# Composition at the Edge of Chaos

**Dave Burraston**
Creativity and Cognition Studios (CCS)
Faculty of Information Technology
University of Technology, Sydney Australia
dave@noyzelab.com

## Abstract

*Cellular Automata (CA) produce a diversity of behaviours and are an important tool for generative music composition. The aim of this paper is to establish, describe and evaluate a general CA composition process as a foundation for future work. Complex emergent behaviour is used to produce a composition, but the techniques described are universally applicable to other CA behaviour. Results are evaluated by recognised criteria Reflections on the work will identify the discrete stages of a generative composition process with CA.*

## Introduction

Cellular Automata (CA) has a diverse history beginning with parallel computation and becoming a foundation in the field of Artificial Life. CA are capable of a wide variety of behaviours and represent an important generative tool for the artist. Potential opportunities in the wider field of evolutionary techniques is presented in (Brown 2002). A review of CA in the MIDI domain was presented in (Burraston, Edmonds, Livingstone & Miranda 2004) which presented most of the research to date.

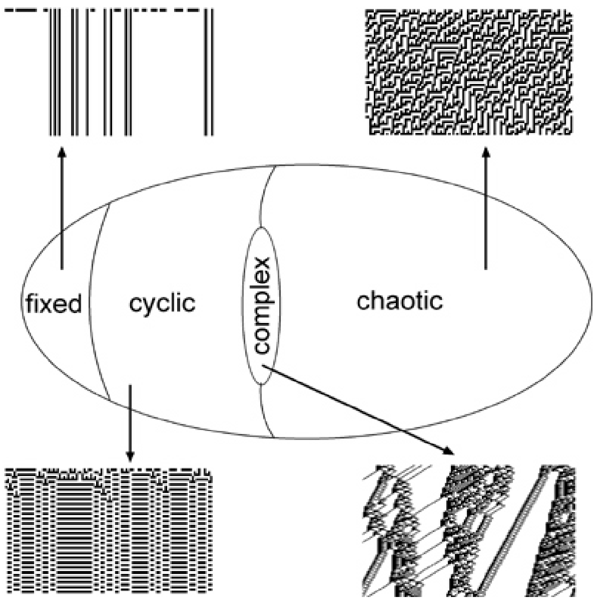


*Figure 1. Langhton's schematic of CA rule space and example spacetime plots of their behaviour*

The majority of work so far has been conducted in the domain of note sequencing, whereas MIDI loudness dynamics and other parameters have received less attention to date. It also clear that in most cases the focus has been building and describing the application software, and not the composition process or technical evaluation. The mappings used for the composition described in this paper are novel in that they are based on the global state of the CA, in contrast to mapping cells to events.

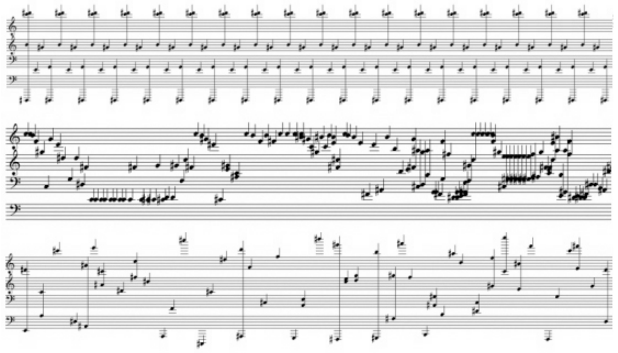


*Figure 2. Musical visualisations of ordered (Class 2), complex and chaotic behaviour.*

In this paper the composition process of using complex emergent behaviour is presented in the context of reflective practice (Schon 2003), enabling further exploration by the author as well as the reader. This description can also be read in the context of Doornbusch's 10th question on producing a concrete example of mapping in algorithmic composition practice (Doornbusch 2002). The overall goal is to identify the discrete stages of the composition process and evaluate the technical methods employed. Reflections on the work will identify the discrete stages of the generative composition process with CA. A further goal of this experiment is to allow for a shift of context away from the authors previous work in Max, and the problems already identified with that approach. Examples of these MIDI experiments, based on ordered and chaotic behaviour, is presented in (Burraston 2005a) (Burraston 2005b) (Burraston & Edmonds 2004).

