# vorbei

**by**

**Paul Paroczai**

Bachelor of Fine Arts, University of California, Berkeley, 2012

Project Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Fine Arts

in the
School for the Contemporary Arts
Faculty of Communication, Art and Technology


SIMON FRASER UNIVERSITY
Spring 2017

# Approval

| | |
|---|---|
| **Name:** | **Paul Paroczai** |
| **Degree:** | **Master of Fine Arts** |
| **Title:** | ***vorbei*** |
| **Examining Committee:** | **Chair:** **Claudette Lauzon**<br>Professor |
| | **Arne Eigenfeldt**<br>Senior Supervisor<br>Professor |
| | **Jim Bizzochi**<br>Internal Examiner<br>Associate Professor<br>School of Interactive Arts and Technology |
| | **Oliver Bown**<br>External Examiner<br>Senior Lecturer<br>Faculty of Art and Design<br>University of New South Wales |
| **Date Defended/Approved:** | April 13, 2017 |

## Abstract

*vorbei* is a program designed to generate a unique and complete piece of music every time it is run, and to do so with no external interaction. Each piece created by *vorbei* begins with the generation of a series of millisecond values known as a gesture, which provides the seed for all subsequent data and sound. Throughout a given run of the program, in order to reinforce emerging trends and encourage movement towards coherent structures, generated data is stored and analyzed to create probabilities that determine its later use. As such, individual runs of *vorbei* can be understood as a series of variations generated from increasingly extended derivations of the initial gesture.

# Table of Contents

Approval ..... ii
Abstract ..... iii
Table of Contents ..... iv

**Defence Statement ..... 1**
Introduction ..... 1
Context and Related Work ..... 2
Conception ..... 3
Description ..... 6
Reflection ..... 8
Conclusion ..... 11
Works Cited ..... 12

**Appendix A. Research Paper ..... 13**

**Appendix B. *vorbei:* A Generative Music System ..... 26**

# Defence Statement

## Introduction

While there is currently a wide variety of generative systems capable of expertly managing specific aspects of music composition, few are designed with the intention of creating every musical component from within the program itself. The still somewhat recent releases of Sony's Computer Science Laboratory Flow Machines are an excellent example of music co-written by computers and humans – the former providing a score while the latter realizes that score using instruments external to the generative software. In this case, though the Flow Machines are entrusted with the writing of a song, it's realization is left as a human task. *Vorbei* is a program designed to generate every component of a unique and complete piece of music every time it is run, and to do so with no external interaction.

Paraphrasing Marshall McLuhan's speculations on the emergence of new forms of media, Alexander Galloway explains in his book *The Interface Effect* that "a new medium is invented, and as such its role is as a container for a previous media format." The research preceding my work on *vorbei* attempted to identify what might constitute a uniquely computational practice, as opposed to one which used computers as containers for old media formats. Observing that computational processes are tied to neither the materials which make them possible nor the materials to which they are applied, I concluded that the unique ability of a computer is to codify and crystallize processes of thought. Though the original concept for my project was less concerned with directly attempting to realize a uniquely computational practice, and more focused on taking the first steps towards a specific style of music composition, I ended up somewhere between these two goals. While *vorbei* no doubt makes music, an old medium, it does so in a way

that foregrounds the unique capabilities of the container within which a given piece is generated.

## Context and Related Work

According to Philip Galanter, generative art is “any art practice where the artist uses a system...which is set into motion with some degree of autonomy contributing to or resulting in a completed work of art” (2003). Initially, I became interested in generative techniques as a means to assist my existing compositional practice. Realizing that I could describe certain musical gestures much more easily with code than with modern staff notation, I became increasingly curious about how much of the compositional process I could hand over to my computer. Though my early experiments were usually designed to generate a single musical gesture, which would then often be spliced into a fixed recording, later projects, including my audio-visual installation *Intraface*, and sound designs for a variety of theatrical productions, featured software written to operate with increasing amounts of autonomy. However, each of these works were made to be used in live settings and were in one way or another guided by external input from either myself or an audience. Having amassed a variety of techniques for generating sound and music over the course of creating these pieces, I wanted to begin designing programs that could not only autonomously generate an entire piece of music without any external interaction, but which could also be distributed in the same way as any other piece of recorded digital media. Instead of using software agents as co-performers or compositional aids for work presented in live settings, I wanted the software itself to be the entire piece.

Searching for references to guide the development of my own work towards this goal, I found a number of pieces which addressed some aspect of what I was hoping to

accomplish. Icarus' *Fake Fish Distribution* and Arne Eigenfeldt's Generative Electronica Statistical Modelling Instrument provided examples of music generated entirely by a computer, but the actual software used to generate this music was never distributed. Instead, pieces generated by these systems were recorded and then made available for download. Brian Eno's *Bloom* and *Trope*, and Joshue Ott and Morgan Packard's *Thicket*, on the other hand, offer examples of generative software made available as interactive smartphone applications. However, their significant reliance on user input limits the autonomy of the programs written to manage the components of a given piece.

The reference with which *vorbei* has the most in common is Gwylim Gold's Bronze music format, which allows a piece to rearrange itself every time it's played. Importantly, Bronze generates each iteration of a piece by selecting from a set of pre-recorded materials, meaning that a significant amount of musical decisions are made in advance of the software's operation. Regardless, as a format which enables real-time generation of music requiring no input from a user, Bronze served as a template for a number of experiments which contributed significantly to methods used in *vorbei*.

## Conception

In the early stages of the project, I planned to build four separate pieces of software, each designed around a different generative technique. The first three of these techniques were identified as "Audio File Processing," "Audio File Arrangement," and "Audio File Response." As their names suggest, each of these techniques would generate pieces based on analysis and manipulation of some amount of pre-recorded audio. However, in early tests, a number of problems arose that revealed these methods to be impractical as first steps towards the practice I intended the project to initiate. The first major problem was that high quality audio files can quickly become quite large, and

as I began to build the pieces that would use these files, I realized that creative resources were increasingly being diverted towards considerations of what would be practical to release as a download.

The second major problem that arose was perhaps the most significant in my eventual decision to exclude pre-recorded material from the project entirely. When creating fixed pieces of music using a set of either recorded or synthesized audio files, a wide variety of effects such as equalization, compression, reverb, and other signal processing tools are typically used to balance the sounds of a given piece into a cohesive mix. Often times, decisions made regarding the parameters of these effects are tied to very specific moments in a piece based on which sounds occur simultaneously. For example, while a kick drum presented in isolation might be equalized in a way that emphasizes its unique resonance, the same drum could later have certain frequencies dampened to avoid clashing with the spectral properties of whatever other instruments enter at a later point in the music. Since these sorts of signal processing tasks are inherently context dependent, managing them within a generative musical work, in which sonic relationships evolve over the course of a given generation, can be quite difficult. Recognizing that the development of a system to manage these tasks was outside the scope of my research, I decided to move away from working with pre-recorded material and to instead focus on the last of the four generative techniques I had originally planned to use.

Identified quite broadly as “Synthesis,” the goal of the fourth generative technique was to design a piece of software capable of creating every part of a complete piece of music. That is, whereas the programs using audio files would have simply arranged pre-existing material, Synthesis would not only arrange its materials, but also create the materials themselves. Of course, with as blank and broad a canvas as a computer, the

range of possible material is nearly limitless, and determining the best first steps toward designing such a program quickly revealed itself to be a difficult task.

