ca : A System for Granular Processing of sound using Cellular Automata.

Shyamsundar Vaidhyanathan

Cellular Infrastructure Group
Motorola Inc.

IL-27, Mailstop 2-G5, 1501, West Shure Drive, Arlington Heights IL 60004
vaidhy@cig.mot.com

Dr. Ali Minai

Complex Adaptive Systems Laboratory

Department of E.C.E.C.S.
University of Cincinnati,
Cincinnati OH 45221-0030
Ali.Minai@uc.edu

Prof. Mara Helmuth

CCM [2] – Center for Computer Music, College Conservatory of Music
University of Cincinnati
PO Box 210095
Cincinnati, OH 45221
Mara@meowing.ccm.uc.edu

ABSTRACT

ca is a tool for the granular processing of sound using cellular automata developed on the SGI-Indy platform. It investigates the effects of change in the timbre of sound using a cellular automaton in real-time. The cellular automaton generated by the chosen rule controls parameters of a bank of filters. The system uses standard infinite impulse response filters and a general model of three neighborhood cellular automata. The composer[1] can configure the filter banks by adjusting bandwidths and center frequencies through the graphical interface. CA is very well suited as a tool for computer music composition because it is capable of creating a new palette of sounds for the composer and it is easy to use.

1. INTRODUCTION

A cellular automaton (CA) is a system defined on a discrete lattice where each lattice position ("cell") can take one of ***k*** values. In **ca** we use only binary states, so each cell has either a **1** or **0** value. The values of these cells change synchronously depending on a relationship with their neighboring cells. The value of a particular site depends on previous values of a neighborhood of sites around it. In more formal notation, if we have a one dimension automaton denoted by **a** then $a_i^{(T)}$ is the value of site **i** at time step **T**. Each site has a specific integer value, in our case either **0** or **1**. The evolution of the automaton depends on a neighborhood defined by a range **r**, the value of a given site depends on the last values of a neighborhood of at most **2r+1** sites.

$$a_i^{(T)} = F[a_{i-r}^{(T-1)}, a_{i-r+1}^{(T-1)}, \ldots, a_{i+r}^{(T-1)}] \quad (1)$$

where **F** is an arbitrary function which specifies the cellular automaton rule. Further reading in [1].

ca makes use of cellular automata to aid in composition of granular music. The cellular automata are used as a transformation mechanism for sampled sounds. The harmonic structure of the sampled sound is changed through this CA-based transformation mechanism.

1.1. Classification of Cellular Automata

Cellular automata can be considered to be a discrete dynamical system. In most cases the cellular automaton's evolution is irreversible. Trajectories in the configuration space for cellular automata therefore merge with time and after many time steps, trajectories which began from some possibly random initial states become concentrated onto *attractors*. The nature of the attractor determines the form and extent of such structures. They fall into one of the following qualitative classes.

- *A homogenous state.*
- *A set of separated simple stable or periodic structures.*
- *A chaotic pattern .*
- *A complex localized structure, sometimes long-lived.*

For further reading on this classification please refer [2].

1.2. Cellular Automata used in **ca**

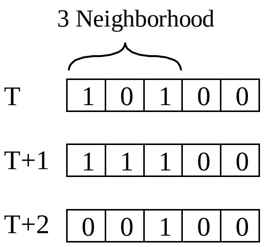
The cellular automata used in **ca** are one dimensional with a length of 32 and a neighborhood range of r=1.
Thus our rule is stated simply as

$$a_i^{(T)} = F[a_{i-1}^{(T-1)}, a_i^{(T-1)}, a_{i+1}^{(T-1)}] \quad (2)$$

where i =1 …32.

For the last cell on either end since they have just one neighbor the other neighboring cell is forced to = 0. Since each of the sites can be either a 0 or a 1 and r=1, we have 8 different neighborhood states which map to 0 or 1 at the next time step. Thus we can have a total of 2^8 = 256 different rules. Any of these 256 different rules can be used to evolve the cellular automaton.

[1] CA is used by computer music composers at CCM[2]



Rule	Outcome
000	0
001	1
010	1
011	0
100	0
101	1
110	1
111	0

Rule number 102 [01100110] in outcome

Figure 1: *Growth of a 5 cell CA with* ***r =1*** *and* ***rule =102***

Fig. 1 shows typical 5 cell automata with a 3 neighborhood. Rule number chosen in this case = 102. The rule number 102 is applied is as follows. The number 102 represented in binary is **[0 1 1 0 0 1 1 0].** So the 8 possible states shown in the rule table have these 8 bits as their respective outcomes. We see three time stages of growth for this automaton.

