Simulating Music Composition using 1D Cellular Automata[*]

Badr Aldeen Alhaffar,[†] Wesley Concepcion,[‡] Jonny Fredriksson,[§] and Jules Sintes[¶]
Chalmers University of Technology
Simulation of Complex Systems
FFR120/FIM750
(Dated: May 16, 2023)

Cellular automata are a grid of cells in which every cell have a definite state (0 or 1) that changes every time step on the basis of simple and fixed rules. Cellular automata have many applications like generating random numbers, modeling physical and biological systems etc. In this report Cellular automata will be used to generate audibly pleasing composition. The different visual structures which emerge from such systems are well known, e.g. the Sierpinski triangle. However, we aim to explore a different approach, namely interpreting these structures as audible outputs, i.e. music compositions. By using a 1D cellular automaton, can we produce audibly pleasing music to listen to? Here, we show that, with a given rule set, and carefully cropping the 1D cellular automata, we are able to generate audibly pleasurable music. The following report proves that using cellular automata to generate audibly pleasurable music is possible when appropriate constrains are applied. Post processing is important to have when using CA to complexify the composer. Our simulations were judged by our peers with the understanding that what was being composed was simulated and would not compare to the great human compositions. However, our goal was to be able to generate audibly pleasing music that our peers enjoy listening to using cellular automata. Using simulated music can be beneficial for many different industries. By generating unique and new music, one can avoid copyright infringement and the costs associated with composing music for the specified need.

I. INTRODUCTION

The emergence of machine learning algorithms and especially deep learning created new techniques of music production. Research and development in the field of music for automatic music composition is highly active thanks to company such as iZotope or Aiva and even fundamental research laboratory such as the CSL in Paris who works on new data-driven product for music production [2].

Before new advanced data-science techniques, more simpler algorithms such as genetic algorithms or cellular automata have been used to generate music and melody [1]. Cellular automata, firstly theorized in the 1940's, have been widely used in automatic music composition project [3] and even using hardware architecture [6]. Thanks to the complex structured behaviour of cellular automata, it can be used to generate melody [7] and it can be combined with other algorithms such as neural network [5] to create more advanced tools for music composition. Automate music composition could have several applications for example in therapy or in sound-design, or even for video game and film soundtracks. Thus, having a cost-less tool that can compose music following specific rules could be useful.

By starting from a very simple cellular automata, we will try to interpret such automata in terms of sounds and music just like tools such as Wolfram tones [9, 4]. We want to see how far we can take such an automate composer created using complex adaptive systems theory. And more specifically we want to explore :
How to create, improve and complexify an automated composer starting from a simple one-dimensional cellular automata ?

A. Background information

Cellular Automata
A one-dimensional cellular automaton's universe is made up of cells arranged in an infinite line. Each cell has one neighbor one the left and one neighbor on the right. As a result, when examining the status of a cell and its two closest neighbors, there are only $2^3 = 8$ possibilities. Starting from the first generation (first line), the cell will change it's state depending on it's current state and the state of it's neighbors. This is defined by rules which need to be specified at the beginning. A two-dimensional figure is usually used to show the generation sequence of a one-dimensional cellular automaton. Starting with the parent generation at the top row and moving down with the next generation, each row represents a generation as shown in Fig 1. Despite being limited to one dimension in this case and appearing simple, they have remarkable features and can be used in important application. In this project they will be used to generate music.

[*] A footnote to the article title
[†] aldeen@student.chalmers.se
[‡] wesleyco@student.chalmers.se
[§] jonnyfr@student.chalmers.se
[¶] sintes@student.chalmers.se

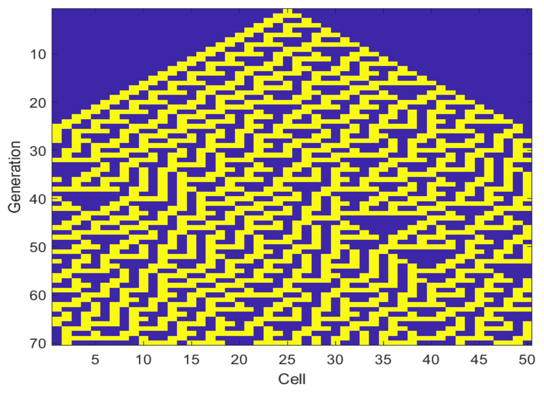


FIG. 1: One dimensional cellular automata. The first row shows the first generation with only one alive cell then next row shows how the second generation has evolved. This figure has been generated by using Rule 30.

