Report on Embedded and Distributed Somax for MicroController implementation

Joakim Borg, Jan 2022

Introduction

Somax is an interactive system which improvises around a musical material, aiming to provide a stylistically coherent improvisation while in real-time listening to and adapting to input from a musician. The system is trained on some musical material selected by the user, from which it constructs a corpus that will serve as a basis for the improvisation. The main idea is that Somax should serve as a co-creative agent in the improvisational process, where the system after some initial tuning is able to listen and adapt to the musician in a self-sufficient manner.

The Somax Front-end

The Somax system consists of two main components: the back-end server, which is implemented in Python, and the front-end which is implemented in the Max programming language [1]. The front-end has gone through a number of changes during the past two years, where the most important one is the wireless paradigm described in [4]. With this update, the system has gone from supporting a single co-creative agent to being able to support an entire ensemble of co-creative agents, each of them improvising over their own musical material, where the user may specify how the agents should listen to each other and/or to the musician (or musicians) interacting with the system. This is possible thanks to a fully parallelized implementation of the back-end server, where each of the parallel agents communicates with a single, centralized scheduler to ensure that the musical timing and tempo is maintained by all agents.

Embedded and Distributed Somax

In collaboration with HyVibe, a first step towards integrating the Somax system in the HyVibe Guitar has been taken. A number of optimizations of the Somax system has been made to facilitate the future integration of the system into the microcontroller used in the HyVibe Guitar (which does not have the same processing power as a modern laptop, which Somax normally runs on). An intermediate prototype has been constructed, where the back- end server of the Somax system is running on a separate microcontroller and communicating directly with the HyVibe Guitar as well as the Max front- end on a separate computer.

For this particular purpose, distributed versions of the server and front- end were implemented, which in turn allows multiple machines to communicate with a single server over a network. This means that several agents may exist on multiple machines and communicate with each other wirelessly and thereby allowing Somax to work as a fully distributed system. The original architecture, as described in chapter 4 of [5], was already designed as a local back-end server communicating with a local front-end client over a network-compatible protocol (OSC [2]), with the intention of being able to move to a fully distributed system at some point. With this in mind, no large architectural changes were needed to adapt Somax from a local system to a fully distributed one.

In terms of optimizations, the most expensive runtime operations in So- max revolve around the aspects of creating, shifting and scaling peaks (see chapter 3 and 4 of [6] for details). Here, a number of strategies to handle the peaks along a discretized time axis were implemented, in order to reduce the computational cost of the above mentioned operations. The source code for the distributed Somax architecture is available at [4].

References

[1] Cycling74 - Max. https://cycling74.com/products/max/. Accessed: 2022-01-31.

[2] Osc. https://opensoundcontrol.stanford.edu/. Accessed: 2022-02- 01.

[3] Somax - PyPI. https://pypi.org/project/somax/0.1.0/. Accessed: 2022-01-31.

[4] Somax2 at dev-hyvibe. https://github.com/DYCI2/Somax2/tree/ dev-hyvibe. Accessed: 2022-01-31.

[5] The Somax Software Architecture. Ircam-STMS, Technical Report, 2021.

[6] The Somax Theoretical Model. Ircam-STMS, Technical Report, 2021.

[7] Laurent Bonnasse-Gahot. An update on the SOMax project. Ircam- STMS, Tech. Rep, 2014.

[8] Joakim Borg. Somax 2: A Real-time Framework for Human-Machine Improvisation. Internal Report - Aalborg University Copenhagen, 2019.