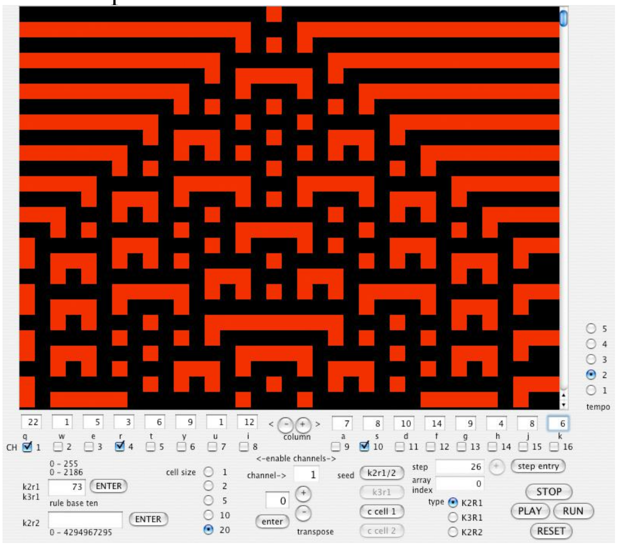
Complex systems such as CA produce global behaviour based on the interactions of simple units. CA have been classed by Stephen Wolfram with one of four behaviours (Wolfram 1984). Class 1 evolve to fixed points, Class 2 become cyclic, Class 3 evolve randomly and Class 4 produce complex forms. It known that it is formally undecidable to assign a CA to a behaviour class (McIntosh 1990).The relatively rare complex behaviour of class 4 was later shown to occur between ordered (class 1 and 2) and chaotic (class 3), termed the "edge of chaos" (Langton 1990). Langton produced an example schematic illustrating this result and this is shown in Figure 1, also including spacetime plots of example 1D CA evolutions for each class. The evolutions show space (cells) in the horizontal and time runs vertically downwards. An example visualization of CA behaviour converted to music is shown in Figure 2. This paper will focus on the musical mapping of complex behaviour, but the composition techniques described are universally applicable.

## Two Early CA Music Systems

CA have been utilised in a number of novel applications in the MIDI domain. Predominantly these applications have been in the area of MIDI sequencing, using a multiplicity of CA types and mapping strategies. Examples exist in the fields of overall structural composition, MIDI sequencing and sound synthesis/modification techniques. A more in depth review of CA in the MIDI domain is presented in (Burraston, Edmonds, Livingstone & Miranda, 2004). Two early MIDI experimenters of CA were Peter Beyls and Dale Millen, working independently in academia on slightly different systems.

Beyls early work (Beyls 1980, 1989, 1990), one of the first examples of a CA music system, is interesting both technically and also from a music industry standpoint. It was commercially sponsored by Atari and Yamaha for a brief period (Beyls 89). The types of CA investigated were varied and novel, drawing from a mixture of 1 and 2 Dimensions. A further interesting avenue was the use of time dependent rules, where the rule itself changes during the CA evolution. Beyls wanted a flexible MIDI mapping process for real-time composition and performance, after his earlier work in non real-time. This mapping drew from the CA history evaluation, a user defined root and the current cell value. Selection of MIDI channel was by cell index number, using the modulus function to map to the available number of MIDI channels. Durational values were computed by matching the CA cell value with a condition in a large decision tree. Beyls has extended his work to include selections based on Genetic Algorithms with his CA Explorer program (Beyls 2003). The CA are viewed as genotypes and are subject to mutation and crossover operations, the latest incarnation consists of two CA with a voting rule (Beyls 2004).

Cellular Automata Music (CAM) was created by Dale Millen at the University of Arkansas. Music is created from 1D binary, 2D Life and 3D Life CA by mapping the results to pitch and duration values (Millen 1990). The 1D and 2D lattice sizes are scalable up to 100. The 1D binary rule is completely definable as a 32 bit pattern by on/off switches, thus allowing any of the 2^32 possible rules to be selected.



22 1 5 3 6 9 1 12 | 7 8 10 14 9 4 8 6

q w e r t y u i | a s d f g h j k

CH [✓ 1] 2 3 [✓ 4] 5 6 7 8 | 9 [✓ 10] 11 12 13 14 15 16

k2r1 / k3r1: 73 (0 - 255, 0 - 2186) rule base ten

k2r2: (0 - 4294967295)

cell size: 1 2 5 10 [✓ 20]

channel-> 1

0

seed

step 26

array index 0

type [✓ K2R1] K3R1 K2R2

tempo: 5 4 3 [✓ 2] 1

Fig 3. *Dale Millen's Cellular Automata Music program*

Pitches are entered as a set of values and durations are applied by the program automatically, except in the 3D case where the calculated duration is also scalable. The seeding of the CA can be achieved by a number of methods, in the case of 2D Life this can be entered graphically with the mouse or by automatic seeding of Life forms. Millen later investigated the generation of formal musical structures with CAM, using a cyclic 1D CA rule (Millen 1992). This involved the arbitrary mapping of musically related pitch values to the CA rule. CAM runs on the Macintosh and a new version for Macintosh OSX (Millen 2004), shown in Figure 3, is available for free download (Millen 2005).

## CA and Complex Behaviour

CA are dynamic systems in which time and space are discrete. They may have a number of dimensions, single linear arrays or two dimensional arrays of cells being the most common forms. The CA algorithm is a parallel process operating on this array of cells. Each cell can have one of a number of possible states. The simultaneous change of state of each cell is specified by a local transition rule. This local transition rule is applied to a specified neighbourhood around each cell.