After experimenting with a number of strategies, I determined that the best way forward would be to create a system that utilized a computer's most basic sound-making unit: the sample. Represented for programming purposes as values between -1 and 1, individual samples specify an amount of voltage which determines the extension or retraction of a speaker's cone from its resting position. While programs built around the aforementioned audio file techniques would have essentially served as containers capable of arranging and outputting collections of predetermined sample-streams for which speakers would serve only as a necessary conduit in the transformation of those samples to sounds, an approach centered around the generation and organization of individual sample values would be directly focused on managing the interaction between the computer and the speaker. Discarding algorithmic synthesis routines in favor of sample-by-sample generation, *vorbei* essentially produces a set of instructions for playing the speaker.

Having determined that samples would be the primary material of the piece, the next task was to figure out how to build a system capable of deciding what to do with this material. That is, now that I knew what the computer would be using to make sound, how could it manage decisions about when and how to use it?

Traditionally, the structure for managing decisions about when a certain sound will or won't occur in a given piece of music is rhythm. In modern staff notation, notes are drawn to indicate when and for how long a given sound should be produced. However, a note within this system requires an understanding of two additional higher level structures. The first of these is meter, which specifies a grid of all possible temporal

locations at which a sound could be produced. The second structure, tempo, determines the amount of time between any two points on this grid. While these concepts are extremely powerful in the dissemination of music amongst human performers, describing them to a computer can be incredibly problematic and potentially unnecessary.

Alongside the complexity of this task, a recognition of the extent to which I had reduced the sonic material of the piece led me to be curious about whether or not I could similarly reduce the structures guiding its decision-making. If *vorbei* would never look outside itself to determine how to generate sounds, why would it inherit externally-defined compositional methods? Eventually, this question led me to abandon rhythm, meter and tempo as the core concepts guiding sonic action or inaction. The system that emerged to replace these concepts ended up not only providing a conceptual template for the generative techniques around which the rest of the program was built, but also serving as an origin for all sound generated within the piece.

## Description[1]

A given run of *vorbei* always begins with the generation of a series of millisecond values known as a “gesture.” To create a gesture, values between 50 and 12000 milliseconds (the former being the amount of time needed between two sounds for them to be considered separate, and the latter the longest possible value in traditional musical notation – a double whole note at 40 beats per minute) are selected randomly, one at a time, until a given value is determined not to be a part of the gesture.

[1] This section gives a brief description of the system. For a more complete description, please see the Appendix.

The decision to stop generating values is based on a comparison of each newly generated value to all preceding values. If a new value is determined to be consistent enough with the trend of preceding values, it is considered to be a part of the gesture, and another new value is generated and tested. Once a new value is determined to not be part of the gesture, it is discarded, and generation of the gesture stops. Since new values are tested against the emerging trend of preceding values as opposed to any predefined trend, the computer is not being pushed in the direction of any specific behavior in the generation of this initial gesture. Instead, it is simply being asked to recognize whatever behavior ends up emerging, and to then determine whether or not the next value it generates fits with that behavior. This technique of referring the computer back to itself as opposed to directing it towards any sort of preexisting behavior not associated with its own decisions was maintained throughout the creation of *vorbei*. As a result, every piece of data generated in a given run of the program can be traced back to the initial gesture.

Once a gesture has been generated, individual durations within the gesture, referred to as “phrases,” progress independently through a series of four sections. Each section is associated with a different signal processing technique and condition for progressing to the next section. In the first section, the proportions of each phrase within the gesture determine a series of ratios which then subdivide individual phrases. These subdivisions are articulated by clicks of random amplitude. In the second section, a synthesizer uses stored sequences of clicks from section 1 as wavetables which are read through at frequencies determined by multiplying the length of a given phrase by subdivision ratios from section 1. In the third section, spectral analysis of sounds generated by the synthesizer determines specifications for filtering its output. In the

fourth and final section, data generated earlier in the piece is analyzed and adjusted to create new data, which is articulated by additional synthesizers.

Throughout each of the four sections, *vorbei* constantly analyzes the data it generates, and uses these analyses to inform future decision-making. For every sound heard, a corresponding set of data is stored and compared to all previously generated data. These comparisons are then used to construct probability tables which determine the re-use of stored data at a later point in the piece. This constant evaluation of past actions as a means for structuring future decision-making allows the system to recognize and reinforce whatever trends emerge in its behavior following the initially random seed of the gesture. As a result, *vorbei* tends to arrive at an aurally consistent behavior determined by an understanding of its own actions as opposed to by decisions based on externally-defined structures.

## Reflection

Conceived without any notion of either the sound-making parameters described by modern staff notation or the signal processing techniques common in recorded music, *vorbei* generates a complete piece of music without reference to external methods of musical syntax. Given no means for understanding rhythm, meter, or tempo, its temporal structure is determined from the proportions of a series of millisecond values related only to one another. Rules of voice leading and orchestration which tend to guide the timbral characteristics of a given piece are replaced by analyses which detect and emphasize the dominant spectral characteristics of initially noisy sources. Signal processors designed to shape streams of samples based on a certain effect or instrument are discarded in favor of systems which write and organize individual sample values directly.

All decisions regarding data used to generate any of the sounds heard throughout the course of a piece are made based on an evaluation of preceding decisions.

Referring to Wooler, et. al's framework for comparing processes in algorithmic music systems, *vorbei* is designed with elements of each of the three categories described by the authors. The derivation of every piece of data used throughout a run of the program from an initial series of millisecond values is a generative process to the extent that these durations are eventually used to determine a piece's rhythmic and timbral qualities. Analytical and transformational processes are most noticeable in *vorbei's* fourth section, during which key features of sounds heard in the preceding three sections are extracted and adjusted to produce unique sounds using the same types of data as were used to generate the sounds being analyzed.

Given the incredibly wide variety of sounds modern computers are capable of producing, I had expected that *vorbei* would yield extremely varied sonic output as a result of the openness of its compositional approach. However, though many of the sounds made by the program were surprising the first time I heard them, an even greater surprise was the timbral consistency of the system across multiple generations. While I had completely anticipated that the clicks of the first section would always sound fairly similar, I had not at all expected that the following sections would also tend towards a recognizable sonic profile. Although I am satisfied with the extent to which the distinct timbral qualities of each section lend clarity to not only the overall progress of a given piece, but also the generative structures of the system as a whole, I'm still curious about how the sonic palette of a program like *vorbei* might be expanded.

Sonic expectations aside, the biggest question I had throughout my work on this project was how closely a listener would be able to follow the relationships between

specific pieces of data. Once I had dissolved both the sonic material and the compositional structures as much as I could, how clear would the progress of and relationship between any two sounds generated by the program be? Admittedly, even after nearly six months of constant testing and listening, I can't reasonably claim that I'm able to hear the relationship between the temporal structure of a series of clicks in section one, and the spectral structure of a sound produced in section four. In this respect I seem to have arrived at the same wall found by the serialist composers of the early 20th century such as Arnold Schoenberg, Anton Webern, and Alban Berg, whose tone rows became equally obscured when developed over the course of large musical forms. However, the emergence of recognizable timbral and temporal structures within phrases is, to my ear, quite clear. Additionally, phrases tend to distinguish themselves by their level of activity, such that over the course of a given piece, I can eventually keep track of progress through the overall gesture.

In its best moments, I've found that *vorbei* is capable of generating sound in a way that plays around the border of my ability to follow a piece of music. While there have been a number of instances in which the program arrives at and maintains a recognizable rhythm, timbre, or melodic contour in one phrase which is then dissolved, exploded, or smeared across other phrases, there are equally as many examples of moments when data and the probabilities controlling them assemble within a phrase in such a way that its output remains completely unpredictable. To me, the most rewarding listening experiences generated by *vorbei* contain both of these types of phrases. When this is the case, I can track the progress of a given gesture as an alternation between grounding moments of consistency and stretches of completely unpredictable behavior. As such, the structure of pieces which manage to accomplish this are built around my ability to conceptualize and organize my reception of what I'm hearing.