MIDI

Musical Instrument Digital Interface, commonly called MIDI, is a communication protocol and a file format dedicated to music. It was firstly implemented in the 80's to normalized communication between instruments and computer with Digital Audio Workstation (i.e software for music). In this project, we use the file format MIDI as the output.
MIDI does not define audio signal but only command information. It provides information on starting and stopping a note with a certain velocity. All audible notes from C1 to G9 and velocities are encoded as a 7bits integer (from 0 to 127). MIDI files can be read by virtual instruments (synthesizer) to create an audio signal but also by music software to get a readable music sheet to make the music actually playable by a musician.

II. METHODS

A. General framework of the project

In this project, we chose to work with Python since it provides both the algorithmic tools to build and visualize cellular automata and a complete user-friendly library (MIDIUtil) to generate MIDI files. Thus, a single python script is needed to generate the cellular automata and transform it into a MIDI file. Then, it can be processed to generate both an audio signal (using VST) or music sheet (with MuseScore software).

B. Generating Cellular Automata

The tracks generated in this project are based on random rules that are generated by a python script. The rules consider the state of the cell and its nearest four neighbours in one approach and its nearest two neighbours in another approach. The width of the cellular automata is a parameter and we consider it as a torus to avoid boundary conditions problem. The seed of the cellular automata can be a single cell alive or a random-generated seed with a given density of alive cells.

In order to make the music more organic, we chose to associate the density of living cells at a given time step to the velocity (i.e. volume) of the note played at this time. It creates dynamics in the music and make it more human like. Moreover, to create harmony and avoid too many notes played at the same time by the same instrument, we tried a different approach to get music with several instruments at the same time. Firstly, we tried to take several windows on the same cellular automata and assign each window to a different track. However, it leads to music with too many notes and instruments which sounds muddy. Secondly, we considered multi-states cellular automata where a cell has more than 2 possible states (one dead state and several living states) and each living states is assigned to a different track.

C. Generating a MIDI file

Written music has been discretized for several centuries and there are 12 different notes usually called with letters A to G in English system. The $A_4 = 440Hz$ is the fundamental note used in western music to define all the others. From there, the notes are repeated cyclically by doubling the frequencies since two notes $N_2 = 2^n N_1$ sound very similar.
In music theory, a scale is a set of ordered notes defined by the interval between the notes starting from a root note. For example, the common Major Scale is defined by the interval between the notes *[2,2,1,2,2,2,1]* starting from a root note. If the root note is a C, then the C-Major scale corresponds to the white keys of the piano. There are a lot of different scales and each gives different colour to a music. While Major Scale are commonly used for "happy" songs, Minor Scale would usually lead to melancholic music. A scale can contains from 1 (primitive music) to 12 notes (chromatic scale that contains all the notes).

The cellular automata method coupled with MIDI file generation is a very intuitive approach. Each cell of the automata can be linked to a specific note and every time the cell is alive the note is played. It can be easily compared to a piano keyboard.

In music, the rhythm is built upon the subdivision of time also called "beat". Then each time step of the automata corresponds to a time step of the music which can be either a beat or its subdivision, commonly a quarter-beat.

Using this approach, the output of a simple 1D CA can be derived to notes with a certain rhythm which creates the basis of the music. We also tried to derive rules for velocity of the notes to make the music more organic and lively.