2. GRANULAR SYNTHESIS AND PROCESSING

The *Granular Synthesis* of sound involves the production of thousands of short acoustic particles in order to form larger sound events [3]. References [4-9] have detailed work on some proposed granular synthesis systems. A *Grain* is a signal with a specific amplitude envelope. The duration of a grain usually falls in the range of 1-50 milliseconds. *Grain Density* is number of grains occurring within a given time interval. The grains themselves are either just pure tones or generated through the implementation of independently applied non-linear functions. Once the fundamental grain is synthesized they are combined in several possible methods. Many granular synthesis systems have used stochastic methods to control the production of sound grains and the distributions and density of these grains [10-12]. *Granular Processing* is processing of sound by decomposing the given sound sample into small grains of sound, transforming these grains of sound and then recombining the grains to reconstruct a sound sample from it. E.g. In time scale modification, the original sample is granulated into a long series of grains and these grains are then simply recombined with different spacing between them [13,14].

2.1. Granular Processing /Synthesis in ca

In **ca**, cellular automata are combined with granular processing. The sound sample is granulated and each grain is passed through a bank of filters. These grains are then recombined synchronously. The main focus of **ca** is in transforming the harmonic structure of the grain. It deals with sampled sounds only and not synthesis of sound grains [15].

3. THE SYSTEM MODEL

In this system there are two different representations of time.

- **t** : This represents continuous time - in this case as it is all digital data it refers to time period for one iteration. E.g. In playing all the samples we go through **t=0** to **t=N** when we have **N** samples.

- **T**: This represents the time intervals when the CA grows. The CA grows to its next state only every T periods of time. E.g. At **T=0** the CA transforms from initial state to its first state and then at **T=1** it changes to its next subsequent state.

The Cellular Automata used in **ca** Ver 5.0 has 32 cells and a neighborhood of 3 represented by Eq. (2). Fig. 2 is a representation of the 32 cell CA system model. The input to **ca** is an audio file chosen by the composer. The input audio file is represented by an audio data buffer **D.**
N= length(D) represents the number of samples in the file.

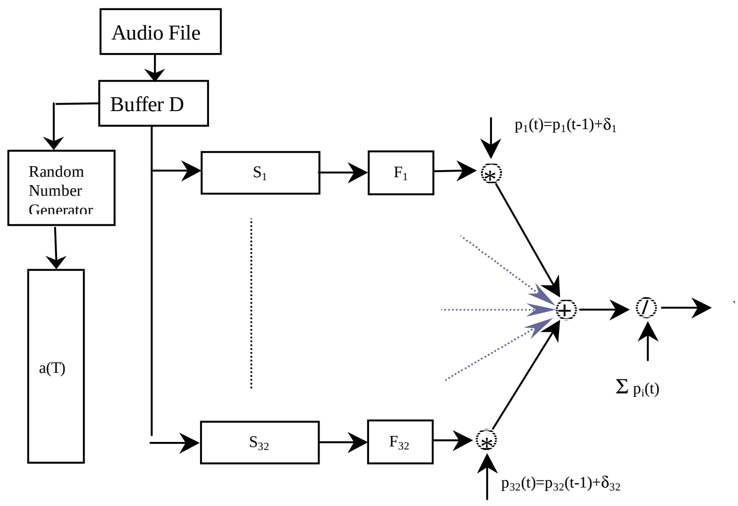


Figure 2: *Block Diagram for the System Model*

There are 32 cells in the automaton and each of these cells control a harmonic component of the audio data. To do this, we make copies of Buffer **D.**

$$S_i = D \quad \textbf{(3)}$$

where i = 1…32. These $\mathbf{S_i}$ are filtered through the filters $\mathbf{F_i}$. The filters $\mathbf{F_i}$ are 2nd order IIR (Infinite Impulse Response) filters whose transfer function is given by

$$H(z)=\frac{(\frac{G_o+G\beta}{1+\beta})-2(\frac{G_o\cos(\omega_o)}{1+\beta})z^{-1}+(\frac{G_o-G\beta}{1+\beta})z^{-2}}{1-2(\frac{\cos(\omega_o)}{1+\beta})z^{-1}+(\frac{(1-\beta)}{(1+\beta})z^{-2}}$$

where

$$\beta=\sqrt{\frac{G_B{}^2-G_o{}^2}{G^2-G_B{}^2}}\tan(\frac{\Delta\Omega}{2}) \qquad \textbf{(4)}$$

$\mathbf{G_o}$ -D.C.Gain, $\boldsymbol{\omega_o}$- Center Frequency, **G** – Gain at Center frequency. For more detailed discussion on the design of the filters used in **ca** refer. [16].

