SPECTRALCONWAY: CELLULAR AUTOMATA OFF THE GRID

Dr Scott McLaughlin *Dr Pierre Alexandre Tremblay*

CeReNeM
University of Huddersfield, HD1 3DH, United Kingdom
s.mclaughlin@hud.ac.uk p.a.tremblay@hud.ac.uk

ABSTRACT

SpectralConway is a spectral implementation of cellular automata (CA), specifically, John Conway's Game of Life (GoL)[3]. The process eschews the traditional quantised grid paradigm of the CA in favour of a frequency continuum. We apply the neighbour rules to a set of floating-point values (partials), with the neighbourhood defined as a pitch interval. This paper outlines the reasons for taking this approach, the concept behind the algorithm, contextualises this approach within new developments in CA, and assesses the algorithm in the context of the piece *Whitewater*.

1. INTRODUCTION

The initial concept for SpectralConway came from writing the piece *Whitewater* (for wind instrument and live electronics, performed at ICMC 2007 [7][1]). The piece is based on slowly changing multiphonics, and we wanted a process that could allow the computer to grow and evolve sounds based on the live input: creating synthetic multiphonic-like timbres that the player can interact with. The reason for using GoL as the basis of the algorithm is almost completely aesthetic and subjective—beginning as the composer's fascination with the "lifelike" [16] motion of the graphical CA. Given an input, GoL is a simple system that will create variation that can exhibit both stability/repetition and novel variation, providing a level of interactive unpredictability within a globally stable sonic texture.

Also, because the piece relies on bounded improvisation by the player to guide the structure, a CA approach seemed most viable as it fosters local level variation while allowing emergent structures on a larger scale: as Eduardo Reck Miranda notes, "the inner structures of sounds seem more susceptible to cellular automata modelling than larger musical structures." [8] We have used a CA to emulate in the audio world of sound spectrum the lifelike inner motion observed on graphic CA. The algorithm receives analysis of multiphonic timbres in real-time and drives their temporal proliferation, with the larger structure being guided by live performer input.

Another aesthetic reason for using GoL—although many other CA would have been suitable here also—is that the rules of the process tend to generate cluster-based patterns. In the musical mapping this often leads to pitch clusters, with their resultant acoustic beating patterns. Such patterns are characteristic of the inharmonic multiphonics that make up the material in *Whitewater*.

All these aesthetic choices were based on the contemplation by the composer of a graphic-based GoL—not limited to GoL implementation—but we welcome other CA rule-systems to explore different properties. However, the first and foremost issue we were confronted with was the question of musical mapping, as we were unsatisfied with the obvious simple mappings.

1.1. Previous Musical implementations of Cellular automata

Our focus in this paper is not explicitly to extend CA theory, but to discuss our particular mapping approach from an artistic perspective. To give our work a context it is worth briefly examining the field of CA implementations in music: for a detailed explanation of CA technique, and a history of CA implementations in music up to 2005, please refer to the article by Burraston and Edmonds [1], or for a general survey of CA outside the arts see Ganguly *et al* 2003 [2].

We have observed that previous implementations of CA in music have fallen into two main paradigms: (a) note-to-cell mapping which focuses on localised pattern variation in midi/note-based music; (b) synthesis (mainly granular) implementations, focusing on stochastic pattern variation and swarm effects in timbre-based music, such as Eduardo Reck Miranda's *ChaosSynth* [8]. Since 2005 there have been efforts to improve the understanding of mapping CA processes onto musical parameters [5], as well as research into less linear mapping approaches, such as radial mapping [4], and histogram to spectrum mapping [12]. Toguchi *et al* use a wave propagation CA model to simulate impact sounds [14]: this type of CA has a different mode of proliferation to GoL and would not have suited our aesthetic considerations. These recent approaches are beginning to explore the space in-between linear and stochastic paradigm, but not to our satisfaction

[1] see http://eprints.hud.ac.uk/7396/ for the score and a recording of the piece, as well as for the compiled Max/MSP external.

for this purpose: we needed to be able to manipulate individual partials (frequency/amplitude pairs) that change over time in the potentially infinite domain of floating-point frequency values. The stochastic approach lacks sufficient control at the frequency level, while note-to-cell mapping strategies require us to quantise the frequency values to a limited set, more amenable to a fixed grid; this would reduce the possibilities for interesting timbres with partials whose frequencies lie outside that set. After some initial trials with mapping the graphic CA to the audio, we reassessed our approach.

