STIRRING UP PEA STEW: A NETWORKED FEEDBACK STRUCTURE FOR LIVE CODING

Scott Wilson *Norah Lorway* *Tim Moyers* *Rosalyn Coull* *Visa Kuoppala*

Music Department
University of Birmingham
Edgbaston, Birmingham
B15 2TT
United Kingdom
beerensemble@contacts.bham.ac.uk

ABSTRACT

This short paper discusses Pea Stew, a musical work that consists of a wireless meshed audio network including an arbitrary number of laptop performers, and a structure for improvisation using live coding techniques. Pea Stew draws upon various precedents in 'feedback music', most notably Nicolas Collins' classic analogue work *Pea Soup*, but also Atau Tanaka and Kasper Toeplitz's Global String, David Tudor's *Rainforest* pieces, and Toshimaru Nakamura's 'No-input Mixing Board', amongst others. Using a design initially developed by Wilson, ongoing development of the piece has taken place during workshop sessions with the Birmingham Ensemble for Electroacoustic Research. Once seeded with noise, performers use live coding techniques to intervene in the signal chain. The result is collectively produced, and indeterminate: While every change has an effect, the system is too complicated to allow for the result to be predictable.

1. INTRODUCTION

In this paper we will discuss the musical work *Pea Stew*, its design, implementation, and aspects of its performance. The work was initially conceived and designed by Wilson, and then further refined and developed in workshops and performances with all the authors (collectively BEER, the Birmingham Ensemble for Electroacoustic Research). *Pea Stew* makes use of audio feedback, with an arbitrary number of performers sharing audio streams over a wireless network. Performers intervene in the signal chain using live coding techniques.

2. PRECEDENTS

There is of course a large body of 'feedback music'. Of particular relevance to *Pea Stew* are works such as David Tudor's *Rainforest IV* [1], which makes use of a series of objects which serve as resonators for electronic signals (in some versions with signals circulating in 'networks' of objects); Atau Tanaka and Kasper Toeplitz's *Global String* [2], which uses a wide-area network as part of resonant system involving a multi-site art installation; and Toshimaru Nakamura's 'No-input Mixing Board' approach [3], which uses feedback loops and processing to manipulate sound originally deriving from circuit noise, rather than an external source.

As is likely apparent from its name, however, *Pea Stew* is most immediately inspired by Nicolas Collins' *Pea Soup*. Originally an analogue work, *Pea Soup* consists in short of one or more feedback loops each made up of a microphone, a limiter, a phase shifter controlled by an amplitude follower, filtering, and a loudspeaker. This creates a 'site-specific "architectural raga"', in which feedback is controlled, and different patterns of pitches emerge, as a result of the phase shifter changing the resonances of the system [4].

3. IMPLEMENTATION

3.1. Normal or Body Text

Pea Stew is implemented in the SuperCollider language [5], and makes use of the *Republic* Quark [6] (an SC extension) for its basic network setup. Wilson has written a custom SC class to interface with JackTrip [7], an extension to the Jack audio system [8] that allows for users to share audio streams over a network. JackTrip requires individual pairs of client and server 'devices' for each connection, which makes configuration non-trivial. Each pair allows for one or more bidirectional audio streams. Called *RepublicJackTrip*, the custom class allows for the straightforward creation of a network of audio streams between an arbitrary number of users, automating the allocation and creation of JackTrip clients and servers. (Groups from sizes two to five have been tested with successful results.) *RepublicJackTrip* implements a fully meshed network topology (see Figure 1).

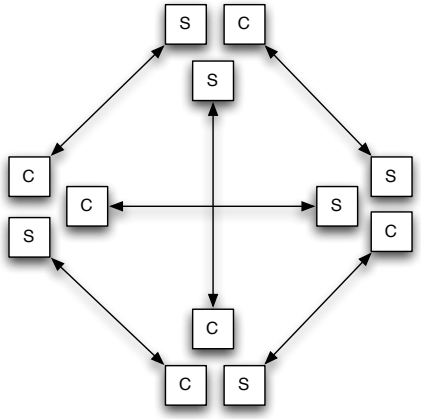

Figure 1. A RepublicJackTrip network topology for four players. An 'S' represents a JackTrip server 'device', a 'C' a client 'device'. Each cluster of (in this case) three devices, represents an individual node on the network, i.e. a performer's computer.

In the current configuration, each player runs a basic processing node that collects and scales the signals sent from other players. These are summed, and then fed into a delay. The output of this is then sent to an FFT-based process. This does the phase shifting a la *Pea Soup*, but on a bin-by-bin basis, with each bin's phase being shifted according to its magnitude. In testing this seemed to allow for more complicated sonic results than a single phase shift.[1] The output of this is then sent to a limiter.

At this point in the signal path performers intervene using live coding techniques to alter the sound. This is generally done using the Just in Time Library (JITLib) [5] included with SuperCollider. Processing can be serial or parallel, and in practice we have not limited the sorts of processing used.

The output from each player's processing is sent to other performers via the JackTrip connections, and played over one or more loudspeakers (see Figure 2).

[1] 'More complicated' refers here to the number of pitches in play (which may or may not be desirable) and the patterns of movement, and does not represent an aesthetic judgment.