The CA needs to have a randomly generated initial value to start its growth. The first data element of the data buffer **D** is used as a seeding value for this random number generation. In **ca** we have chosen to use a one dimensional CA which depend on a 3 neighborhood. Thus every cell depends on its adjacent two cells. We do not allow fold over at the boundaries and so for the first cell and the last cell we have one of its neighbors always fixed at **0**. Since we have a 3 neighborhood cell, it can take one of the 8 different states. At a given time **T** any triplet $\mathbf{a_{i-1}^{(T)}}$, $\mathbf{a_i^{(T)}}$, $\mathbf{a_{i+1}^{(T)}}$ represent the 3 neighborhood for the cell **a[i]**. This triplet is compared with all of the possible outcomes. An outcome for each one of these 8 states as defined by the rule is assigned to the cell at the next period of time $\mathbf{a_{i+1}^{(T+1)}}$. In **ca** the composer chooses the rule. The CA grows once in every ***w*** samples where

$$T=w*t \qquad \textbf{(5)}$$

With $\mathbf{a_i^{(T)}}$ and $\mathbf{a_i^{(T+1)}}$ an increment buffer $\boldsymbol{\delta_i}$ is generated based on the rule shown in the table below.

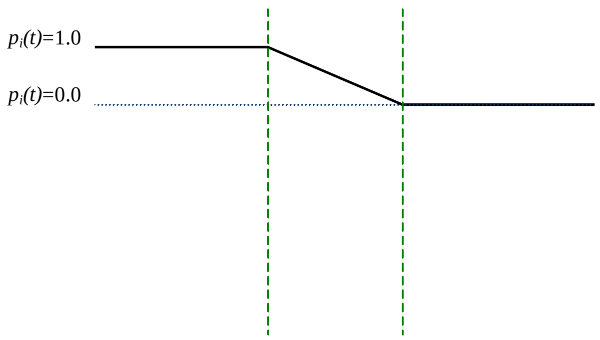
$a_i^{(T)}$	$a_i^{(T+1)}$	δ_i
0	0	0
0	1	+ε
1	0	-ε
1	1	0

Where
$$\varepsilon=\frac{1.0-0.0}{(w/4)} \qquad \textbf{(6)}$$

$\boldsymbol{\delta_i}$ is added to the gain multiplier $\mathbf{p_i}$ in every iteration.

$$p_i(t)=p_i(t-1)+\delta_i \qquad \textbf{(7)}$$

Fig. **3** shows plot of the gain multipliers defined in Eq. (7).



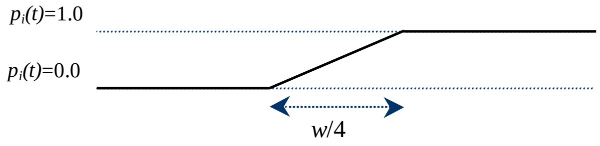


Figure 3: *Plot of the gain multiplier $p_i(t)$*

$$V(t)=\frac{\sum_{i=1}^{12}p_i(t)S'_i(t)}{\sum_{i=1}^{12}p_i(t)} \qquad \textbf{(8)}$$

V(t) - Final normalized output, $\mathbf{S'_i}$ - Output from the filter $\mathbf{F_i}$
This **V(t)** is sent to other optional modules (described in section 4) before it is played back through the audio hardware.

4. IMPLEMENTATION AND FUTURE WORK

ca has four modules. The first module is a *control module* where the composer chooses the CA Rule and the window size and the audio parameters for the file and filters. The second is a *plot module* and merely shows the waveforms of the audio files and the CA. The third module is an *analysis module* where the composer can get a 3D plot of the FFT analysis of the audio clips. The fourth module, a *pan module*, gives a choice between 2-channel and 4-channel audio modes for panning the sound clip. In the 4-channel audio, the 4-channel capabilities of the SGI hardware are exploited. Fig. 4 shows the *plot module* showing the plot of the audio clip and the CA in use.

