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 of algorithmic music generated by and distributed on computers 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).

The Fifth International Workshop on Musical Metacreation, MUME 2017.
www.musicalmetacreation.org.

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 (indeed all documentation of works presented at past MuMe workshops feature such interactive systems). Few systems generate entire and complete 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 ratio of phrase 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.

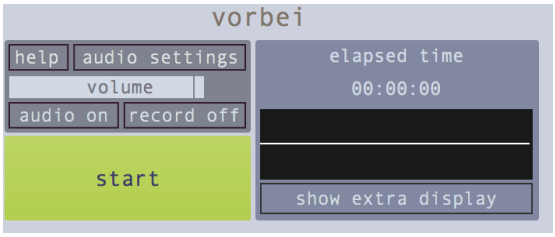

Fig. 1. *The vorbei user interface.*

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).

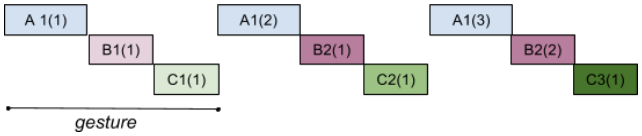

Fig. 2. *Example vorbei generated structure, displaying three gestures. Three phrases – A, B, C – have been created, each of different duration. Phrase A has undergone three cycles (shown by parentheses), and is still in section 1; Phrase B performed one cycle of section 1, then two cycles of section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

Importantly, while this general strategy of deriving new material by measuring difference in initial random seeds was based on a generalization of the composer's personal understanding of effective methods for managing musical structures and forms, following the establishment of this broad conceptual framework, all decisions made in the design of the program prioritized enabling the system to logically reinforce trends emerging in its own behavior. That is, to the extent that the composer was able to avoid direct impositions of his personal artistic preferences, *vorbei* was designed to be as self-referential in its decision-making as possible.

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

Section 1: Structure

The composition begins with the generation of phrase durations, each of which is a randomly selected value between 50 and 12,000 milliseconds. Though the determination of this range was ultimately a heuristic decision made to provide necessary limits on duration, the specific values were chosen based on the former being the amount of time needed between two sounds for them to be considered separate, and the latter the longest possible single value in traditional musical notation – a double whole note at the slowest standard metronome marking of 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).

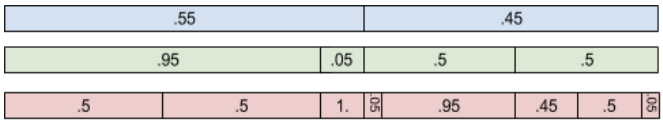

Figure 3. *A phrase after three cycles: cycle1 (blue) with two subdivisions; cycle 2 (green) with two further subdivisions; cycle 3 (red) with eight subdivisions.*

Section 1: 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.

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

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

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.

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).

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		

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.

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.

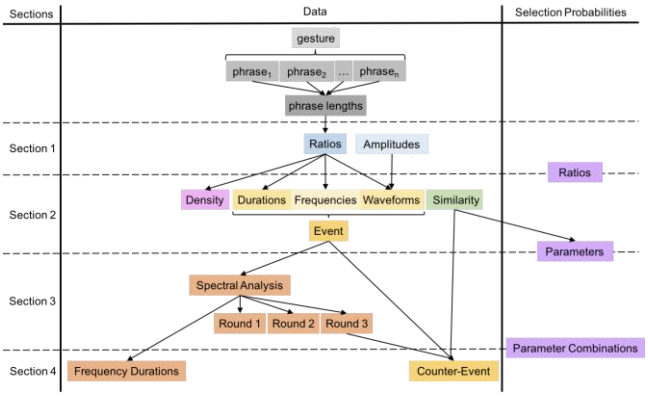

Fig. 4 *Data flow and (re-)use in* vorbei.

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*.

*Selected recordings of pieces generated by *vorbei* can be found at https://vorbei.bandcamp.com/releases.

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.