The "CA to MIDI"-converter takes a raw cellular automata output, a root note and a scale (so a list of interval). Then, each row of the output is considered as a time step of the music and is converted accorded to the rule specified by the root note and the scale. Precisely, each cell of the automata is mapped to a single note in the scale.
The converter also take the height of the music as a parameter. It specifies the size of the window in the automata output to convert and thus, it corresponds to the number of possible notes in the music. This parameter is independent from the size of the generated automata which allow us to generate MIDI from different windowing of the same automata output, to create different voices for example by using patterns of specific rules to create harmonies between voices.

D. Improving the music

Through trial and error it soon became clear that further steps were needed in order to better the quality of the music. In order to avoid false tones, i.e. tones that doesn't sound good when played together, chord constraints was implemented. Unlike the 12-tone chromatic scale or the 7-tone pentatonic scales, chords generally consist of 3-4 tones which, by design, sound especially good together. By only assigning the notes of an arbitrary chord to the cells the false notes are eliminated, hence the quality of the music is arguably improved. However, when constrained to a single chord the music becomes very repetitious, even when the CA isn't. In most human compositions there are frequent chord changes throughout the song. By forcing a change of chords every fourth time step the issue of repetitious music was addressed. However, with a constant chord progression the music approaches rule-invariance, i.e. different CA's would sound very similar. To avoid this problem we implemented CA-specific chord progression by letting the number of alive cells in each section of four time steps decide which chord to assign.

The next step in improving the music came down to simulating more human-like compositions. Even with the aforementioned implementations there was still a robotic feel to the music. Two of the most distinct robotic qualities were the constant volume and note duration. To address the latter we summed the duration of cells when alive for consecutive time steps, e.g. making one tone last for twice as long instead of playing it twice in succession. To implement a dynamic volume we chose a method closely related to the one used for chord changes, letting the number of dead cells in each time step scale the volume of the tones played in that same time step.

As a final step in the process of improving the music we have the post-editing. After converting the CA to a MIDI-file, we used MuseScore to assign different instruments to different cells, further adding dimension to the music. This would be especially useful when the aim is to generate music in a specific genre.

E. Summarizing the process

- Generate the CA with a given set of rules (figure 2a).
- Crop out the edges, leaving a centered strip of a prespecified width (figure 2b).
- Rotate the strip 90 degrees clockwise, such that the time evolution goes from left to right, starting at the end of the CA (figure 2c).
- Assign notes to each cell, updated at every fourth time step (figure 2d).
- Apply rules of dynamic volume and note duration.
- Convert to MIDI-file.
- Post-edit the music using a MIDI-software (figure 3).

III. RESULTS

Quantifying how audibly pleasing music sounds is difficult to measure, given that what one person thinks sounds good might be different than what another thinks sounds audibly pleasing. We surveyed our peers with selected outputs using different rules and constraints and asked them to provide a rating from 1-5, where 1 is the lowest score and a 5 is the highest rating. We provided to the participants that the music that was being shown was created using automated techniques.

As can be seen in Table I, the song with the highest grade was the Constrained Chord (4). This particular song was created using notes that are constrained to 6 chords. Every bar the chord changes based on the density of chords within that bar. There are 6 different chords to choose from. The rule used for generating this CA was randomly selected by the program. It is clear from the results that when adding the chord constraints, the opinion of the song is higher according to those we surveyed with all of the constrained chord songs receiving an average score of 4 out of 5 or higher. The generated CA for the constrained chord (4) sample that received the highest grade can be seen in Figure 4. The resulting music score can be found in Appendix B. The sample music files for