2. SPECTRALCONWAY

2.1. Conceptual Description

What is novel about our approach to CA mapping is that it does not rely on a fixed grid of discrete values with a neighbourhood based on adjacency; instead, we use a floating-point continuum with a neighbourhood based on musical interval distance.

Let's compare the standard CA with SpectralConway. In a basic 2D graphic CA, the world is quantised as a grid of cells, each of which may be alive or dead. In successive generations, each cell is tested against a neighbourhood of the eight cells immediately surrounding it, as shown in Figure 1. Eric Weisstein defines the GoL rules as:

> 1. Death: if the count is less than two or greater than three, the current cell is switched off.
>
> 2. Survival: if (a) the count is exactly two, or (b) the count is exactly three and the current cell is on, the current cell is left unchanged.
>
> 3. Birth: if the current cell is off and the count is exactly three, the current cell is switched on [17].

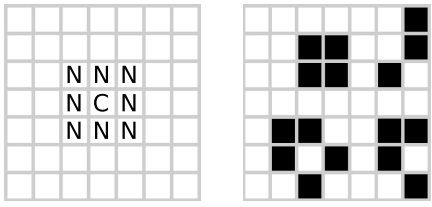

Figure 1. (Left) Graphic CA with neighbourhood based on adjacency: C = current cell; N = neighbour. (Right) Example of Graphic CA.

In SpectralConway, cells are replaced with spectral partial descriptors (frequency/amplitude pairs), and each partial is tested against a neighbourhood defined by a pitch interval defined in semitones, such as 12 for an octave or 7 for a fifth. SpectralConway tests to see how many other partials lie within this pitch interval and applies GoL-type rules: the next state (alive or dead) of the tested partial is determined by the amount of live neighbours. Note that the algorithm is concerned only with pitch: each partial's amplitude is passed through the algorithm unaltered.

Figure 2 shows spectralConway's intervallic neighbourhood, defined here as an interval of an octave and applied to each of the partials: the large dot is the partial currently being tested, and *a-d* are other partials alive in the current generation. Here, the partials *a* and *b* are within the neighbourhood of the tested partial, *c* and *d* are not. Applying the classic GoL rules described earlier, the target partial would survive to the next generation as there are exactly two other partials within its neighbourhood. Partials *a* and *b* would also survive, but partials *c* and *d* would die as they each have less than two partials in their respective neighbourhoods.

2.2. Technical Implementation

SpectralConway was originally written in Javascript for Max/MSP [11], and has since been hard-coded as a C external object for greater efficiency. Our object does not analyse the audio input, this is performed by other objects: it has been positively tested with fiddle~ [9] and sigmund~ [10] and FTM/Gabor gbr.peaks [13]. SpectralConway receives the resultant input as list of partials (frequency/amplitude pairs) and places them as cells in an array of fixed size (the "world"): 32 cells is the default but this can be changed at instantiation time. Our object also does not deal with audio synthesis, but again outputs a list of the living partials' pitch and amplitude to be re-synthesised. By reducing the processing to the control data only, we both reduce the CPU load for the object, and open up possibilities for enhanced reusability/adaptability of the idea.

The system's initial state is with all cells dead, and the list of incoming partials are systematically declared as live cells. The CA rules—the maximum and minimum living neighbour count required to stay alive—are preset to the standard GoL rules as described above, and the neighbourhood's defining pitch-interval is set to an octave (12) as default: these three parameters can be altered in real-time with Max messages sent to the instance.

A 'seeding' mechanism takes each incoming cell (as a list of frequency/amplitude pairs) and places it in the array, replacing the nearest dead cell; if there are no more dead cells then the array is full and the input is discarded. An 'evolve' message applies the CA rules to all the current cells. A 'play' message outputs the list of live cells as frequency/amplitude pairs for synthesis purposes.

The 'seeding', 'evolve' and 'play' mechanisms are all triggered by external messages, and are independent to allow for separate control: this means that the evolution of the CA is not tied to the rate of incoming partials. For example, in the *Whitewater* implementation, the rate of evolution is controlled by an external signal, so that even if there is no new seeding of the world, the evolution of the spectrum continues.