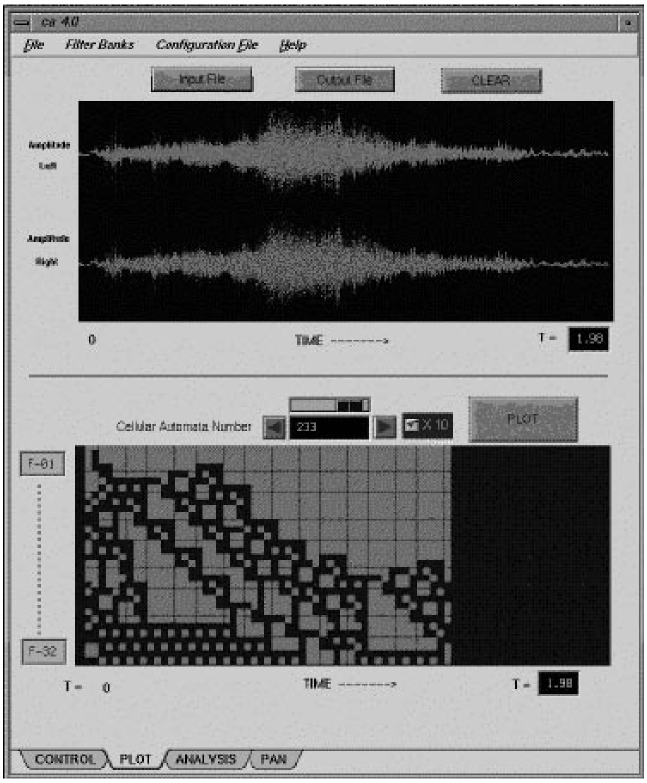


Figure 4: *Plot Module shows plot of the audio clip and the growth of the CA*

ca can be downloaded at no cost from the URL for **CCM^2** (Center for Computer Music, College Conservatory of Music) http://meowing.ccm.uc.edu/softmus.htm

Future work in this area could include -

- A DSP based implementation of the same system, which could probably handle real-time audio streams.
- A Web based application, that can handle web-audio streams and have several people interacting to contribute to the creation of new sounds.

5. REFERENCES

[1] Codd, E.F., Cellular Automata, New York. Academic Press 1968, pp. 122

[2] Wolfram, S., Universality and Complexity in Cellular Automata, Physics, Vol. 10, pp. 1-35, 1984

[3] Gabor, D., Acoustic Quanta and the Theory of Hearing, Nature, Vol 159, No. 1044, pp. 591-594

[4] Xenakis, I., Formalized Music: Thought and mathematics in composition, Bloomington, Indiana University Press, 1971, 273 pages

[5] Truax, B., Real-time Granular Synthesis with a Digital Signal Processor, Computer Music Journal, Vol. 12, No. 2, summer 1988

[6] Truax, B., Real-time Granular Synthesis with the DMX-1000 Digital Signal Processor, Proceedings of the International Computer Music Conference, 1986, San Francisco

[7] Truax, B., Discovering Inner Complexity : Time Shifting and Transposition with a Real-time Granulation Technique, Computer Music Journal, Vol. 18, No. 2, summer 1994

[8] Roads, C., Introduction to Granular Synthesis, Computer Music Journal, Vol. 12, No. 2, summer 1988

[9] Roads, C. and Strawn, J., Granular Synthesis of Sound, Foundations of Computer Music, Cambridge, Massachusetts, MIT Press, pp. 145-159

[10] Helmuth, M., Timbral Composition with Granular Synthesis, http://meowing.ccm.uc.edu/~mara/papers/

[11] Hunt, A., Kirk, R. and Orton, R., Graphical control of Granular Synthesis using Cellular Automata, Proceedings of the International Computer Music Conference, 1991, pp. 416

[12] Jones, D. and Parks, T., Generation and Combination of Grains for Music Synthesis, Computer Music Journal, Vol. 12, No. 2, summer 1988

[13] Lippe, C., A Musical Application of Real-time Granular Sampling Using the IRCAM Signal Processing Workstation, Proceedings of the International Computer Music Conference, 1993

[14] Hamman, M., Mapping Granular Complex Systems Using Granular Synthesis, Proceedings of the International Computer Music Conference, 1993

[15] Hunt, A., Kirk, R. and Orton, R., Musical Applications of a Cellular Automata Workstation, Proceedings of the International Computer Music Conference , 1991, pp. 165

[16] Orfanidis, S.J., Digital Parametric Equalizer Design with Prescribed Nyquist-Frequency Gain, Journal of Audio Engineering Society, Vol. 45, No. 6, 1997, pp. 444-455