Procedural mixed-initiative music composition with hierarchical Wave Function Collapse

Pál Patrik Varga, Rafael Bidarra
VargaPalPatrik@student.tudelft.nl,R.Bidarra@tudelft.nl
Delft University of Technology
Delft, The Netherlands

ABSTRACT

Mixed-initiative music composition systems are sought after by both amateur and professional musicians. A well-known PCG algorithm is Wave Function Collapse (WFC), which has recently been extended with various mixed-initiative capabilities [6]. So far, WFC has mainly been used for generating video game levels, textures and graphical patterns. This demo presents and illustrates an approach for an interactive WFC-based music composition editor.

CCS CONCEPTS

• **Applied computing → Sound and music computing**.

KEYWORDS

procedural content generation, mixed-initiative, wave function collapse, music composition

ACM Reference Format:
Pál Patrik Varga, Rafael Bidarra. 2023. Procedural mixed-initiative music composition with hierarchical Wave Function Collapse. In *Proceedings of 14th PCG Workshop (PCG Workshop 2023).* ACM, New York, NY, USA, 3 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

The bloom of AI in recent years has led to a landscape of AI-based art tools available to the public. Most of these tools focus on textual and visual content, although recently some research prototypes have been focusing on generation of audio content [1, 3, 4]. However, many of these systems hide the exact factors used by the machine when taking generative decisions. One question that arises is: *how could a generative system give users access to the content structure, and even let them (re)define the rules to manipulate it?*

Wave function collapse (WFC) is a widely used algorithm that is powerful yet simple enough to explore for these purposes [2]. Although it has been used extensively for level and image generation [5], its potential regarding music composition is yet to be harnessed.

This project aims at designing and exploring a mixed-initiative WFC-based editor for music composition. Using it, a composer can set up guidelines by defining *constraints* and *prototypes*, Moreover, it also allows for directly placing some notes or chords on the canvas. The system will then "fill in the gaps", creating a unique interpretation of the input guidelines. Of course, running it multiple times from the same point may lead to wildly different final results, depending on how strict the constraints are.

Another – similar – use case is the endless generation of music, for example, section-by-section. One could even 'tweak' the guidelines while the music is playing, and the changes would, say, kick-in when generating the next section, about to be played. This could easily be used for generating live music for video games.

PCG Workshop 2023, April 11, 2023, Lisbon, Portugal
© 2023 Copyright held by the owner/author(s).
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/XXXXXXX.XXXXXXX

2 WFC IN A NUTSHELL

In WFC, the *canvas* is composed of *cells*, each of which can be collapsed into a *state*. For example, when generating an image, each pixel could be a cell, and the possible states could be colors. The algorithm requires as input the set of allowed states and a set of *constraints*. These constraints dictate the relationships neighboring cells can have. The system then collapses the remaining cells one-by-one: it chooses one of the possible states for a cell in *superposition*, collapses it, then updates the possible states of the cell's neighbors accordingly.

3 HIERARCHICAL STRUCTURE

An inherent challenge of using WFC for music composition is that traditional WFC only looks at the neighbors of a given cell, thus excluding the opportunity to conform to a higher-level pattern within the piece. To address this, we propose a hierarchical structure for musical composition, so that at each level, WFC will be invoked to collapse cells for the respective canvas. Characteristic of this hierarchy is that the state of each collapsed cell in a high-level layer (e.g. *section*) defines constraints for collapsing cells in a low-level layer (e.g. *measure*). Figure 1 illustrates this relationship, depicting canvas examples at both levels (representing cell states by colors, for clarity). The rationale is that on higher-level layers, cell states do not necessarily have to stand for 'audible entities', but rather be something more abstract, that represents, for example, (a set of) constraints for the next layer(s) below. The hierarchical WFC

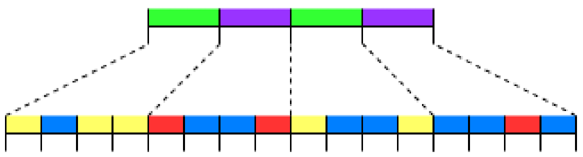


Figure 1: A WFC hierarchy with two layers. The green cell only allows for blue and yellow cells, the purple cell only allows for red and blue cells on the layer below.

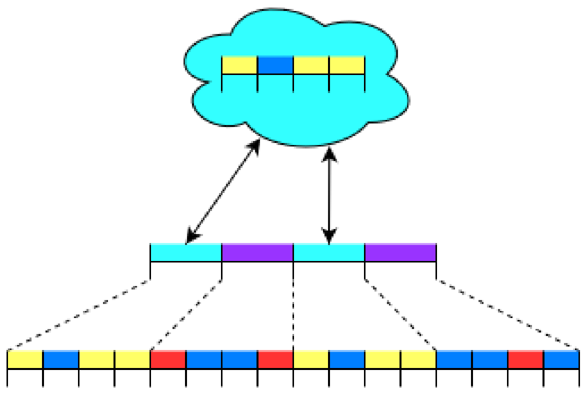


Figure 2: An example of using prototypes. A prototype *yellow-blue-yellow-yellow* is defined (top), and associated to the cyan state. Therefore, every cell on the higher-level layer that collapses to the cyan state will impose the contents of the prototype on the lower-level layer, by default. Note the repetition of the *yellow-blue-yellow-yellow* "theme".

algorithm takes each collapsed cell on a high-level layer, creates a new canvas for it at the next layer below, and imposes on it its own set of constraints. WFC will then run on this canvas, collapsing each of its cells subject to all constraints imposed at the present level, including those received from higher-levels.