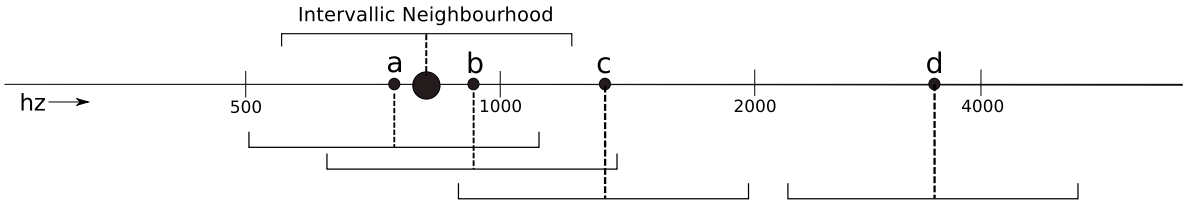

Figure 2. SpectralConway distance-based neighbourhood.

2.3. Some Consequences of Continuum-based Approach

In the example from section 2.1, SpectralConway appears to behave largely as a grid-based CA would, but there are consequences to using this continuum-based approach that, while not relevant to our implementation, are worth briefly examining. As SpectralConway uses partials represented as floating-point frequency values, there is the possibility of a very large number of partials being present in a given interval-defined neighbourhood, as opposed to a maximum of eight possible adjacent neighbours on a square grid. In our implementation (see section 2.2), we circumvent this potential problem by using an array of fixed size, thus limiting the amount of possible neighbours. This effectively creates a grid of dynamically changing scale without needing to quantise frequency values.

Also, for our purposes, the possibility of large numbers of neighbours is not relevant because the GoL rules are only concerned with neighbourhoods of a finite range of "cells"; numbers outside this range mean death regardless of size. But it has considerable scope for development in contexts with higher numbers of partials, and allows for the development of rule sets specific to SpectralConway,

This continuum-based approach we have adopted means that SpectralConway is not quite a Cellular Automata (it has no cells), but our implementation behaves like a CA because of the way it has been implemented with a fixed array size, and by being based on population-density rules. However, on the level of analogy, SpectralConway also has some of the characteristics of a Continuous Spatial Automata (CSA), described by MacLennan as, "analogous to a cellular automaton, except that the cells form a continuum, as do the possible states of the cells." [6]

SpectralConway's shares with the CSA the possibility of defining the neighbourhood as a radius, but does not share the continuous state aspect as the partials in SpectralConway have binary states (alive or dead). Also, as far as we are aware, there are no musical implementations of CSA.

We believe that SpectralConway is a unique approach to the implementation of CA in music and opens up considerable possibilities for creative development.

3. ASSESSMENT

What fascinated us originally about the graphical CA was how its patterns were created and evolved: that there was a lifelike quality to their spontaneous variations and transformations. CA evolution tends towards three possible states: 'extinction', where all the cells are dead; 'chaos', where cell states change constantly and there are no perceivable patterns; 'life', where there are varying levels of activity and perceivable patterns. SpectralConway achieves an audio analogue of this in that the spectral content of the live input undergoes spontaneous and unpredictable variation, as well as crossbreeding with previous live input material that is still in the world.

For *Whitewater*, we implemented SpectralConway to fit with a specific musical language: for example, the piece takes advantage of the Game of Life rules' tendency towards settling into patterns of stable periodic oscillators ('blinkers'). The CA process acts like a filter on the incoming multiphonic so that some frequencies die off and others persist, often becoming locked into ostinati. The relationship between the graphical CA, which was the initial inspiration for the piece, and resultant sound is quite clear: the patterns and blinkers visible on the graphical CA become patterns of oscillating pitches in SpectralConway. The instrumental player has the choice of interacting with the resultant pitches or re-seeding the world with a new multiphonic.

SpectralConway, and CA in general, have some musical limitations. The piece *Whitewater* addresses the problem that the initial list of partials will be the only cells available in the world, therefore allowing very limited possibilities for evolution. The *Whitewater* patch addresses this in two ways[2]:

(1) By performing three analyses of the live input per attack, staggered at short intervals (each attack from the live player sends three time-staggered sets of inputs into SpectralConway) the world is 'super-seeded'. This allows

[2] The following ideas are not part of the SpectralConway algorithm, but help to enhance its perception in the piece.

for subtle variation in the frequencies, which alters the inner motion of the sound.