Global dynamics has been used in CA research for over a decade and provides a different perspective on their behaviour, based on the topology of attractor basins, rule symmetry categories and rule clustering (Wuensche and Lesser 1992). Wuensche's Discrete Dynamics Lab (DDLab) software allows for the exploration of global dynamics with CA and other discrete dynamic networks (Wuensche 2005).

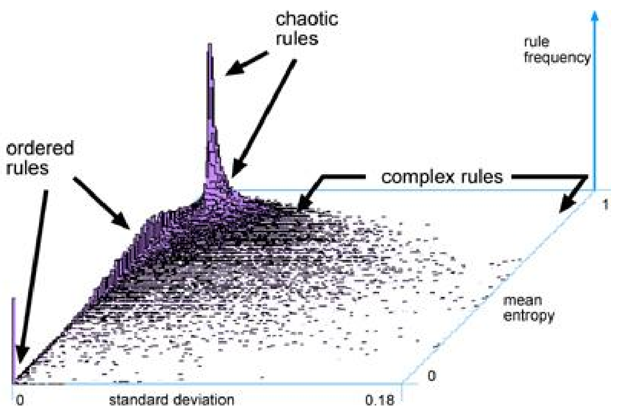


*Figure 4. Example of Wuensche's automatic CA rule space classification using a random sampling of 5 neighbour 1D binary CA (v2k5).*

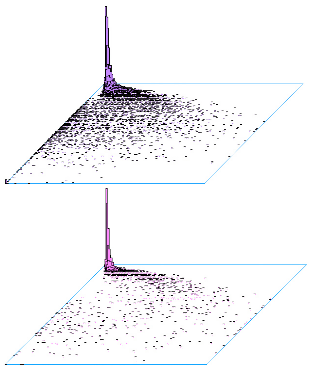


*Figure 5. Example of increase in chaotic rules demonstrated by Wuensche's automatic CA rule space classification using a random sampling of v2k6 (left) and v2k7 (right).*

In DDLab the number of state values is represented by **v** and the number of cells in the neighbourhood by **k**. This differs slightly from Wolfram's numbering scheme of **k** as the number of states and **r** as the neighbourhood radius. Wuensche's numbering scheme allows for even numbers of cells within a neighbourhood.

DDLab can automatically classify rule space from random rule samples based on mean entropy (a measure of amount of disorder) against standard deviation of entropy. Each type of behaviour has its own "signature". Ordered rules will tend to decrease entropy and standard deviation, chaotic rules increase entropy with a low standard deviation, and complex rules tend to have higher standard deviations with entropy values away from the extremes. An example of this method can be seen in Figure 4, where a random sample of over 10,000 rules was measured. The rules here are the 5 neighbourhood 1D binary rules, v2k5 in Wuensche's terminology or k2r2 in Wolfram's. Wuensche has importantly shown with this method that as the neighbourhood size is increased, the proportion of chaotic rules rises very sharply. This can be seen clearly in Figure 5, where similar plots are shown for 6 and 7 neighbour rules showing a chaotic tower as the main feature in the top left of the plots. The implication of this is that if a rule is generated at random it is highly likely to be a chaotic rule, which has implications for the serendipitous use of these systems in artistic endeavour.

The magnitude of the numbers of rules is extremely large, Wentian Li (1989) has commented on the 5 neighbour rules :

"Even if we can produce a spatial-temporal pattern from each rule in 1 second, it is going to take about 138 years to run through all the rules. Considering the redundancy due to equivalence between rules upon 0-to-1 transformations, which cut the time by half, it still requires a solid 69 years."

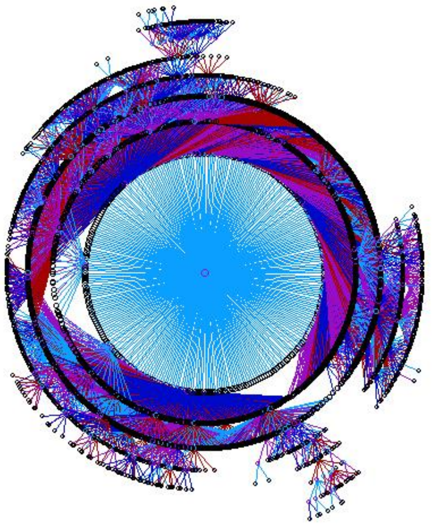
| **Rule type** | **Total number of rules** |
|---|---|
| v2k3 | 2^(2^3) = 256 |
| v2k4 | 2^(2^4) = 65536 |
| v2k5 | 2^(2^5) = 4294967296 |
| v2k6 | 2^(2^6) = 1.844674407370955e+19 |
| v2k7 | 2^(2^7) = 3.402823669209385e+38 |

*Table 1. Total number of CA rules for v2k3 to v2k7 of one dimension.*

The total number of CA rules is a function of the number of states and the size of the neighbourhood. A two state rule with three neighbourhood cells is termed a v2k3 rule and contains a total of 256 rules. Table 1 shows a summary of the total number of rules for 1D binary CA with neighbourhoods of 3 to 7 cells. As the neighbourhood is increased there is dramatic increase in the total number of rules.