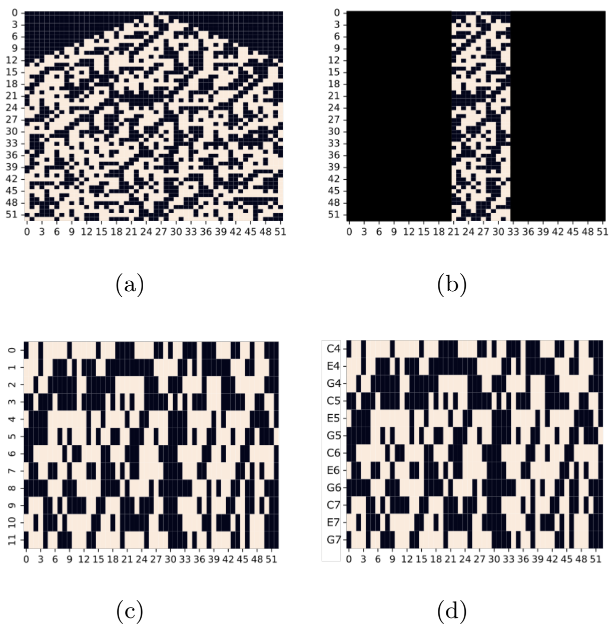


FIG. 2: (a) A cellular automaton generated with a random five-neighbour-rule. (b) With the edges cut off a strip of 12 cells remain.(c) The strip is rotated 90 degrees clockwise. The time evolution of the 12 cells runs from left to right, i.e. starting at the end of the cellular automaton. (d) After assigning a note the each cell the result resembles a music score and is ready to be converted to a MIDI-file. In this specific case the cells are constrained to a C major chord.

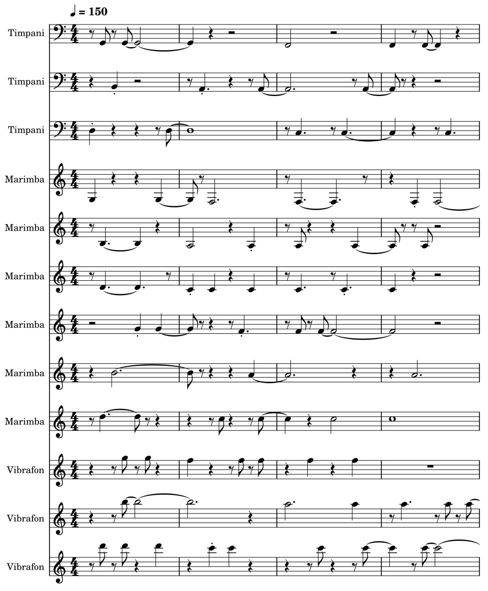
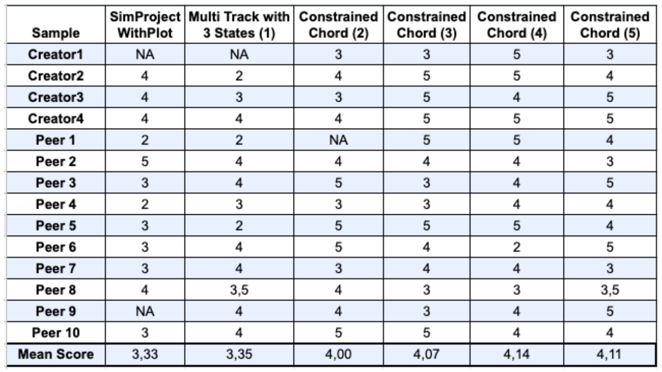


FIG. 3: The resulting MIDI-file, visualized as a music score using the software MuseScore. Different instruments can be assigned to each cell manually.

TABLE I: Peer review results of 6 different tracks different rules and constraints. Scores were given out of 5, with 5 being the highest score. A score of 1 would mean that the listener would never listen to the track again, and a 5 means that the listener thought it would was very good and would listen to the track again.