For the purposes of this demo, the following layers are considered:

- **Base**: this top layer represents those properties that do not change throughout the piece, and therefore constrain all layers. This may include notions like tempo, key, and default duration of chords. This layer can be thought of as having a canvas with one single cell spanning the entire piece.
- **Section**: at this level, we define the large segments within the piece, standing for notions such as "intro", "verse" and "chorus" in traditional pop song structures, or "coda" in classical music. It can define constraints, for example, for chord progressions or general guidelines for the rhythm.
- **Measure**: a section can be seen as divided into a number of measures. At this level, the building block of a measure is the note. A canvas on this layer will contain (a snippet of) a melody.

Overall, melodies are built by composing measures (out of notes and rests), which integrate the various sections. At each layer, this is done subject to the constraints imposed from higher-level layers, concerning for example note length, amount of rests, and the direction and size of melody steps. For example, a chord progression generated at the section layer is typically used to make decisions at the measure layer. Figure 2 presents an example of this top-down constrained collapsing, featuring the notion of *prototype*: a pattern of states that can be reused at various points of a canvas (independently of the layer).

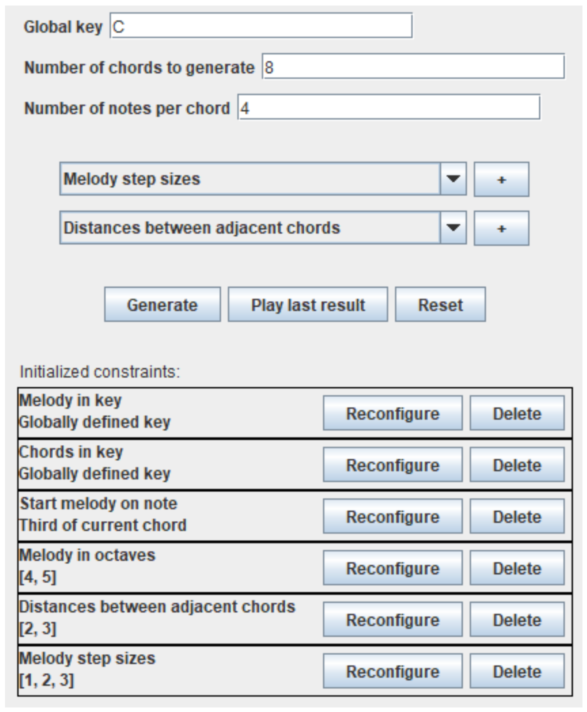
4 CONSTRAINTS AND REPETITIONS

To be able to make sensible decisions when collapsing a cell, we can distinguish between two types of constraints: hard and soft. Hard constraints are used to rule out options that should definitely not be in the given cell, narrowing down the possibilities the algorithm has to choose from. An example of this would be using a set of notes belonging to a given musical key. With a soft constraint, the user assigns a weight to each option, expressing the extent to which it is desirable. These weights are then used to choose a status, when collapsing the cell. For example, one might use a soft constraint to indicate a preference for ascending intervals in the melody, without necessarily ruling out a descending interval.

A very important notion in music composition is repetition. This can be conveniently addressed through the notion of prototype defined above. Once the user creates a prototype, it can be used by the algorithm for collapsing some higher-level cell in a burst, instead of generating all of its lower-level content sequentially. This gives a piece a more cohesive structure, in which recurring themes are present. Moreover, prototypes can also be mutated after being referenced, to allow for more variation.

5 THE DEMONSTRATION

This interactive demo illustrates the basic usage of the mixed-initiative composition prototype system; see Figure 3. At the moment, it simply allows a composer to choose and configure various constraints from an extensive list. The composer does not have to



Global key: C
Number of chords to generate: 8
Number of notes per chord: 4

Initialized constraints:

Melody in key Globally defined key	Reconfigure	Delete
Chords in key Globally defined key	Reconfigure	Delete
Start melody on note Third of current chord	Reconfigure	Delete
Melody in octaves [4, 5]	Reconfigure	Delete
Distances between adjacent chords [2, 3]	Reconfigure	Delete
Melody step sizes [1, 2, 3]	Reconfigure	Delete

Figure 3: The basic GUI for two hierarchical levels: the system generates a series of chords (measures), and for each of them, a series of notes, according to the constraints specified.

be knowledgeable about music theory to use this tool, though that could definitely speed up the composition process.

REFERENCES

[1] Lucas N. Ferreira and Jim Whitehead. 2021. Learning to Generate Music With Sentiment. https://doi.org/10.48550/ARXIV.2103.06125

[2] Maxim Gumin. 2016. *Wave Function Collapse Algorithm.* personal repository. https://github.com/mxgmn/WaveFunctionCollapse

[3] Amy K Hoover, Paul A Szerlip, Marie E Norton, Trevor A Brindle, Zachary Merritt, and Kenneth O Stanley. 2012. Generating a Complete Multipart Musical Composition from a Single Monophonic Melody with Functional Scaffolding. In *Proceedings of the 17th International Conference on Computational Creativity.* University College Dublin, Dublin, Ireland, 111–118.

[4] Amy K Hoover, Julian Togelius, and Georgios N Yannakakis. 2015. Composing video game levels with music metaphors through functional scaffolding. In *Proceedings of the First computational creativity and games workshop.* ACC, 7 pages.

[5] Isaac Karth and Adam Smith. 2021. WaveFunctionCollapse: Content Generation via Constraint Solving and Machine Learning. *IEEE Transactions on Games* PP (05 2021), 1–1. https://doi.org/10.1109/TG.2021.3076368

[6] Thijmen SL Langendam and Rafael Bidarra. 2022. MiWFC - Designer Empowerment through Mixed-Initiative Wave Function Collapse. In *Proceedings of the 17th International Conference on the Foundations of Digital Games* (Athens, Greece) *(FDG '22).* Association for Computing Machinery, New York, NY, USA, Article 66, 8 pages. https://doi.org/10.1145/3555858.3563266