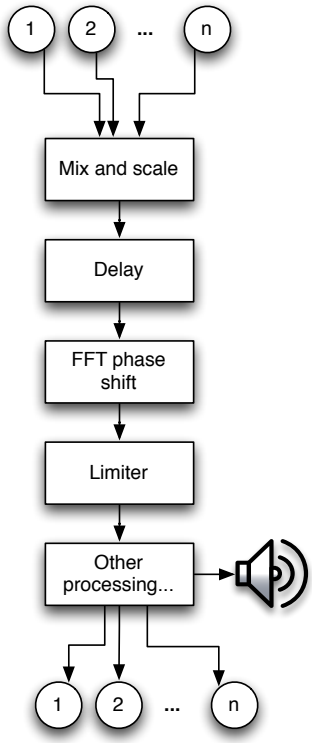

Figure 2. Signal path for each player's computer. The numbers at top and bottom represent JackTrip receives and sends from other players.

4. PLAYING *PEA STEW*

As an input-less digital audio network *Pea Stew* is theoretically noise-less, so a performance cannot rely on system noise as an initial source a la Nakamura, and the network must be primed with some signal. We have found low-level pink noise or impulse streams to be useful for this purpose. Once the signal is introduced, players turn up one or more of their inputs until the system starts resonating, at which point the noise source is generally removed. While it is relatively easy to set the gain at some node low enough to cause the resonance to stop, it is generally not necessary to re-prime the system in such cases, since (at least until enough time has lapsed that rounding errors reduce samples to a value of zero), the signal is not silent, merely currently inaudible, and increasing the gain will cause it to return. That said, in some cases we have found it interesting to inject a new signal at some point in a performance, as this may activate new modes of resonance.

Once seeded, a performance involves improvisation through live processing of the audio stream. In addition to live coding techniques, the code for the basic node provides a GUI interface that allows for performers to scale the input from each of the other players, weighting it to different degrees. This also allows for the effective reconfiguration of the network topology in real-time, since by muting the correct inputs any possible topology can be achieved. Since each connection between players

can be bi-directional or mono-directional, the number of possibilities is large, and with larger numbers of performers, two or more isolated sub-networks can be created. In performance we have found varying the network topology to be very useful. The GUI allows for negative gains, i.e. phase inversion of the audio as it passes that point in the network. The GUI also allows the performer to control a number of parameters, such as the lag time for the phase shift, and the input delay time for this node. By lengthening the latter, a player can change the resonance of the system, effectively lowering the fundamental frequency of all audio paths passing through that node. The value of the delay time is initially set randomly, in order to avoid equivalent resonances between pairs of nodes.

In terms of the processing involved performers have used a variety of techniques, including distortion and modulation, pitch shifting, granular techniques, and applying amplitude envelopes to create rhythmic effects. As one might expect, filtering is a very powerful tool in this context, and can radically change the output of the system, but we have found it to be something that is easily overused.

Playing *Pea Stew* presents challenges somewhat different to normal live-coding situations. The sound produced is truly collective, and while the output heard at each performer's loudspeaker(s) will be different (often quite surprisingly so), any individual change made is likely to have an effect on the entire network, or at least on any sub-networks that the performer is currently a part of. Even straightforward processing is rendered unpredictable (partly because most processing is recursive along some path), and the experience has been likened by one member to 'trying to push around a room-sized blob of jelly'. Each action a player takes has some effect, but it is often impossible to anticipate what it will be. While *Pea Stew* tends to force performers outside their comfort zone in terms of the predictable use of knowledge and skills, the experience of playing it is often delightful and surprising in its indeterminacy.

5. FUTURE WORK

Development of *Pea Stew* is ongoing, and has already involved many hours of workshop-ing and testing, as well as a number of performances. In the future we would like to experiment with a number of different aspects.

One of these is the imposition of musical form. BEER's work has focussed on structured improvisation, and in many cases this has involved strategies to create formal structures that are more complicated than improvisational flow, whether imposed or created on the fly. One possibility would be to move through a number of randomly or pre-determined topologies in a piece. Another might be to make use of slightly different basic processing for different sections. (We have tried a number of variations in the phase shifting algorithm, and they do exhibit different characters.)

Another possibility would be to allow processing that is not recursive, i.e. processing whose output is only to the loudspeakers, and is not sent to the other nodes of the network. Currently this is only possibly in cases where a performer is receiving from other performers, but where all of them have set the input from her to zero. While in some senses this would depart from what we feel is the collective spirit of the piece, this would allow performers greater control.

One final possibility to explore would be performances with large numbers of players. In practice we have found four or five to be more satisfactory (although different in character) to two or three performers, with a good balance between stability, complexity, unpredictability and the ability to influence. It seems likely that large ensembles would further diminish the influence of individual performers, but would provide more opportunities for interesting sub-networks. Network performance might be a concern of course, and it might be worth testing this on a wired network rather than our usual wireless one.

6. LISTENING

A sample performance of *Pea Stew* is available here: http://soundcloud.com/beer-ensemble

7. REFERENCES

[1] Driscoll, J. and M. Rogalsky. "David Tudor's "Rainforest": An Evolving Exploration of Resonance", *Leonardo Music Journal* Vol. 14 (2004): 25-30.

[2] Tanaka, A. and B. Bongers. "Global String: A Musical Instrument for Hybrid Space", *Proceedings of the conference cast01//Living in Mixed Realities,* Bonn, Germany, 2001.

[3] Meyer, W. "Toshimaru Nakamura: Sound Student", *Perfect Sound Forever* (2003). http://www.furious.com/perfect/toshimarunakamura.html

[4] Collins, N. *Pea Soup: A History*. http://www.nicolascollins.com/texts/peasouphistory.pdf

[5] http://supercollider.sourceforge.net

[6] Available at http://quarks.sourceforge.net/

[7] http://code.google.com/p/jacktrip/Jack

[8] http://jackaudio.org/