Sample	SimProject WithPlot	Multi Track with 3 States (1)	Constrained Chord (2)	Constrained Chord (3)	Constrained Chord (4)	Constrained Chord (5)
Creator1	NA	NA	3	3	5	3
Creator2	4	2	4	5	5	4
Creator3	4	3	3	5	4	5
Creator4	4	4	4	5	5	5
Peer 1	2	2	NA	5	5	4
Peer 2	5	4	4	4	4	3
Peer 3	3	4	5	3	4	5
Peer 4	2	3	3	3	4	4
Peer 5	3	2	5	5	5	4
Peer 6	3	4	5	4	2	5
Peer 7	3	4	3	4	4	3
Peer 8	4	3,5	4	3	3	3,5
Peer 9	NA	4	4	3	4	5
Peer 10	3	4	5	5	4	4
Mean Score	3,33	3,35	4,00	4,07	4,14	4,11

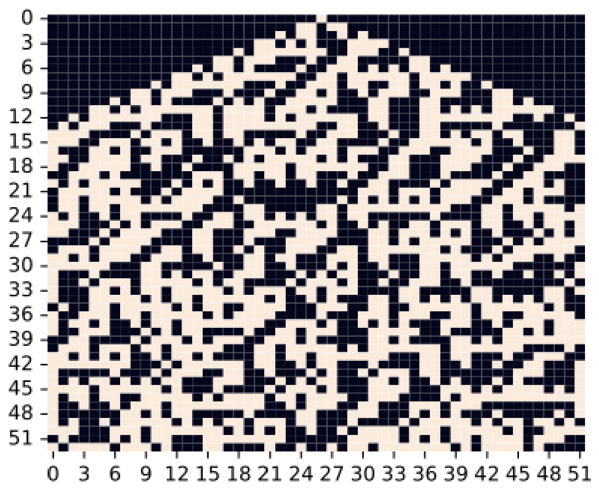


FIG. 4: Constrained Chord with Random Rule

SimProjectWithPlot and MultiTrackwith3States(1) were created using 5 neighbor and a random seed, similar to the constrained chord, but using multiple states. These results were not scored as high as other samples by our peers, and given a score around 3.3. An example CA from one of the multiple state Cellular Automata is shown in Figure 5. Different instruments were selected for the different states, which might have contributed to the score, compared to the piano almost exclusively used in the other constrained chord songs aside from Constrained Chord (5), which used three percussion instruments. It is difficult to know simply from looking at a cellular automata whether that what has been produced will sound good. This is why the post processing of the piece is important to make the track sound more audibly pleasing to the listener. One can view different patterns that appear

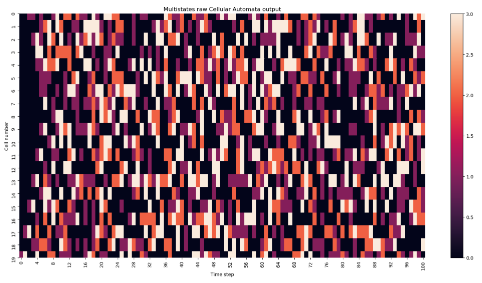


FIG. 5: Multi State Cellular Automata. The different colors represent different instruments, and black represents no note being played.

in the cellular automata and observe which notes will be played more frequently than others, but constraining the music to certain chords or keys is the most important factor for ensuring a good starting point for the music.

The audio files can be listened to online via the SoundCloud link:

https://soundcloud.com/wesley-concepcion-79845177/sets/simulation-of-complex-systems-automated-music-composition

IV. DISCUSSION

Through this project, we explore cellular automata in a different way by interpreting the results of the automata in terms of music. What makes the cellular automata interesting are the patterns that could occure with specific rules and seeds, and the most interesting rules are the one with large patterns that are not too repetitive which makes the melody interesting without being too boring and not too chaotic. Having such rules was impossible using 3 neighbors and that's why we worked with 5-cells neighborhood. Despite the simplicity of our model, it can creates nice melodies and there are very large possibilities to improve and complexify the model. The dynamics added with the global density of living cells was one example as well as chord constraints, but many others improvement are possible to make the generated music more interesting, pleasant and human-like.