## Conclusion

In terms of Galloway's concept of new media serving as containers for old media, *vorbei* creates its own containers for its own media. Instead of using the computer as a container that can re-arrange a predetermined set of contents (be they fixed audio tracks, musical notes, or compositional techniques), I ended up with a system of logic by which the computer is able to recognize, respond to, and extrapolate from trends in data originating from an internally-generated random seed. That this logic is pointed towards a pair of speakers is the result of my background as a composer, but the general concepts shaping its interaction with these speakers could easily be adapted to any other medium. Though *vorbei* only ever produces old media, the only thing it contains is logic.

## Works Cited


Bown, Oliver and Sam Britton. 2013. *Methods for the Flexible Parametrisation of Musical Material in Ableton Live.* Paper presented at the Artificial Intelligence and Interactive Digital Entertainment Conference (AIIDE), Boston, 24-26.

Eigenfeldt, Arne and Philippe Pasquier. *Considering vertical and horizontal context in corpus-based generative electronic dance music.* Proceedings of the fourth international conference on computational creativity. Vol. 72. 2013.

Eno, Brian and Peter Chilvers. *Bloom*. 2012. iPhone application.

Eno, Brian and Peter Chilvers. *Trope*. 2012. iPhone application.

Galanter, Philip. *What is Generative Art? Complexity Theory as a Context for Art Theory*, New York: New York University. Print

Galloway, Alexander R. *The Interface Effect*. Cambridge, UK: Polity, 2012. Print.

Gold, Gwylim. *Bronze.* 2009. iPhone application.

Ott, Joshue and Morgan Packard. *Thicket*. 2014. iPhone application.

# Appendix A.

# Research Paper

In his book *The Interface Effect*, Alexander Galloway explains that “a new medium is invented, and as such its role is as a container for a previous media format. So, film is invented at the tail end of the nineteenth century as a container for photography, music, and various theatrical formats” (2012, 31). When film was eventually understood as “a superb conciliation of the Rhythms of Space (the Plastic Arts) and the Rhythms of Time (music and Poetry)” by Ricciotto Canudo in his 1911 manifesto *The Birth of the Sixth Art* (in which film is actually identified as the seventh art, with dance as the sixth), it began to be seen as something that could offer a unique perspective unavailable to the media for which it had previously acted as a “container”. In their current and most popular iteration, computers appear to be the media container par excellence. With the assistance of a screen and pair of speakers (keyboard and mouse optional), we can watch a movie, read a book, make a phone call, write a book, play video games, compose music, and so on. However, given the seemingly limitless capabilities of these machines, it can be difficult to identify what they offer that is unavailable to other media formats. If film was set apart by its exclusive access to novel explorations of time and space in a two-dimensional image, what might allow computers to similarly distinguish themselves?

First, it’s important to acknowledge that the now ubiquitous desktop, laptop, or smartphone configurations are by no means the only forms in which computers can currently be observed. They may take the shape of a digital cable box, an electronic price tag at a supermarket, a wireless internet router, or any number of objects that perform computational tasks with or without any sort of outside interaction. Understandably, this wide variety of applications and appearances can make the task of describing these increasingly limitless machines a bit difficult, so it becomes useful to first draw a line between what a computer *can do* and what a computer *is*.

The first step towards making this distinction is to recognize that while you can listen to a podcast, capture video, or check the time on a computer, it is not a radio, camera, or clock despite its ability to perform each of these functions. Turning to Wikipedia, we’re told that a computer is “a general-purpose device that can be

programmed to carry out a set of arithmetic or logical operations automatically," while Merriam-Webster defines its as "one that computes; specifically: a programmable usually electronic device that can store, retrieve, and process data." A crucial part of this second definition is the phrase "usually electronic," which stands slightly at odds with the almost exclusively electronic forms in which we encounter computers today. However, observing such ancient artifacts as the Antikythera mechanism – the Greek clock currently considered to be the world's first analog computer – we can see that electricity is not necessarily a computational prerequisite. By reconstructing the relationships of planetary orbital paths with a series of proportionally related gears, the Antikythera mechanism was capable of predicting astronomical positions and eclipses. What's important to recognize here is the extent to which this machine represented a crystallization of the knowledge it was able to produce. Like a modern calculator, the mechanism didn't allow for anything outside of the scope of what was possible with pen and paper, but in the moment that its designer was able to mechanically abstract the knowledge needed to solve the given problem, and subsequently arrange these abstractions according to the appropriate logic, the object became a material manifestation of the thought process it replicated – it became a thought object.

When the first electronic general-purpose computer, the Electronic Numerical Integrator and Computer (ENIAC), was completed in the 1940's, it operated on the same basic principle. Before ENIAC could perform any of the countless mathematical operations it had been designed to implement, each program had to first be written out on paper before technicians could hard-wire the logic into its many switches and cables. Though the materials had changed, the circuits and switches of ENIAC performed essentially the same function as the gears of the Antikythera mechanism – they allowed for the mechanical representation of a thought process which, once constructed, constituted a material manifestation of the knowledge required to solve the given problem. The necessary condition for a machine to be considered a computer can thus be seen to have nothing to do with the specific materials used to implement the logical processes it recreates. Returning to the earlier question of drawing a line between what computers are and what computers can do, we can now see that they are defined by neither the materials that enable their functions nor the functions they perform. Instead, it is this ability of an object to replicate logic, to exist as a collection of materials that constitute a crystallization

of the knowledge required to carry out a given task, that allows us to call these machines computers.

It's important at this point to note that all of what a computer can do is made possible by these systems of logic. Given the current way in which we interact with the numerous files, folders, and drives of laptops, desktops, smartphones, tablets, and so on, its easy to misunderstand the extent to which a computer actually "contains" the things it's capable of representing. Indeed, it's tempting to think of these devices as something like the latest iteration of microfilm; as libraries of microscopic images and sounds for which various software interfaces act as virtual librarians, needing only to locate and magnify the stored information. No doubt these library-like reference structures are implemented in determining the locations of files in memory, but what is being recalled in these instances is not the materials (images, characters, sounds) themselves. Instead, the digital "librarian" is directed towards a set of instructions that first identify what materials will be used to represent the information contained in the file, and then guide interface components in arranging that information for display, be it on the screen or from the speakers. Importantly, any given set of rules can be applied to any of the material the computer is capable of managing. Enabled by what Galloway refers to as "a radical standardization at the atomic level" (*Laruelle: Against the Digital*, 2014, Ch. 5), a user could send instructions intended for pixel arrangement to the speakers, or represent a sound file as text simply by changing a few lines of code. In fact, even by doing something as simple as loading a song to a playlist, a user can observe the ability of these rules to be applied to a variety of representational materials, as the single audio file, when loaded to the appropriate media player, not only enables playback of the given piece, but also displays text listing the name of the song, artist, and album within the playlist window. Clearly then, there is an extent to which the systems of logic replicated in the code and circuitry of the computer are not strictly linked to the actual materials they manage.

The question then becomes, if both the materials enabling computational processes and the materials managed by computational processes are to some extent arbitrary, what can be considered the essential material of computation? The common thread in each of the examples discussed so far has been the creation of mechanisms capable of replicating logic. Whether by the gears of the Antikythera mechanism, the circuitry of ENIAC, or the code of modern digital computers, each of these machines arranges some form of inanimate material in a way that allows it to imitate a certain

process of thought. The essential component in a computational process is thus neither the material that enables it, nor the material that it produces, but the thought that it replicates. The essential material of computation is thought.