CA behaviour is often described based on a subjective assessment of its space-time images. As a composer wishing to use complex behaviour my first question was, what is it? A thorough account of complex behaviour, its automatic classification by various parameters and example rules is given in (Wuensche 1997 & 1999) and a brief overview is now presented.

Complex behaviour emerges (self-organises) in spacetime from random initial conditions to form gliders, particles or self-sustaining patterns existing over a uniform or periodic background. A glider can be seen as a dislocation or defect of this background. At least 150 cells, and more cells for larger k neighbourhood sizes, are required before complex behaviour can emerge. Attractor basin topology will typically consist of long transients of complex behaviour leading to small attractor cycles. An example of this attractor basin topology is shown in Figure 6. The top shows a subtree section entering the attractor cycle and below are 8 transient levels of a subtree from a randomly generated seed. This rule has been used for the composition described in the section Composition Process as Experiment.



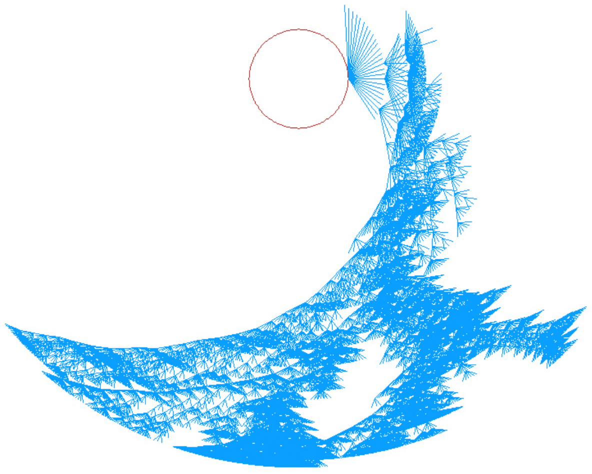


*Figure 6. Partial subtree of an attractor cycle (top) and eight transient levels from random seed (bottom) for complex rule 978ecee4*

## Evaluation Strategy

Reflection-in-action is a method of researching within a practical context placing priority on the interest in change, and the logic of affirmation and exploration. The generative music composition process thus relies on experiments that demonstrate change and affirm actions.

The structure of reflection-in-action in practice based research, termed reflective practice, suggests move testing experiment as a useful methodology. In move testing an action is undertaken to produce intended change. Good results will affirm that move and bad results will negate it. Importantly unintended results that are good, should also affirm the move. Reflection on outcome and potential for future directions will be discussed.

Technical criteria for evaluating sound synthesis techniques were suggested by Jaffe (1995), however these are recognised as guides not rigid definitions. These criteria were suggested in the context of sound synthesis, however some apply to generative music production with MIDI. The most appropriate criteria in this context will now be discussed. How efficient is the algorithm? Defined as an extremely important, yet context dependant criterion. Three efficiency categories suggested are memory, processing and control stream. How sparse is the control stream? This relates to efficiency and has particular impact on a real time system. How robust is the sounds identity? Asks if parametric adjustment alters the sound too extremely.

What classes of sounds can be represented? Conversely this asks if a broad range of sound classes can be achieved by parametric adjustment.

Castagne and Cadoz (2003) revisited and extended these guideline criteria in the context of physical modeling sound synthesis. The context of physical modeling is relevant because it addresses the entire musical creation process and does not just represent a sound synthesis perspective. Ten criteria for evaluating physical modeling were defined in four areas: computer efficiency, phenomenology, usability of scheme and environment for using the scheme. The efficiency area is similar to Jaffe's proposed scheme. An extended phenomenological criteria is diversity of context and relates to Jaffe's robustness and sound class criteria. In usability particular emphasis is placed on modularity. Modular principles are a very important criterion in obtaining generality, power and simplicity. The environment for using the scheme has two criterion. The first studies if generation algorithms already exist and how effective they are. The second asks if there is a friendly musician oriented environment for using this scheme. Results of this work will be judged on these criteria.

