Design and Simulation of Spiking Programmable Neural Computers

https://github.com/mkturkcan/spikingprogrammableneuralcomputers

Mehmet Kerem Turkcan
Department of Electrical Engineering, Columbia University, USA

Overview

We consider the design of spiking neural machines capable of implementing Turing machines, and using the design principles considered we propose a programmable spiking neural computer that can be programmed by providing external input. In the computer, specific subcircuits openly represent instructions and variables.

We extend the architecture to allow for task parallelism via the fork-join model, which can be used to add arbitrary SNNs as computational submodules into a computer.

In addition to the design, we implement a simulator for our computer that can be executed efficiently on GPUs. Our approach explicitly gives a computational meaning to each neuron in terms of the instruction it implements or in regards to the variable register it represents.

In this work:

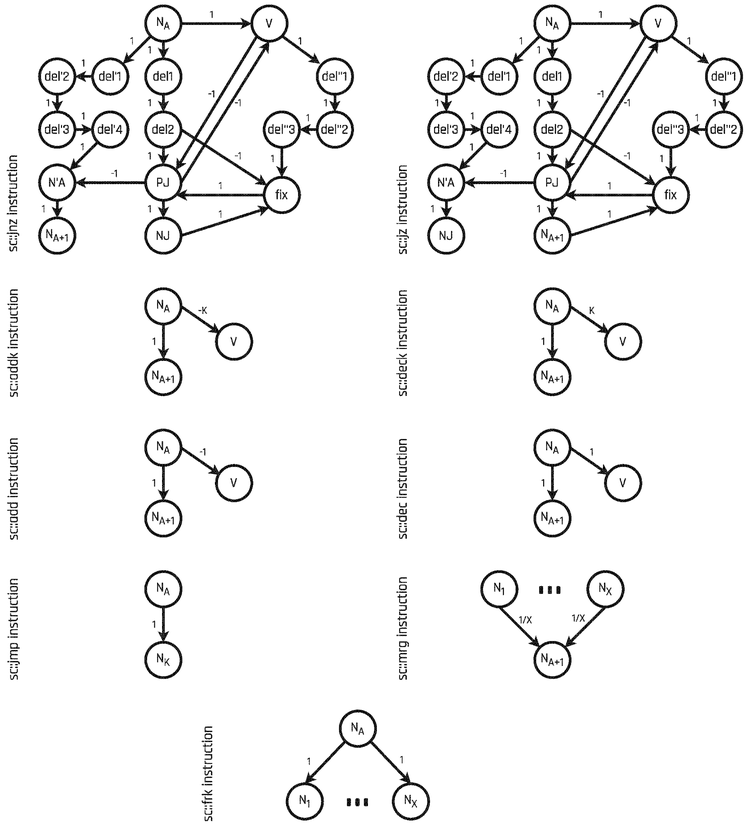
1. We present a set of instructions for a programmable spiking neural computer. The instruction set we use is sufficient to implement any Turing computable function [1].
2. We release an open source Python library that allows for the construction of a spiking computer with variable number of registers as well as its simulation with desired inputs.

We present a method to represent Turing machines with spiking neural networks, and use this method to create spiking programmable computers that can be programmed by hand.

Instructions for a Spiking Computer

In [1], an approach was put forward to represent any Turing machine with RNNs using 4 instructions. We extend their approach to use spiking neural network components, namely, integrate-and-fire (IAF) neurons. We implement jmp, jnz, add, and sub instructions required to implement any Turing computable function. In addition to these operations, we provide a number of additional operations that can be efficiently executed on neural computers.

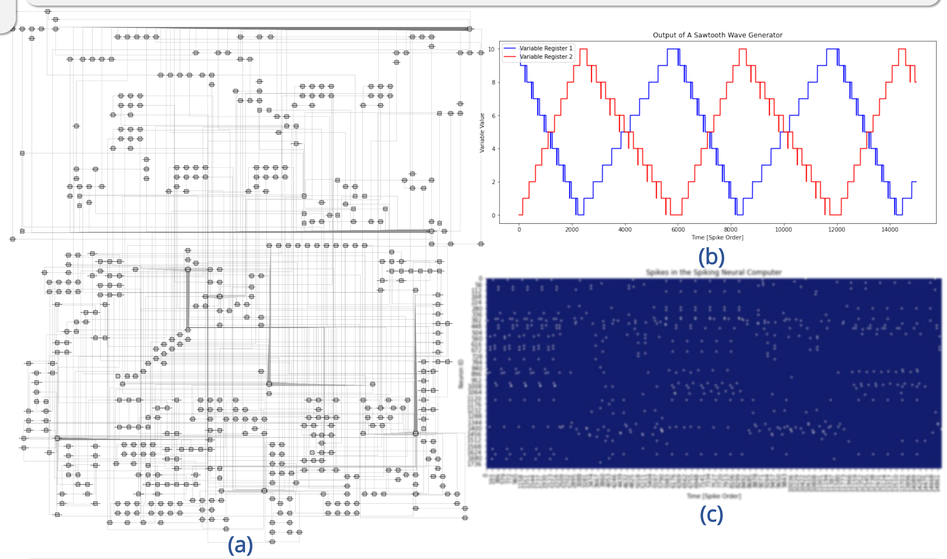
The instructions used are shown below:



Using integrate-and-fire (IAF) neurons and delay units, we can realize various instructions for implementing Turing machines and task parallelism.

Architecture of a Programmable Neural Computer

We design a Python library that, given a number of instructions and variables, creates a 4-instruction set spiking computer with that can be programmed and simulated on GPUs. The computers generated can be visualized as circuit diagrams of either instructions, or individual neurons. Our codebase creates the ALU and hardware required for the interpretation of branching instructions automatically.



(a) An example of an 4 instruction memory, 4 variable memory computer design. This computer is made of 272 instructions and 896 neurons, meaning it can be executed on hardware with <1024 neurons. **(b)** Output of a sawtooth wave generator using 6 lines of code executed with an 8 instruction machine. **(c)** We can visualize each spike as the computer runs.

References

[1] Hyötyniemi, Heikki. "Turing machines are recurrent neural networks." Proceedings of SteP'96 (1996).

We build programmable spiking computers, and code programs for them that can be efficiently executed. We provide a library to enable generation and execution of code in such neural computers.

mkt2126@columbia.edu
http://mehmetkeremturkcan.com