Of course, thought is nothing new, so the preceding observation doesn't quite resolve the earlier question of how computers might distinguish themselves in a way comparable to that which granted cinema art status in the early 20th century. What does set computers apart, however, is the fact that they allow thought to be engaged on a purely material basis. Providing a space within which we can replicate and store nearly infinite amounts of coded logic, computers allow us to implement these isolated instances of thought in the management of whatever material they are capable of processing. Not only this, but insofar as any given set of code can be made to interact with any other set, the computer enables an extremely efficient means of combining and recombining thoughts in a variety of ways. Finally, and perhaps most incredibly, modern computers allow us to create systems in which these thought objects themselves can be operated on; in which a given system of logic can be modified by a larger logical superstructure (something like an Antikythera mechanism mechanism, which not only contains the original mechanism, but can also rearrange its gears to realize other gear-based logic systems). With all this in mind, how might we conceive of an art practice that highlights this unique ability of the computer? If the materials by which it is constructed are more or less arbitrary conduits for the slightly more ephemeral material of thought it contains, and if each coded thought can be applied to an equally arbitrary set of audio-visual materials, how do we realize a specifically computational aesthetic?

Some examples of artworks that engage with this concept of thought as material are given by Christiane Paul in her essay *From Immateriality to Neomateriality: Art and the Conditions of Digital Materiality*. Offering the concept of "neomateriality" to describe works that capture "an objecthood that incorporates networked digital technologies and embeds, processes, and reflects back the data of humans and the environment, or reveals its own coded materiality and the way in which digital processes perceive and shape our world." (2015, 2), Paul conceives of an approach to digital material that reveals the conditions of its creation. All of the works described in the article deal, in one way or another, with computer vision, and given their use of computer generated representations of the real world, have more to say about perception than thought (the former leaning more towards the ways in which information is reconstructed than how it is looked at,

understood, and acted on from outside itself in the latter). However, one of the works presented – Ashely Zelinskie's *Reverse Abstraction* series – provides some interesting insight for the current discussion.

Described on Zelinskie's website as an attempt to bridge the gap between human and computer perception by "constructing traditional objects in dual forms: as the classical object and as the hexadecimal and binary codes that represent them" (2014), the objects of the *Reverse Abstraction* series can to some extent be seen as both enacting and displaying the thought that generated them. "If a computer were to read the code" Zelinskie continues "it would see the object you are seeing," (2014). Thus we are presented with both the object itself and the codified thought that enabled its reproduction by digital means, but ultimately the computer can be seen as a somewhat unnecessary component in the process. For one thing, despite the fact that it would have been impossible for a human to produce them as perfectly as a computer did, any of the objects could have been created without the aid of technology. Furthermore, the claim that a computer would be able to see the objects by reading the symbols on their surfaces overlooks the extent to which a computer would be able to read the symbols at all. That is, the process of a computer "reading" the code would necessarily involve a human typing each string of symbols into whatever software understood those symbols as commands directing it towards the representation of the given object. As the numbers into which computational processes are coded for the convenience of human interpretation make up the skin of an object made to be seen by humans, we find, in these works, an excellent example of the computer being used as a container for (or in this case a creator of) old media. Instead of using thought as material, the *Reverse Abstraction* series uses representations of thought which, in the end, require no computational component. If this representational approach is insufficient in establishing thought as the formative subject (the thing both on display and creating the display) of the piece, what methods might be more effective?

One practice that offers a solution is the field of generative art, which Philip Galanter defines in his 2003 essay *What Is Generative Art?* as "any art practice where the artist uses a system, such as a set of natural language rules, a computer program, a machine, or other procedural invention, which is set into motion with some degree of autonomy contributing to or resulting in a completed work of art." (4). "The key element in generative art," he goes on to say "is then the system to which the artist cedes partial or total subsequent control." (2003, 4). It's important to note that Galanter's definition of

generative art does not require, and in fact predates, modern electronic computers. Mozart's Musical Dice Game, the Jacquard loom, and Islamic textiles are three of the many examples he gives to demonstrate the fact that "generative art is uncoupled from any particular technology" (Galanter, 2003, 4). Already this conception of generative art shows promise, as it is founded not on any specific computational material or process, but instead on the existence of a system that to some extent determines the form of the resulting product. The thought process, first crystallized in a set of instructions, is carried out in the realization of the actual work.

Of course, the presence of a system does not necessarily entail the presence of thought. Hans Haacke's *Condensation Cube*, for example, is a piece of generative art that involves a thoughtless system. The piece, created in 1963, consists of an acrylic cube with a small amount of water pooled at the bottom. When the cube is placed in a room for an extended period of time, condensation forms on its transparent surfaces as a result of various states of light, temperature, air pressure, and other ambient conditions which create miniature weather systems within the sealed container. The condensation patterns which become the content of the work are thus generated by a natural system as opposed to a system of thought. Within the field of generative art then, we must find works that deal specifically with these latter types of systems.

The combinatorial drawings and sculptures of conceptual artist Sol Lewitt are an excellent example of works conceived as and created by systems of thought. In his *Paragraphs on Conceptual Art*, Lewitt explains that in his practice, "The idea becomes a machine that makes the art," (1976, 1). Recalling our earlier examinations of the ways in which ideas can be seen to manifest themselves in the materials of machines such as the Antikythera mechanism and ENIAC, Lewitt's concept of an idea becoming a machine seems perfect for the current discussion. One piece conceived along these lines is Lewitt's Boston Museum *Wall Drawing*, a work to be created according to the following instructions: "On a wall surface, any continuous stretch of wall, using a hard pencil, place fifty points at random. The points should be evenly distributed over the area of the wall. All of the points should be connected by straight lines." These instructions bear a striking similarity to code written for digital computers, which not only enumerates a list of tasks to be carried out, but also defines the set of materials to be used. In this and other pieces made by Lewitt in the same style, it's interesting to consider the extent to which the work exists in both the written instructions and the product they end up generating. In contrast

to Zelinskie's *Reverse Abstractions*, Lewitt's instructions do not attempt to replicate the object that is eventually produced. Where the *Reverse Abstractions* prioritize the mutual perceptibility of a predetermined form, Lewitt's *Wall Drawings* foreground thought as an active creative agent, as it plays a formative role in realizing a drawn product that cannot be inferred from the instructions guiding its creation.

Of course, the human component necessary in the completion of such works cannot be overlooked, and there is an interesting extent to which the indeterminate nature of some of the instructions ("place fifty points *at random*") grant the person realizing each drawing a certain amount of creative agency. However, keeping in mind the fact that computers are also capable of acting randomly, the *Wall Drawings* fulfill Lewitt's machinic priorities. Each piece thus exists as its own computational system, as the set of instructions is processed by an ultimately mechanical agent capable of manipulating the given set of materials.

Though the process realized in the *Wall Drawings* can certainly be considered an adequate template for the creation of specifically computational works, one issue that arises is the extent to which any of the very rich conceptual details of such pieces are perceivable to viewers. Indeed, this problem is something of a constant in generative art. In the case of Lewitt, insofar as the instructions result in the production of a single static object, it can be difficult to see the thought at work and distinguish the drawings from similar pieces created by other means. To some extent this problem could be solved by an exhibition of multiple realizations of a single *Wall Drawing*, which, by allowing viewers to see various iterations of the same set of instructions, would give a glimpse into the thought-system being activated in such works. Still, it's worth asking if there might be a way for all this to be done within the context of a single presentation? Instead of displaying a thought system across [as opposed to "between" which suggests an exclusion of the bookends] the instructions and the product, how can a work display both simultaneously?