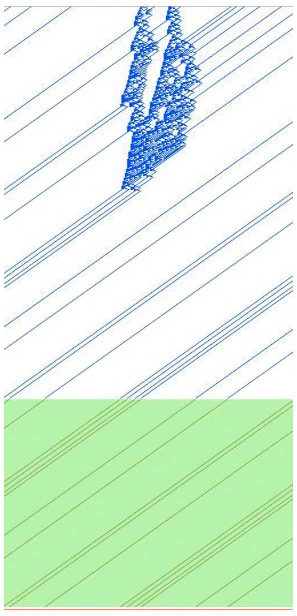
## Composition Process as Experiment

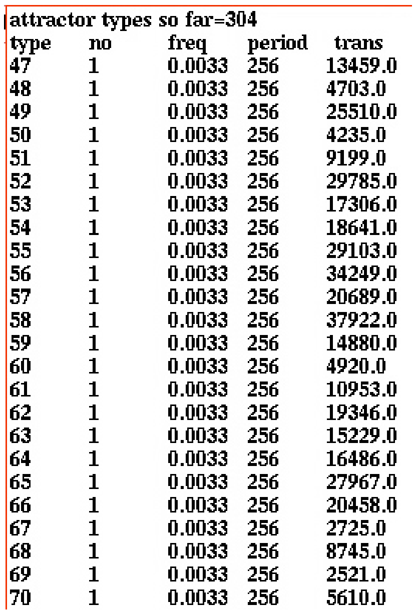
The process of composition for a piece titled "Dislocation or Defect" will now be described, and has been used by the author in other CA based compositions. The composition process is defined as a Move Testing experiment :

**Action** : Composition based on complex behaviour

**Expected Outcome** : Identify discrete stages of CA composition process. Shift of context away from Max based system.

The 1D v2k5 transition rules are specified by a 32 bit binary number usually labeled in hexadecimal. The rule chosen for this study was the known complex v2k5 rule 978ecee4 (Wuenshce 1997) with a size of 256 cells. The attractor basin topology was shown in Figure 6. For this composition 512 generations were evolved, capturing the complex behaviour well before an attractor cycle is entered. The number of generations was decided based on an attractor frequency histogram analysis from 304 random seeds using DDLab, shown in Figure 7. The top image shows DDLab locating an attractor, the shaded area, after several thousand generations. The bottom image shows a sample of the attractor frequency histogram data. The average attractor period from these random seeds was 256 and the transients were typically large, often in excess of 10,000 generations.





```
attractor types so far=304
type   no    freq     period   trans
47     1     0.0033   256      13459.0
48     1     0.0033   256      4703.0
49     1     0.0033   256      25510.0
50     1     0.0033   256      4235.0
51     1     0.0033   256      9199.0
52     1     0.0033   256      29785.0
53     1     0.0033   256      17306.0
54     1     0.0033   256      18641.0
55     1     0.0033   256      29103.0
56     1     0.0033   256      34249.0
57     1     0.0033   256      20689.0
58     1     0.0033   256      37922.0
59     1     0.0033   256      14880.0
60     1     0.0033   256      4920.0
61     1     0.0033   256      10953.0
62     1     0.0033   256      19346.0
63     1     0.0033   256      15229.0
64     1     0.0033   256      16486.0
65     1     0.0033   256      27967.0
66     1     0.0033   256      20458.0
67     1     0.0033   256      2725.0
68     1     0.0033   256      8745.0
69     1     0.0033   256      2521.0
70     1     0.0033   256      5610.0
```

*Figure 7. Attractor search of complex rule 978ecee4 finding a period 256 attractor (top) and a sample of attractor histogram data (bottom).*

Mathematica 5 (Wolfram 2005) was used to generate four CA evolutions and convert the binary output into decimal format text files. The code for generating the four data files is shown in Figure 8. This code is quite general and can be used for any size, generations and number of data files for a CA rule conforming to Wolframs **k2rn** neighbourhood convention. The built-in function CellularAutomaton was used to create a binary list of data for each randomly seeded evolution. An example of one of the four data sets used in this composition is shown as a spacetime plot in Figure 9 (left). A close up section of the right edge of this plot is shown in Figure 9 (right) where the complex behaviour can be seen more clearly.

```
CAlength = 256;
generations = 511; (* TOTAL NUMBER OF
GENERATIONS - 1 *)
init = {};
numfiles = 4;
For[numdata=0, num-
data<numfiles,numdata++,
    For[i=0, i<CAlength, i++,
      AppendTo[init,Random[Integer]];
       ];
    singlerun =
    CellularAutoma-
ton[{16^^978ecee4,2,2},
init,generations];
    declist = Table[ FromDigits[
singlerun[[j]],2],
{j,1,generations+1}];
    cadecfile = StringJoin["v2k5-data"
, ToString[numdata],".txt"];
    Export[cadecfile,declist,"Table"];
    init = {};
    ]
```

