# Implementation of a method for identifying polychronous patterns from NN spiking data

# 1 Description

Spiking networks can serve as powerful supervised learning and memory representation tools with several potential applications in the machine learning domain [?].

Spiking network polychronization refers to reproducible time-locked but not synchronous firing patterns with millisecond precision. It was originally studied by Izhikevich [?] in a model of cortical spiking neurons with axonal conduction delays and spike-timing plasticity. The analysis of polychronous models is of interest because it is possible to use polychronous groups to encode information in different tasks [?, ?]. It is also possible to optimize a spiking network to maximize the number of polychronous groups [?, ?].

However, an obstacle to the efficient use of polychronous networks is that the original methods proposed to identify polychronous patterns from spiking data are highly inefficient. Different alternative methods have been proposed [?, ?, ?].

### 2 Objectives

The goal of the project is to implement an algorithm that, given the spiking output of Izhikevich's network, identifies the polychronous groups. Implementation of the network is NOT required. An existing implementation (in any language) could be used to generate the spiking data. The Python implementation is reduced to the analysis of the data. Obviously, to identify polychronous groups, it is important to understand what polychronous groups are and how they are generated. This is what this project is about.

As in other projects, a report should describe the characteristics of the design, implementation, and results. A Jupyter notebook should include calls to the implemented function that illustrate the way it works.

# 3 Suggestions

- There are available implementations of the model in C++ and Matlab https://www.izhikevich.org/publications/spnet.htm that illustrate the way polychronous network produces its output.
- Check different approaches to detect the groups [?, ?, ?]. Any of these approaches, or a new one, could be implemented.
- Implementations can use any Python library.
- Visualization of the identified patterns is encouraged as an additional, but not required, delivery of the project.

#### References

[1] Hélène Paugam-Moisy, Régis Martinez, and Samy Bengio. Delay learning and polychronization for reservoir computing. *Neurocomputing*, 71(7-9):1143–1158, 2008.

- [2] E. M. Izhikevich. Polychronization: Computation with spikes. *Neural Computation*, 18(2):245–282, 2006.
- [3] Joseph Chrol-Cannon, Yaochu Jin, and André Grüning. An efficient method for online detection of polychronous patterns in spiking neural network. *CoRR*, abs/1702.05939, 2017.
- [4] Runchun M Wang, Tara J Hamilton, Jonathan C Tapson, and André van Schaik. A mixed-signal implementation of a polychronous spiking neural network with delay adaptation. *Frontiers in neuroscience*, 8, 2014.
- [5] R. Santana, C. Bielza, and P. Larrañaga. Maximizing the number of polychronous groups in spiking networks. In *Companion Proceedings of the 2012 Genetic and Evolutionary Computation Conference GECCO-2012*, pages 1499–1500, Philadelphia, US, 2012. ACM Press.
- [6] R. Santana, C. Bielza, and P. Larrañaga. Changing conduction delays to maximize the number of polychronous groups with an estimation of distribution algorithm. Technical Report UPM-FI/DIA/2013-1, Department of Artificial Intelligence, Faculty of Informatics, Technical University of Madrid, 2013.
- [7] W.L. Maier and B.N. Miller. A minimal model for the study of polychronous groups. *Arxiv* preprint, arXiv:0806.1070, 2008.
- [8] R. Martinez and H. Paugam-Moisy. Algorithms for structural and dynamical polychronous groups detection. pages 75–84. Springer, 2009.