A number of possible answers can be found in the work of Philip Galanter, who provided some of the reflections on generative art quoted earlier. In his *ga1 generative animations*, Galanter created a series of generative audiovisual projections that can be seen evolving in real time. One piece in the series, called *untitled (bars)*, generates a collection of lines, each of varying color and width, to be projected onto a wall. In the notes for the piece, Galanter explains that the color of each bar is determined by the position of

the three corners of a triangle rotating within a color wheel. The sound is then generated based on the color of each bar, with one set of pitches being associated with primary colors and another with secondary colors. Here, instead of the static lines of Lewitt's *Wall Drawings*, we see a piece of art actively developing in time. Of course, film and music do this as well, but with the aid of Galanter's notes, we can understand the extent to which a thought process is being realized by the piece itself, while at the same time watching the process enacted in front of us.

Still, given the fact that each of the *ga1 generative animations* is presented as a 60-minute looping video, it can be difficult to distinguish them from any other piece of abstract video art. Their containment within not only the frame of a projected image, but also the fixed sequence in which images appear compromises the extent to which they can be viewed as being shaped in real time by an active process of thought. Galanter's 2013 installation *xepa* – featuring a collection of light sculptures, each designed to observe the state of all the other sculptures in the room and move towards a common audiovisual theme – allows viewers to see a sort of collaboration of and communication between a variety of enacted thought processes. Here, to the extent that we can observe and understand that each sculpture is independently moving towards the state of other sculptures, the sense of containment that points viewers of the *ga1 generative animations* back to film is slightly mitigated. Of course, while an installation functioning in this way would conceivably never loop, in order to be absolutely certain that none of the light sequences were pre-recorded, the viewer would have to remain with the installation for its entire existence. How then, could a piece of art make a viewer certain that what they are witnessing is a completely novel deployment of a given set of materials based on the real-time implementation of a system of thought?

To some extent, the problem with works like Galanter's is that they are displayed in spaces that viewers must travel to and eventually exit. As such, any given person's experience is limited to the window of time they're able to spend with a piece, which, at the very most, is the operating hours of the space in which it's presented. Ideally, for a work like *xepa*, the absence of exact repetition, coupled with an understanding of the interaction between materials, would allow viewers to infer that there is an active thought process being played out on the spot. Still, since they have no way of knowing at what point in the lifespan of the piece they are entering and exiting, it can be difficult to determine whether what's being seen is actually developing in the moment or simply a

segment of a recording. Fortunately, given the now ubiquitous presence of portable devices such as laptops, smartphones, and tablets, there are many alternatives to the gallery presentation model that allow viewers far greater agency in the experience of a piece.

Brian Eno and Peter Chilvers' generative music applications *Trope* and *Bloom*, distributed for use on smart phones and tablets, provide a platform within which users [note the shift from viewers to users] can either listen to or participate in the creation of an audiovisual piece being generated in real time. While the variety of interactive elements of each application is certainly interesting, perhaps the most important aspect for the current discussion is the user's ability to start and stop any given run of the software at will. As such, what sets these applications apart from installations like *ga1 generative animations* and *xepa*, is that the viewer can very quickly see that any two iterations of the piece are different in corresponding temporal points of their generation ["generation of iteration" being substituted for "existence of the piece", which now becomes difficult to define given the extent to which both a single run of the software and the software itself can be considered components of a given work]. That is, when the user starts the app, lets it run for 5 seconds, closes it and starts it again, they are given a different arrangement of material in what they know to be the same position of the piece's progress.

It's also important to note that in both *Trope and Bloom*, the materials remain quite consistent across iterations. Though such features as color palette and tuning might change slightly from run to run, the same (fairly small) bank of shapes and instruments is deployed each time. As such, it becomes increasingly clear, as users continue to stop and restart each application, that the active component in the work is a process of thought that is actively arranging [as opposed to "rearranging", which suggests an original arrangement that is deviated from] a fixed set of materials.

A number of examples of generative artwork distributed as software applications now exist. *Thicket,* by Joshue Ott and Morgan Packard, expands on the idea of pieces like *Trope* and *Bloom* by offering a variety of aesthetic modes, each of which offers a unique audiovisual experience. Within each mode, the style remains consistent enough to communicate that the same set of materials are being acted on in real time. In contrast to the sort of blank canvas approach found in the visual component of Eno and Chilver's work, each mode in *Thicket* presents a contained visual object that has its own unique

behavior when left untouched by the user. Each object in *Thicket* thus reveals the thought process employed in its creation and activation, and gives users the sense that they are working with an active realization of these underlying thought systems instead of providing input to the static open containers of applications like *Trope* and *Bloom*.

The work of Australian software artist LIA also lends some interesting insight towards possibilities for application-based art. Originally released as an interactive website in 1999, her piece *Re-move* "features algorithmic compositions that endlessly auto-generate into abstract patterns of lines, circles, and waves," (Brucker-Cohen, 2015, 1). As LIA explains in her notes on the work "The aim was…to play with code itself…simply put: exchange a plus with a minus in a formula that you don't understand and a) something interesting might happen and b) you might learn how the formula actually works." Already we can see a desire to work with codified thought, as the piece is described as organizing itself around creative manipulations of formulas. What is perhaps most interesting about the compositions of the *Re-move* series is the ways in which each deals with erasure. For example, the piece labeled *10* generates a cluster of shapes that arrange themselves in response to the user's movement of the mouse, but instead of displaying only its current position, each shape leaves a trail of its past locations. Similarly, the piece *05* presents a grid of shapes that are repelled by the mouse, but which move back towards their original position when the mouse moves away. Again, each shape leaves a trail of its motion and thus contributes to a sort of visual sediment that documents not only the thought process employed in designing the motive character of each object, but also the thought process of the user as they interact with and develop an understanding of the piece.

In each of the many preceding examples of generative artworks based on systems of thought, we can begin to see a variety of possibilities for what might constitute a specifically computational practice. Of course, as noted earlier, thought is by no means a material that is unique to computational art, and indeed it's not difficult to see that in fact every art form uses thought as material in some way or another [with theatre, film, music, and other time-based practices having the capability of developing and changing the underlying thoughts guiding a given piece over time]. However, it is this ability of machines such as the Antikythera mechanism, ENIAC, or any of the multitude of modern personal computation devices to crystallize the thought systems they implement that sets them apart from other mediums of artistic expression. By enabling the creation of a sort of fixed thought object that can manipulate and interact with any other similarly constructed object,

computers (in the many forms they're capable of taking) establish themselves as platforms for a unique art practice outside of their (no less useful) role as containers of old media.

## Works Cited


Attali, Jacques. *Noise: The Political Economy of Music.* Minneapolis: U of Minnesota, 1985. Print

Bown, Oliver, Benjamin Carey and Arne Eigenfeldt (2015, August). *Manifesto for a Musebot Ensemble: A platform for live interactive performance between multiple autonomous musical agents*. Paper presented at the International Symposium of Electronic Arts, Vancouver, BC.

Bown, Oliver, Arne Eigenfeldt, Aengus Martin, Philippe Pasquier (2013, November). *Towards a Taxonomy of Musical Metacreation: Reflections on the First Musical Metacreation Weekend*. Paper presented at the Conference on Artificial Intelligence and Interactive Digital Entertainment, Santa Cruz, CA.

Brucker-Cohen, Jonah. "Art in Your Pocket 4: Net Art and Abstraction for the Small Screen." *Rhizome*. N.p., 2 Aug. 2015. Web. 06 Dec. 2015.