*Figure 8. Mathematica 5 code for generating the data files.*

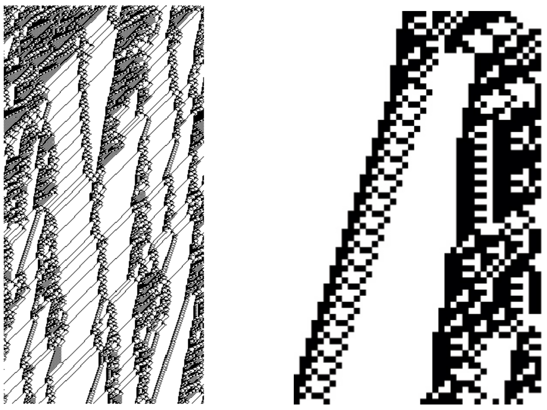
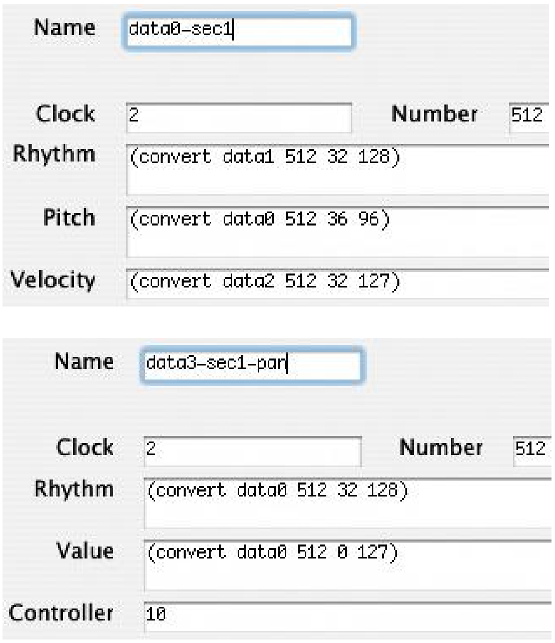


*Figure 9. Sample spacetime plot of complex behaviour (left) and close up of bottom right hand edge section (right).*

The mapping process was implemented in AC Toolbox (Berg 2004) using four **Stockpile** objects as a repository for the CA data. Mappings were produced for MIDI note, velocity, rhythmic timing and pan. The **convert** command was utilized in order to rescale the 2^256 data range to the desired compositional ranges. Two examples of this process are shown in Figure 10 detailing Data Section and MIDI Controller Data Object construction. The Data Section is used to produce note based events and the MIDI Controller for the pan events. Fifteen data objects were constructed from permutations of the four CA stockpiles and then exported to MIDI files.

The composition was produced to have a dislocated and defective feel. Aliased percussion sound samples were used in both a percussive context emphasizing defectiveness, and also in short looped fragments to create dislocated tones. An unrecognisable spoken word fragment was also used to contribute to the mood. These samples were loaded into the Reaktor (NI 2005) ensemble rAmpler and presets were constructed to implement aliasing and looping. The MIDI files were auditioned and composition elements were constructed as audio files at the original generated tempo, and also at 1/4 and 1/2 tempo. The composition was completed in Logic Audio (Emagic 2005) where these elements were arranged, edited, equalised and mixed.



| | |
|---|---|
| Name | data0-sec1 |
| Clock | 2 |
| Number | 512 |
| Rhythm | (convert data1 512 32 128) |
| Pitch | (convert data0 512 36 96) |
| Velocity | (convert data2 512 32 127) |

| | |
|---|---|
| Name | data3-sec1-pan |
| Clock | 2 |
| Number | 512 |
| Rhythm | (convert data0 512 32 128) |
| Value | (convert data0 512 0 127) |
| Controller | 10 |

*Figure 10. Example AC Toolbox dialogues showing MIDI note and controller mappings.*

## Results

Experimental results are now evaluated in terms of the four criteria areas previously introduced; computer efficiency, phenomenology, usability of scheme and environment for using the scheme.

In terms of memory the process is very efficient using only four data sets of 512 x 256 bits. The resulting text files containing the decimal conversions are 40k each. The size of this file will obviously depend on both the number of cells and the number of generations. Expanding the number of cells to 1024 and generations to 2048 resulted in a 620k file size. The data was generated using 1.5 GHz Mac G4 Powerbook. The processing time for the creation of the four files used in the composition was almost imperceptible on initiating the computation in Mathematica. For larger files, such as 1024 cells by 2048 generations, about 1 second per file would be required. The control stream is highly dependent on the CA output when mapped to short timing clocks. In these cases a high number of events can occur in a short space of time if the CA is outputting small state values. This can have the effect of delaying the output of MIDI data or overloading the MIDI buffer.

A reasonable diversity of material was produced during the composition process. CA were employed in the production of percussive and tonal elements at varied tempos. Unintended good results occurred in the form of repeated note sections. These were a product of the linear mapping from the large data range. A deterministic CA will never produce repeated output and then depart from this without external input, such as reseeding or changing the rule in a time dependent manner. It should be noted that greater diversity could be achieved by using multiple rule sets, for the focus of this work it was decided to limit the composition to a single rule for clarity of description.

The important principle of modularity has been achieved in an abstracted sense. The basic process of data file production and mapping has the advantage of being reasonably universal for CA parameters of rule, size, generations and seed. Creating a composition based on different rules for each parameter is achievable by this method. More importantly the problems associated with modular criteria identified in the authors previous approach using Max have been overcome.