(2) To increase the available frequency-space for mutation and cell redefinition that would enrich the *Whitewater* world, we have introduced another form of 'super-seeding' wherein frequencies not present in the initial sound are added to the seed list after a delay. These frequencies are the combination tones between the most prominent recent frequency and the frequencies already in the seed list. These extra frequencies increase the potential richness of the sound and the possibilities for mutation; generation of harmony not present in the input sound.

4. FUTURE DEVELOPMENTS

The next step is to release SpectralConway publicly with the present paper. At present, SpectralConway is used in very different ways in the authors' pieces *Whitewater* and *Sandbox#2* [15] but we welcome other user's perspective on this approach.

We would also like to consider other approaches to mapping the frequency continuum, such as using spectral identity: each partial would be tested against integer multiples/divisors, with wider neighbourhoods going further up and down the spectra. Each different mapping has different consequences for the musical outcomes.

Finally, we would like to bring back the idea of discrete, floating cell boundary definition in CA back to the visual world, in a swarm-like agent interaction.

We hope that with this paper and the release of the Max object, other people will take this rich spectral implementation of Conway's Game of Life and find their own creative application for it.

5. REFERENCES

[1] Burraston, D. Edmonds, E. "Cellular Automata in Generative Electronic Music and Sonic Art: A Historical and Technical Review", *Digital Creativity* Vol. 16, No. 3, 2005, pp. 165-185.

[2] Ganguly, N Sikdar, B. Deutsch, A. Canright, G. Chaudhuri, P. "A survey on cellular automata", *Technical report*, Centre for High Performance Computing, Dresden University of Technology, 2003.

[3] Gardner, Martin "Mathematical Games: The fantastic combinations of John Conway's new solitaire game "Life"", *Scientific American* 223 (October 1970), pp. 120–123

[4] Kirke, A. and Miranda, E. R. "Capturing the aesthetic: Radial mappings for cellular automata music" *Journal of the ITC Sangeet Research Academy*, 21, 2007, pp. 15-23.

[5] Kirke, A. and Miranda, E. R. 2007b. Evaluating mappings for cellular automata music. In Proceedings of ECAL Workshop on Music and Artificial Life (MusicAL, Lisbon, Portugal).

[6] MacLennan., B. J. "Continuous spatial automata" *Technical Report CS-90-121*, University of Tennessee, Dept. of Computer Science, Knoxville, 1990. http://www.cs.utk.edu/mclennan/eldcompbiblio.html, accessed 04/01/2010.

[7] Mc Laughlin, S. "Whitewater", www.lutins.co.uk/whitewater, accessed 07/01/2010.

[8] Miranda, E. R. "Evolving Cellular Automata Music: From Sound Synthesis to Composition", *Proceedings of the Workshop on Artificial Life Models for Musical Applications - ECAL 2001*, Prague, Czech Republic 2001.

[9] Puckette M., Apel T., Lippe C., *fiddle~*, Center for Research in Computing and the Arts, UCSD, 2007.

[10] Puckette M., Apel T., Lippe C., *sigmund~*, Center for Research in Computing and the Arts, UCSD, 2009.

[11] Puckette M., Zicarelli D., *et al*, Max/MSP, www.cycling74.com

[12] Serquera, J. Miranda, E. R. "Spectral Synthesis and Control with Cellular Automata" *Proceedings of the International Computer Music Conference*, Belfast, NI, 2008.

[13] Schnell N., Schwarz D., *et al*, "FTM & Co: Gabor", http://ftm.ircam.fr/index.php/Gabor, 2005.

[14] Toguchi, S., Akamine, Y., Satoshi, E., "Research into the Generation of Sound Effects Using a Cellular Automaton" *Lecture Notes in Computer Science: Cellular Automata*, Volume 5191/2009, Springer, 2008.

[15] Tremblay, P. A. http://www.pierrealexandretremblay.com/welcome.html, accessed 06/01/2010.

[16] Waldrop, M. M. *Complexity: The Emerging Science at the Edge of Order and Chaos*, New York: Simon and Schuster, 1992, pp. 234.

[17] Weisstein, E. W. "Life." MathWorld-A Wolfram Web Resource. http://mathworld.wolfram.com/Life.html, accessed 04/01/2010.