Canudo, Ricciotto. " Birth of a Sixth Art" [1911]. In Abel, Richard, ed. *French Film Theory and Criticism.* Princeton U P, 1988. 58-65.

Dyson, Frances. "The Genealogy of the Radio Voice." *Radio Rethink: Art, Sound and Transmission*. By Daina Augaitis and Dan Lander. Banff, Alta., Canada: Walter Phillips Gallery, 1994. N. pag. Print.

Eigenfeldt, Arne (2015, August). *A Composer's Search for Creativity Within Computational Style Modeling* Paper presented at the International Symposium of Electronic Arts, Vancouver, BC.

Eno, Brian and Peter Chilvers. *Bloom*. 2012. iPhone application.

Eno, Brian and Peter Chilvers. *Trope*. 2012. iPhone application.

Galanter, Philip. *ga1 generative animations*. 1999. Video.

Galanter, Philip. *What is Generative Art? Complexity Theory as a Context for Art Theory*, New York: New York University. Print

Galanter, Philip. *Xepa*. 2013. Industrial led lighting, microcontrollers, custom circuit boards, power supplies, acrylic plastic, audio speakers, software.

Galloway, Alexander R. *The Interface Effect*. Cambridge, UK: Polity, 2012. Print.

Galloway, Alexander R. *Laruelle: Against the Digital*. N.p.: n.p., n.d. Kindle File.

Hutchison, Steph, John McCormick, Jordan Beth Vincent, and Kim Vincs. (2015, August) *Emergent Behavior: Learning from An Artificially Intelligent Performing Software Agent* Paper presented at the International Symposium of Electronic Arts, Vancouver, BC.

Koenig, Gottfried Michael. "Aesthetic Integration of Computer-Composed Scores." *Computer Music Journal* 7.4 (1983): 27. Web.

LeWitt, Sol. "Paragraphs on Conceptual Art." *Artforum* (1967): 79-83. Web.

Lewitt, Sol. *Wall Drawing*. 1971. Pencil. Museum of Fine Arts, Boston.

LIA. "Re-move by Lia." *Re-move by Lia*. N.p., 1999. Web. 06 Dec. 2015.

Manovich, L. "Media After Software." *Journal of Visual Culture* 12.1 (2013): 30-37. Web.

Merz, Evan X. (2015, August). *Musical Structure Imitation using Segmentation, and k-Nearest Neighbors (kNN).* Paper presented at the International Symposium of Electronic Arts, Vancouver, BC.

Obermeier, Philipp, Sarah Opolka, and Torsten Schaub (2015, August). *Automatic Genre-Dependent Composition using Answer Set Programming.* Paper presented at the International Symposium of Electronic Arts, Vancouver, BC.

Ott, Joshue and Morgan Packard. *Thicket*. 2014. iPhone application.

Paul, Christiane (2015, August). *From Immateriality to Neomateriality: Art and the Conditions of Digital Materiality.* Paper presented at the International Symposium of Electronic Arts, Vancouver, BC.

Roads, Curtis, and John Strawn. *The Computer Music Tutorial*. Cambridge, MA: MIT, 1996. Print.

Turing, A. M. "I.—Computing Machinery And Intelligence." *Mind* LIX.236 (1950): 433-60. Web.

Zelinskie, Ashley. *Reverse Abstraction*. 2014. Plastic.

Zelinskie, Ashley. "Reverse Abstraction." *Ashley Zelinskie*. N.p., 2014. Web. 07 Dec. 2015.

Zielinski, Siegfried. *Deep Time of the Media: Toward an Archaeology of Hearing and Seeing by Technical Means*. Cambridge, MA: MIT, 2006. Kindle File.

# Appendix B.

# *vorbei:* A Generative Music System

**Paul Paroczai**
School for the Contemporary Arts
Simon Fraser University
pparocza@sfu.ca

**Arne Eigenfeldt**
School for the Contemporary Arts
Simon Fraser University
arne_e@sfu.ca

**Abstract**

Despite the many examples of computer programs currently capable of managing some aspect of music composition, systems that are given complete autonomy in the creation of a piece remain fairly rare. A program that generates harmonic progressions with flawless adherence to traditional rules of voice-leading may have no sense of how to conceive of individual chords or progressions within a larger-scale structure, while another program capable of the latter may be less suited to manage its more specific details. *Vorbei* is a program designed to generate a unique and complete piece of music every time it is run, and to do so with no external interaction. Each piece generated by *vorbei* can be understood as a series of variations derived from a gesture generated at the beginning of a given run.

## Introduction

Algorithmic music has a long history, with examples reaching at least as far back as the musical dice games of the Eighteenth century (Hedges 1978), and extending through numerous compositional practices, including the serialist techniques of composers such as Schoenberg, Webern and Berg in the early 20th century and Steve Reich's experiments with process in the 1960s and 70s. Current examples relevant to the project being discussed include Brian Eno's *Bloom* and *Trope* (2012), Joshue Ott and Morgan Packard's *Thicket* (2014), and Icarus' (Oliver Bown and Sam Britton) *Fake FIsh Distribution* (2012). Musical metacreation is a contemporary exploration and evolution of algorithmic music in which computational systems are designed to contribute to the creation of a fully finished artwork (Pasquier et al. 2016).

Given Galanter's definition of generative art as "any art practice where the artist uses a system...which is set into motion with some degree of autonomy contributing to or resulting in a completed work of art" (Galanter 2003) human interaction does not necessarily need to be excluded from such works. As a result, many MuMe systems have included a human, either as a performer, interacting with the system, or "nudging" the system along in response to its output. Few systems generate entire and complete


The Fourth International Workshop on Musical Metacreation, MUME 2016.
www.musicalmetacreation.org.

musical compositions. *Vorbei* generates a structure, but also uses these structural elements as sonic material, in effect, sonifying its own structure.

A given run of *vorbei* always begins with the generation of phrase durations, the successive combination of each is considered a gesture. The ratios of phrases durations to one another becomes an integral component of *vorbei*'s generation. Throughout the piece, in order to reinforce emerging trends and push the program towards arriving at coherent structures, generated data is stored and analyzed to create probabilities which determine its later use. Each sound heard in the piece is derived from manipulation and analysis of data used to generate preceding sounds. Since initial sounds contain temporal spacings derived from an analysis of the initial phrase structure itself, every sound generated in *vorbei* can, in one way or another, be traced back to the initial gesture. As such, individual runs of *vorbei* can be understood as a series of variations generated from increasingly extended derivations of the initial series of values.

## Description

*vorbei* is a fully generative work, in that there is no opportunity for interaction by the listener. Once launched, a minimal user interface is presented (see Figure 1). The "start" button initiates the composition and realtime performance.


***Figure 1.*** *The vorbei user interface.*

Each composition consists of four major structural components: sections, gestures, phrases, and events. A gesture is a series of phrases, while events take place within individual phrases. Phrases progress independently through four sections (see Figure 2).

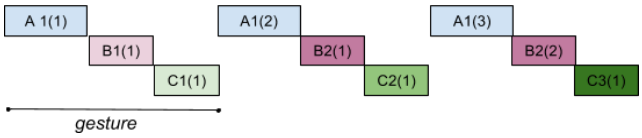


***Fig. 2.*** *Example vorbei generated structure, displaying three gestures. Three phrases – A, B, C – have been created, each of different duration. Phrase A has undergone three cycles (shown by parentheses), and is still in section 1; Phrase B performed one cycle of section 1, then two cycles of section 2; Phrase C has performed one cycle in sections 1, 2, and 3.*

A critical aspect of *vorbei* is the conception that the structure is not only audible, but actually sonified. Throughout its progressive structure, *vorbei* generates data, then analyses this data to determine statistical information, such as overall mean, and differences between individual datum and the mean. Probabilities for action are often calculated based on these differences, by generating random values and comparing them to the difference and/or mean.