The overall environment for using the scheme is straightforward and easy to use. Generation algorithms exist in the context of both visualising CA dynamics and generating data. Global dynamics is easily studied on a variety of platforms with DDLab, with the benefit of producing meaningful measures on CA behaviour. With a minimal amount of coding it was possible to create CA data files from Mathematica's built in CellularAutomaton function and importing this into AC Toolbox is a straightforward operation. Auditioning of material sometimes occurs within the AC Toolbox environment, but the playback of elements is only possible from their start points. Detailed auditioning is performed by exporting the MIDI file and using Logic Audio to access arbitrary points within the file. The problems associated with environment criteria identified in the author's previous approach using Max have been overcome. The only criticism of the current approach is that each stage is performed using a different application. I personally do not have any difficulty with this as a working method and it has also helped in identifying the discrete stages of my compositional process.

## Reflections

Reflecting on the approaches presented the compositional process may be summarised as shown in Figure 11. The process diagram also indicates with dashed lines the areas where iteration may take place. The results of this work demonstrate the key importance of rule choice as a composition parameter in my arts practice. All the parameters of CA are delicately intertwined with each other but rule choices, or the type of dynamic behaviour, are most often the fundamental decision. System size, seed method and number of evolutions are decisions that occur later. The generation and examination of the dynamic behaviour is the next stage in the process. This is often in the form of a visualisation, for example in spacetime or attractor basin plots. Rejection of CA material may occur here, for example if the incorrect type of desired behaviour is produced.

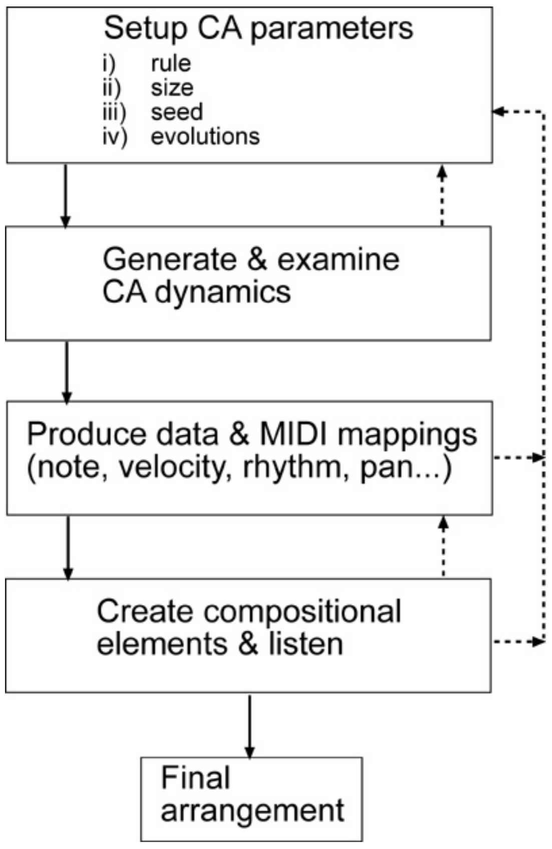

Setup CA parameters
i) rule
ii) size
iii) seed
iv) evolutions
Generate & examine CA dynamics
Produce data & MIDI mappings (note, velocity, rhythm, pan...)
Create compositional elements & listen
Final arrangement

*Figure 11. Composition process as discrete stages.*

Files containing the evolution data will then be saved and this behaviour is mapped to a set of MIDI parameters. Compositional elements are then created, usually in the form of MIDI sequences, and auditioned in a musical context. The musical context may or may not have been chosen in advance. Rejection of composition material takes place at the auditioning stage, based on aesthetic judgements. After this stage is completed satisfactorily the final arrangement of the musical work takes place.

A number of open questions occur when reflecting on using CA for generative music. Is it currently possible for an engineer, artist or scientist to buy a CA chip or hardware module of any kind? Is there a market for such a device in any type of appliance, whether consumer, artistic or scientific? How much space would a device occupy in a studio? What are the differences between hardware and software implementation? How would the interface be presented? To date the only music industry interest in these systems was briefly during the early work of Peter Beyls. Music industry interest would surely require the additional criteria of cost and development time to be factored into the evaluation.

## Conclusions and Future Work

Describing the creative process has successfully exposed the main steps of generative music composition with CA. This paper focused on complex behaviour but the methods of investigation, data production and mapping are universally applicable for composition with one dimensional CA of any size or behaviour. The data production stage particularly allows for general investigation with other software capable of data import. The process as a whole is also applicable to data sonification of CA.

The shift of context away from Max has helped to solve the problems identified in the author's previous work. Immediate interaction with the system has been reduced and the role of the CA within composition has increased. The mappings described have been applied to four primary parameters of musical composition being pitch, loudness, rhythmic/timing and spatial location.

Future work will build on this foundation now that a process for data production and mapping is in place. Areas of future study may take the form of multiple rule composition, timbre modification or non-linear mapping.