It is hardly controversial to claim that there is no consensus on what constitutes good music. There are, however, certain principles which are implemented in most popular music. One such principle regards false notes, i.e. sets of notes that does not sound good when played together. With this in mind the initial constraint from the chromatic scale to the pentatonic major/minor scales, inspired by [4], naturally led us to the chord constraints used in this project. Since there are numerous different chords the choice of constraint we have chosen are highly subjective, and it is possible that a different set of chords would be more or less appealing to the consumer. However, our choice was an attempt to approximate generality, using chords widely used in popular music [8]. The aspect of relationships between certain chord progressions and different genres, together with the assignment of different instruments to different tracks, is a very interesting route in the further development of this product, nearing the impressive implementation done by WolframTones [9].

V. CONCLUSION

By layering different instruments or sounds, we can add dimension to the music. Further constraining the music to a certain scale, this ensures that all of the different instruments will layer in harmony. By choosing a rule that shows a repeating pattern, we have now set the baseline or the beat of the song for which a melody can be added on top. In MuseScore, we can set the baseline to be of a lower starting note such that there are multiple pitches that can be picked up by the listener. By setting the baseline to be a repeating pattern, a higher pitch melody can be added to further add emotion to the song. If we reference **figure 3**, we can see that the lowest notes are set to the timpani, or commonly known as kettledrums, while the higher pitch notes are set to the vibrafon and marimba. The combinations of these three instruments of different pitch while kept in the same key will almost automatically produce an audibly pleasing song.

Simulating music is not a novel idea. There are many different strategies to try to simulate automatically generated music. The most advanced being the machine learning approach using deep learning and neural networks. However, the most advanced automated techniques require the creative input to learn from previous compositions. In order to generate complex music without the aid of human composition, we employ the cellular automata to generate the chaotic patterns and by post processing we are able to polish the composition to sound pleasing to the listener.

We have not explicitly tried every rule for generating the cellular automata to find the perfect rule. However, given more time, we would be able to find which rules work with different constraints, and provide an improved experience for the listener. However, the biggest improvement lies in the post processing. Being able to process the output to provide a better final result can be viewed in parallel as a human composition, albeit a different technique. Through our simulations and tests, we have shown that we can generate audibly pleasing music as confirmed by our peers.

VI. CONTRIBUTIONS

All members of the team contributed equally to the project.

REFERENCES

[1] E. R. Miranda J. A. Biles (Eds). *Evolutionary computer music*. Springer-Verlag London Limited, 2007.

[2] Cyran Aouameur, Philippe Esling, and Gaëtan Hadjeres. "Neural Drum Machine : An Interactive System for Real-time Synthesis of Drum Sounds". In: *arXiv preprint arXiv:1907.02637* (2019).

[3] Dave Burraston Ernest Edmonds. "Cellular automata in generative electronic music and sonic art: a historical and technical review". In: (2005), pp. 165–185.

[4] Jeff Holtzkener. *Listening to Elementary Cellular Automata*. 2018. URL: https://medium.com/code-music-noise/listening-to-elementary-cellular-automata-661018229362 (visited on 01/04/2022).

[5] A. P. Oliveira I. D. Matic and A. Cardoso. "Automatic Melody Generation using Neural Networks and Cellular Automata". In: Faculty of Electrical Engineering, University of Belgrade, Serbia. 2012.

[6] Luiza M. Mourelle Nadia Nedjah Heloísa D. Bezerra. "Automatic generation of harmonious music using cellular automata based hardware design". In: *Integration, the VLSI Journal 62* (2018), pp. 205–223.

[7] L.B. Soros Omar Delarosa. "Growing MIDI Music Files Using Convolutional Cellular Automata". In: *IEEE Symposium Series on Computational Intelligence* (2020).

[8] Wikipedia contributors. *I–V–vi–IV progression — Wikipedia, The Free Encyclopedia*. [Online; accessed 9-January-2022]. 2022. URL: https://en.wikipedia.org/w/index.php?title=I%E2%80%93V%E2%80%93vi%E2%80%93IV_progression&oldid=1064447651.

[9] *Wolfram Tones*. 2022. URL: https://tones.wolfram.com/generate (visited on 01/04/2022).