Each section will be described: first in terms of its structure; then in terms of how the structure is made audible.

## Section One: Structure

The composition begins with the generation of phrase durations, each of which is a randomly selected value between 50 and 12,000 milliseconds (the former being the amount of time needed between two sounds for them to be considered separate, and the latter the longest possible value in traditional musical notation – a double whole note at 40 beats per minute). Every time a new duration is generated, it is compared to previous durations in an effort to discover trends in similarity. By comparing the average difference between existing values, a new value that is significantly different than the current average will cause the generation of duration values to end (see Table 1). Termination can be caused by a significantly different value (i.e. phrase C in Table 1), but also a value that is very similar to a previous value that followed significantly differing values. The total number of phrases that have been created is considered a *gesture*.

**Table 1.** Three phrases (A, B, C) with independent durations, resulting in a gesture of ten seconds. The subdivisions are the ratio of individual phrase durations relative to the gesture's total duration.

| Phrase | Duration (ms) | Ratio |
|---|---|---|
| A | 4500 | 0.45 |
| B | 5000 | 0.5 |
| C | 500 | 0.05 |
| (total) | 10000 | 1.0 |

Phrase durations remain constant throughout the composition. Since phrases are presented sequentially, resulting gesture durations are also constant. Each phrase progresses independently through a series of four sections, each of which is associated with a specific sound-generating technique. Phrases remain in a given section until a specific condition for the end of the section is met. Repetitions of a section are referred to as cycles.

The ratio of every phrase's duration to the entire gesture's duration (see Table 1, column 3) is calculated, and all possible combinations of ratios are calculated and stored (see Table 2).

**Table 2.** All possible combinations of ratios from Table 1.

| | | | | |
|---|---|---|---|---|
| .45 .5 .05 | .95 .05 | .45 .55 | .5 .5 | 1.0 |

At the beginning of a phrase's cycle, the phrase duration is subdivided by a random selection from this ratio list. With each subsequent cycle, these subdivisions are then further subdivided, with the same method of random selection (see Figure 3).

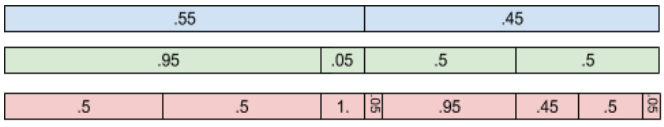


***Figure 3.*** *A phrase after three cycles: cycle1 (blue) with two subdivisions; cycle 2 (green) with two further subdivisions; cycle 3 (red) with eight subdivisions.*

## Section One: Content

The first section's structure is audibly delineated by clicks at the beginning of each subdivision. Each click is given a random amplitude value between -1 and 1, and randomly sent to either the left or right channel. Each click is also stored in a wavetable as a single sample; the size of the wavetable corresponds to the duration of the phrase. The position of each sample in the table corresponds to the time at which the click occurred. Each cycle of a phrase generates a new stored wavetable.

**Ending Section 1**

At the end of each cycle, a check is made to determine the number of subdivisions within the phrase with a duration of less than 50 ms. This sum is divided by the actual number of subdivisions in that phrase's cycle; if this result is greater than a generated random value between 0 and 1, the section is considered complete for this phrase. As the subdivisions are further subdivided with each subsequent cycle, the probability of progressing to the next section naturally increases.

## Section 2: Structure

In Section 2, data generated in Section 1 is sonified via wavetable synthesis. Individual events are determined by three main parameters: duration, frequency, and waveform.

**Duration**

Subdivision ratios generated in Section 1 are used to generate event durations in Section 2, using a roulette wheel selection (see Table 3). Because these selections are made sequentially, events can exceed their phrase's duration: for example, a selection of .5 followed by .55. Once selected, the ratios are multiplied by the phrase's duration (i.e. 4500 ms for phrase 1 in Table 1) to determine the event's duration.

***Table 3.*** *Probabilities for event durations, based upon ratios derived from example subdivisions shown in Figure 3*

| Ratio | # of occurences | Probability |
|---|---|---|
| .05 | 3 | .214 |
| .45 | 2 | .143 |
| .5 | 5 | .333 |
| .95 | 2 | .143 |
| 1. | 1 | .071 |

## Section 2: Content

All events are not automatically sonified in Section 2; instead, each event's performance is dependent upon a calculated density probability, a value derived from the data generated in Section 1. The final number of subdivisions of a phrase (e.g. in Figure 3, the third phrase produced 8 subdivisions) is divided by the number of phrases in the gesture (e.g. 3 in Table 1) raised by the number of cycles achieved by the phrase (e.g. 3 in Figure 3); in our example, this is 8 divided by 9, resulting in a probability of .888 for any event to be performed in Section 2.

**Frequency**

Subdivisions for each phrase from Section 1 were stored as individual samples in wavetables; in the second section, these wavetables are read at audio rates for events (if they pass the density test, described above). The frequency of the wavetable is based on a roulette wheel selection from the subdivision ratios, displayed in Table 3, using the following formula:

(1000) ÷ ((selected ratio * 50))

**Waveforms**
Waveforms for individual events within phrases in Section 2 are selected randomly from the wavetables stored for the given phrase during Section 1.

**Similarity and Identity**
The first event generated by a phrase in Section 2 has its parameter data – the duration, frequency, and waveform data shown in Table 4 – stored regardless of whether or not it is actually heard. Subsequent events in the phrase, as well as in following cycles, are compared to the event data, and similarity values are continually calculated. Space does not permit describing these calculations in detail, other than to note that this continually recalculated information is stored with the event data (see Table 4).

Additionally, a *similarity identity* for the phrase is continuously calculated. Each new event's parameter data – i.e. duration, frequency, waveform – is compared to the current phrase mean for each parameter: the difference between the mean of similarity values before and after the most recent storage of event data is considered the phrase's similarity identity. Like most data in *vorbei*, these values become a resource for events generated in later sections.

***Table 4.*** *Probabilities for event durations, based upon ratios derived from example subdivisions shown in Figure 3*

| Ratio | Duration | Ratio | Frequency | Waveform | Similarity values | | |
|---|---|---|---|---|---|---|---|
| | | | | | Duration | Frequency | Waveform |
| .45 | 2025 | .05 | 400 | 1 | .5 | .98 | .66 |
| .5 | 2250 | .55 | 36.4 | 2 | .0 | .96 | .33 |
| .45 | 2025 | .45 | 44.4 | 1 | .5 | .96 | .66 |
| | | | | Average Similarity | .33 | .96 | .55 |
| | | | | Similarity identity | .952 | | |

**Ending Section 2**
The inverse of the similarity identity for a phrase determines the probability of progressing to the next section. To clarify, the similarity identity is not a similarity measure between instances of a parameter, but a value that represents the current parameter state of the phrase; by comparing this value at the beginning and end of the cycle, parameter convergence is determined.

## Section 3: Structure
Event durations in Section 3 are generated exactly as they are in Section 2: using ratios derived from Section 1.

## Section 3: Content
Section 3 generates audio using the same wavetable synthesis techniques described above. Individual events continue to be created, and stored, based upon selection from data generated in Section 2. Criteria for selection from the duration, frequency, and waveform data shown in Table 4 is based upon similarity ratings. Individual similarity values of each stored parameter are compared to the average similarity of the

given parameter: the difference between these two values is inversely proportional to the probability that the parameter value will be selected as an event in Section 3. Thus, given the data shown in Table 4, the most likely selection will be a duration of 2025 (its similarity value of .5 is closest to the mean of .33); a frequency of either 36.4 or 44.4 (their similarity values of .96 are equal to the mean of .96); wavetable #1 (its similarity value of .66 is closest to the mean of .55).