A final open question : Given the vast behaviour space available from such a simple mechanism as CA, "patterns for free" to paraphrase Stuart Kauffman, when will the instrument manufacturers show an interest?

## Acknowledgements

The author would like to thank Andrew Wuensche for his continued support in allowing me to use data and present images made with DDLab software. The entropy plots of Figures 4 and 5 are originally presented in Wuensche (1997 & 1999), and have been computed and graphed for this paper by the author.

## References

Beyls, P. 1980. *Action.* Exhibition catalogue, Kindt Editions, Belgium.

Beyls, P. 1989. The Musical Universe of Cellular Automata. In T. Wells & D. Butler, Eds., *Proceedings of the 1989 International Computer Music Conference.* International Computer Music Association pp34-41.

Beyls, P. 1990. Musical Morphologies from Self-organising Systems. *Interface, Journal of New Music Research,* 19(2-3), pp205-218.

Beyls, P. 2003. Selectionist musical automata : Integrating explicit instruction and evolutionary algorithms. *IX Brazilian Symposium on Computer Music.* Brazilian Computing Society.

Beyls, P. 2004. Cellular Automata Mapping Procedures. . *Proceedings of the 2004 International Computer Music Conference.*

Berg, P. 2004. *Using the ACToolbox.* www.koncon.nl/ACToolbox

Brown, P. 2002. Opportunities for Evolutionary Music Composition. *Proceedings of the 2004 Australasian Computer Music Conference.*

Burraston, D. Edmonds, E. Livingstone, D & Miranda, E. 2004. Cellular Automata in MIDI based Computer Music. *Proceedings of the 2004 International Computer Music Conference.*

Burraston, D. & Edmonds, E. 2004. Global Dynamics Approach to Generative Music Experiments with One Dimensional Cellular Automata. *Proceedings of the 2004 Australasian Computer Music Conference.*

Burraston, D. 2005a. *Research.* www.noyzelab.com

Burraston, D. 2005b. One Dimensional Cellular Automata Musical Experiments with Max**.** *Proceedings of the11th International Conference on Human-Computer Interaction*. HCI International. (Invited paper)

Castagne, N. & Cadoz, C. 2003. Ten Criteria for Evaluating Physical Modelling Schemes for Music Creation. *Proc. of the 6th Int. Conference on Digital Audio Effects.*

Dornbusch, P. 2002. Composers Views on Mapping in Algorithmic Composition. *Organised Sound* 7(2): 145-156.

Emagic. 2005. *Logic Audio.* www.emagic.de

Jaffe, D. 1995. Ten Criteria for Evaluating Synthesis Techniques. *Computer Music Journal.* 19(1):76-87

Langton, C. 1990. Computation at the Edge of Chaos : Phase Transitions and Emergent Computation. *Physica D*. 42, 12-37.

Li, W. 1989. Complex Patterns Generated by Next Nearest Neigbors Cellular Automata. *Comput. & Graphics* (13)4 pp531-537

Millen, D. 1990. Cellular Automata Music. In S. Arnold & D. Hair, Eds. *Proceedings of the 1990 International Computer Music Conference.* San Francisco: ICMA pp314-316.

Millen, D. 1992.Generation of formal patterns for music composition by means of cellular automata. In A. Strange Ed. *Proceedings of the 1992 International Computer Music Conference.* San Francisco: ICMA pp398-399.

Millen, D. 2004. An Interactive Cellular Automata Music Application in Cocoa. . *Proceedings of the 2004 International Computer Music Conference.*

Millen, D. 2005. *CAM Software.* comp.uark.edu/~dmillen/cam.html

McIntosh, H. V. 1990. What Has and Hasn't Been Done With Cellular Automata. http://delta.cs.cinvestav.mx/~mcintosh/newweb/marcowhat.html

NI. 2005. *Reaktor.* www.native-instruments.com

Schon, D. 2003. *The Reflective Practioner.* Ashgate Publishing

Wolfram, S. 1984. Universality and Complexity in Cellular Automata. *Physica D*. 10D, 1-35.

Wolfram, S. 2005. *Mathematica 5.* www.wolfram.com

Wuensche, A. & M. Lesser. 1992. *The Global Dynamics of Cellular Automata : An Atlas of Basin of Attraction Fields of One-Dimensional Cellular Automata.* Addison-Wesley. (Available as PDF from www.ddlab.com)

Wuensche, A. 1997. *Attractor Basins of Discrete Networks.* Cognitive Science Research Paper 461, Univ. of Sussex, D.Phil thesis.

Wuensche, A. 1999. "Classifying Cellular Automata Automatically: Finding gliders, filtering, and relating space-time patterns, attractor basins, and the Z parameter", *Complexity*, Vol.4 no.3, 47-66

Wuensche, A. 2005. *The DDLab Manual.* Discrete Dynamics Inc. www.ddlab.com