SNNs with the heterogeneity in both synapses and neuron types' simulations. Parallel processing is achieved on multiprocessor systems and clusters of systems through threading and message passing. Simulator outfits a time-driven simulation method and is available for Linux, UNIX, Mac OS, and MS Windows environments. The NEST simulator is released to the scientific community with an open-source license. The Brian simulator is extremely exile and extensible for the simulation of SNNs on all operating system environments, including MS Windows, Linux, and Mac OS. This simulator is very popular in the research community since it is easy to use and learn and can simulate various spiking neuron models such as H&H and I&F and can be extended to support other models. Brian simulator is coded in Python to take advantage of different techniques coded and released as Python libraries. Some such examples are SciPy and NumPy that can be used for statistical designs and PyLab that can be used for graphical visualization.

Python can also be useful in the parallelization processes.

Mvaspike is a general-purpose tool for simulating vast and complex spiking neural networks. This simulator ai equipped with an event-driven simulation approach with an emphasis on SNNs simulation. A new balance between simulation efficiency and the modeling liberty is provided using this simulation technique. The simulator is implemented using C++ though the use of the simulator from different programming language environments is not tricky. Parallel execution is also available for multiprocessor systems and clusters, as shown in Table.

Feature	BRIAN	NEST	NEURON	NeMo
Time Driven	Yes	Yes	Yes	Yes
Event Driven	Yes	No	Yes	No
GPU	Yes	No	No	Yes
Linux	Yes	Yes	Yes	Yes
Windows	Yes	No	Yes	No
Easy installation	Yes	Yes	No	No

Table 1. Select spiking neural network simulators

SNNs Simulation Using GPU

We have investigated various platforms used for simulating the SNNs on CPU in the previous section. The excellent performance NeMo platforms popularly used for the simulation of large scale SNNs. NeMo simulator supports various SNN models such as I&F and Izhikevich neuron models enabled by CUDA on GPU and implements an algorithm called scatter-gather messaging. The platform promotes additional optimization to the unplanned sparse connectivity and the actual simulations. Furthermore, the simulator supports Matlab, C, and Python. NeMo simulator is released with an open-source license. The GPU Enhanced Neuronal Networks (GeNN) simulator facilitates features to simulate SNNson; a GPU equipped system. This simulator is an open-source archive developed using CUDA, and C/C++ accelerates the performance of the neural simulation using NVIDIA GPU cards. GeNN simulator is extendable in the sense that every neuron structure can be simulated using this simulator. In GeNN simulation, researchers can host their specific synapse models, neuron models, and integration models, which are automatically replaced with neural simulation models through code generation. The GeNN simulator is available for Mac OS, Linux, and MS Windows environments. The Myriad simulator focuses on simulations such as H&H neuron models using CUDA on GPU and also supports simulations on clusters. Myriad simulator delivers an extensible and flexible interface with Python, which is later interpreted into a C, based code.

HYBRID SIMULATION OF SPIKING NEURAL NETWORKS

The hybrid simulation techniques, such as event-driven simulators, relatively use simple neural network models defined by various differential equations. These simulations are carried out frequently to observe random spike response times. The hybrid simulator, such as Event-Driven Lookup Table (EDLUT), which is an open-source simulator, was designed to simulate very large-scale spiking neural networks. The lookup tables in the EDLUT simulator is used to store all expected values of spike fire times. Thus, the total SNN model simulation is coded as a set of classification tables. The tool uses Runge Kutta techniques for the numerical computations, and fast simulation of large scale SNNs are feasible in the EDLUT's computational processes. This simulator

supports both event-driven and time-driven methods, thus making this technique a suitable hybrid technique for simulating the biological spiking neural networks.

Simulation Techniques for Hybrid CPU-GPU Architecture

The EDLUT simulation can be classified into three computational steps:

1. The neuronal dynamics;

2. The spike propagation; and

3. The event of queue management.

Hence, the EDLUT simulation time is classified into time neu dynamics, time spike prop, and time queue manage, correspondingly. The simulator uses methods like Integrate and Fire (I&F) Neuron Model and Hodgkin Huxley (H&H) Neuron Model to increase the dynamic neuron computations, which makes EDLUT simulation to achieve excellent performance when the neural network integration stage: **(i)** succeeds over **(ii)** and, **(iii)** listed above.

The time-driven parallelizing techniques in CPU with GPU improve the performance of the simulation while preserving its correctness and flexibility. The parallelizing technology uses the integration periods, therefore, gaining more accurate results. Additionally, event-driven simulation techniques are best for the simulation of simple neuron network models with extraordinary performance. In the computational process, GPUs are known to speed up computations for the time-driven simulation techniques because of their parallel design. The GPUs rely on CPU for their scheduling of computations; while the CPU initiates the simulation, the GPU completes the complete simulation process by executing the scheduled calculations. This means that though EDLUT is successively running, GPU evaluates the neural variables in time-driven techniques, whereas CPU produces, propagates spikes, and practices are learning procedures in both time-driven and event-driven techniques.

SUMMARY AND CONCLUSION

The simulation technique in spiking neural networks plays an essential role in modern research, whether it is pure or applied sciences. The complexity of biological spiking neural networks, subsystems, and interactions among them demand enormous computing resources to simulate them. Advancements in heterogeneous parallel processing have helped in handling the complexities of simulating the biological spiking neural systems by utilizing the multi-core CPU and many-core GPU architectures.

Heterogeneous parallel processing for the simulation of biological systems in medical applications involves the developing of algorithms, data structures, and other tools to model biological systems and implementing them on heterogeneous parallel processing platforms. The study provides an overview of simulators for spiking neural networks that have good potential for modeling and simulation using different parallel processing platforms.