Every event undergoes spectral analysis and potential signal processing. The specific type of each is dependent upon how many times (rounds) the event has occurred. Each round generates data that is used in later rounds, as well as following sections.

**Round 1**
The spectral centroid is tracked for each event, and all discrete frequencies are stored for every event. No actual signal processing occurs on the audio.

**Round 2**
Wavetables are passed through a 30-band bandpass filter, whose frequencies are logarithmically distributed between 20 Hz and 20 kHz (i.e. 1/3 band per octave). Bands are only active if they contain spectral centroid frequencies from Round 1.

Additional analysis is done on each active band, beginning with the spectral centroid within that band. As this centroid is most likely moving during the event's duration, all centroid frequencies are stored, including the duration that each centroid maintained. Finally, the centroids with the two longest durations are chosen as a subband for each active band in the event, and stored with the event.

**Round 3**
Wavetables are now passed through the subband filters, effectively increasing the filter's slope. The output is then analysed to determine the lowest and highest frequencies present within these subbands, and this data is then stored with the event (see Table 5).

Once an event has had its round 3 subbands calculated, a new event type is introduced for that phrase: *Click-2*. The audio for this event-type – played concurrently with the wavetable synthesiser – is comprised of clicks played through a filter whose parameters are derived from this round's spectral analysis data. *Click-2* onsets are determined in the same way as event onsets for the wavetable synthesiser, albeit independently.

***Table 5.*** *Subband creation in Section 3. Round 1 determines the active ⅓ octave bands based upon the spectral centroids of the entire event; Round 2 determines the subbands within each active band based upon the spectral centroids within each band; Round 3 tightens the bandwidth of the subbands; subsequent events are played through these subbands.*

| Ratio | Frequency Band (heard) | Example centroid frequencies | Subband | Centroid |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| Round 1 | all | 100 450 900 1300 | | |
| Round 2 | 350-710 | 410 625 675 | 410-675 | 320 250 700 |
| Round 3 | 410-675 | 495 525 555 | 495-555 | … |
| … | 495-555 | … | … | … |

**Ending Section 3**

Phrases continue to cycle, with individual events in the phrase at different stages in spectral processing (rounds). The greater number of events that have been through Round 3, the higher the probability that the phrase will progress to Section 4.

## Section 4

Section 4 introduces additional audio processes using existing data. Event data from Section 3 – filter frequency centroids, bandwidths, and centroid durations – are selected using a roulette-wheel selection for use by the wavetable synthesiser and *Click-2*, previously used in Section 3.

In addition to *Click-2* and the wavetable synthesizer, Section 4 introduces four new events: *Sines*, *Counter-Synth*, *Counter-Click*, and *Counter-Sines*.

**Sines**

The wavetable synthesiser output is analysed for its overall spectral centroid, including tracking the duration of each centroid frequency. These durations are then divided by the total time of the event to determine a "frequency-rhythm" (see Table 6).

***Table 6.*** *Example frequency-rhythm pairs for Phrase 1's 4500 ms duration*

| Frequency | Duration | Ratio |
|---|---|---|
| 1235 | 350 | .077 |
| 350 | 250 | .055 |
| 975 | 700 | .155 |

*Sines* events are comprised of sine waves, whose frequencies are selected from the frequency-rhythm data (Table 6). Durations are selected from the event parameter data (Table 4) multiplied by the frequency-rhythm ratio. Onset location within a phrase is randomly selected within the phrase, less *Sines'* duration.

**Counter-Events**

Counter-events are introduced so as to add additional voices to *vorbei*, using existing data in related, but subtly different methods. Prior to their introduction, generation of audio data can be considered as

organising, recognising, and accentuating trends in random sources. Counter-events generate new material, albeit derived from the pool of collected analysis data.

For example, probabilities for any playing voice use an existing event's stored probability, but adjust it based on the event's similarity rating. Counter-events create their own data, adjusted from existing event parameter data. For example, a counter-event's duration will similarly begin with the event duration, but adjusted by the event's similarity. New analysis data includes calculating frequency roughness between all stored events, adjusted by the similarity value for each event.

Each of the three *Counter-Events* generate and use adjusted data independently.

*Counter-Synth* uses adjusted duration values to determine the duration of individual events. Each frequency in a given adjusted frequency list generated by *Counter-Synth* is used as both the center frequency and Q of a single resonant filter in a series of filters through which noise is sent to generate the sound of the event.

*Counter-Click* uses adjusted duration values to determine the amount of time between successive clicks. Each click is filtered using the same technique as described for the *Counter-Synth*.

**Ending Section 4 (and the composition)**

As soon as any phrase completes a cycle of Section 4, the probability that the entire piece will end is calculated. Consistent with earlier analysis, the most recent event data is compared to previous event data in the section: the more often any given frequency or duration value for one event is found to be identical to related values in other events, the higher the probability that the piece will end.

## Conclusion

We have presented a musically metacreative system that generates entire compositions, in which the structure itself is used to determine all aspects of following audio events. Furthermore, every generated event's parameter data is stored, and is potentially used in later stages of the program. Figure 4 presents a graph outlining the data flow through the composition, including when data is generated, and how it is reused.

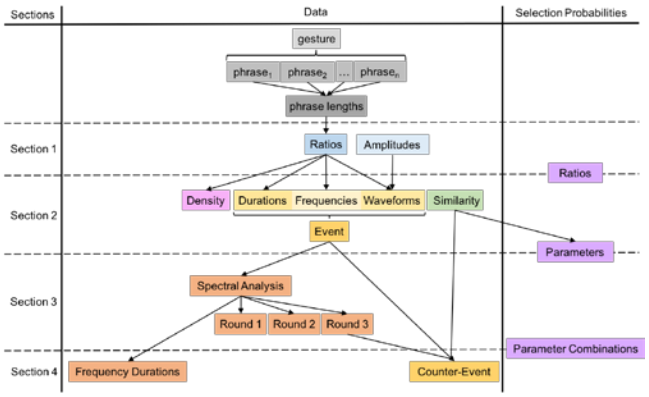


***Fig. 4*** *Data flow and(re-)usage in* vorbei.

Though the relationships of and continuities between specific pieces of data are likely impossible to perceive – for example, hearing the relationship between a click at second 5 of a 10 second phrase and a wavetable in the same phrase being read through at 40 Hz – the continual grouping of parameters and reinforcement of trends via probability tables tends to result in the emergence of recognizable timbral and temporal structures within phrases. Phrases are often also identifiable by their level of activity. Additionally, the distinct timbral qualities of each section lend clarity to the overall progress of a given piece, and the generative structures of the program as a whole.

As *vorbei* is completely new, we have not had the opportunity to evaluate whether listeners can perceive the relationship between structure and audio, or consistencies and differences between multiple generations. We hope to pursue such evaluations in the coming months.

## Acknowledgements

The authors wish to acknowledge the support of a grant from the Social Sciences and Humanities Research Council of Canada, and the School for the Contemporary Arts, Simon Fraser University.

## References

[1] Galanter, P. 2003. What is Generative Art? Complexity theory as a context for art theory. GA, Milan.

[2] Hedges, S. 1978. Dice Music in the Eighteenth Century. Music & Letters 59:2, 180–187.

[3] Pasquier, P., Eigenfeldt, A., Bown, O., & Dubnov, S. 2016. An Introduction to Musical Metacreation. Computers in Entertainment (CIE), 14:2, 2.