Digital adaptions of the scores for Cage Variations I, II and III

Lindsay Vickery
Edith Cowan University

Catherine Hope
Edith Cowan University

Stuart James
Edith Cowan University
Digital adaptations of the scores for Cage Variations I, II and III
Lindsay Vickery, Cat Hope, and Stuart James
Western Australian Academy of Performing Arts, Edith Cowan University

ABSTRACT

Western Australian new music ensemble Decibel have devised a software-based tool for creating realizations of the score for John Cage's Variations I and II. In these works Cage had used multiple transparent plastic sheets with various forms of graphical notation, that were capable of independent positioning in respect to one another, to create specifications for the multiple unique instantiation of these works. The digital versions allow for real-time generation of the specifications of each work, quasi-infinite exploration of diverse realizations of the works and transcription of the data created using Cage’s methodologies into proportionally notated scrolling graphical scores.

1. INTRODUCTION

John Cage's eight Variations (1958-67) occupy a unique position in the composer's output. By the late 1950s, Cage had made significant progress in exploring the use of indeterminate sound sources (such as radio and LP recordings), a range of chance procedures for generating notation and indeterminacy of notation. His attention now turned towards the indeterminacy and "flexibility" of formal structure itself: "a way to further the diversity and flexibility of his compositions by removing the fixity of the score itself" [28].

The eight Variations were the principal vehicle for the exploration of this idea, constituting nearly a quarter of his compositional output during this period. Following the completion of Variations VIII, the most open of the works in every respect, Cage returned, for the most part, to more traditional compositional outcomes marked by his exploration of the "recomposition" of pre-existing works.

<table>
<thead>
<tr>
<th>Score specification</th>
<th>sound sources</th>
<th>performance space</th>
</tr>
</thead>
<tbody>
<tr>
<td>I (1958)</td>
<td>quasi-determinate</td>
<td>unspecified</td>
</tr>
<tr>
<td>II (1961)</td>
<td>sound producing systems</td>
<td></td>
</tr>
<tr>
<td>III (1963)</td>
<td>indeterminate motion</td>
<td></td>
</tr>
<tr>
<td>IV (1963)</td>
<td>topographical map</td>
<td></td>
</tr>
<tr>
<td>V (1965)</td>
<td>astronomical chart</td>
<td>integrated</td>
</tr>
<tr>
<td>VI (1966)</td>
<td>sound system component diagram</td>
<td></td>
</tr>
<tr>
<td>VII (1966)</td>
<td>&quot;silence&quot;</td>
<td>&quot;real-time sounds&quot;</td>
</tr>
<tr>
<td>VIII (1967)</td>
<td></td>
<td>&quot;ambient sounds&quot;</td>
</tr>
</tbody>
</table>

Table 1: A summary of Variations I to VIII.

Over the ten years from 1958 to 1967, Cage revisited to the Variations series as a means of expanding his investigation not only of non-linear interaction with the score but also of instrumentation, sonic materials, the performance space and the environment. The works chart an evolution from the "personal" sound-world of the performer and the score, to a vision potentially embracing the totality of sound on a global scale. Table 1 gives a summary of the evolution of Cage’s approach to the score, sound sources and the performance space in the Variations series.

1 An early example is Oedipus in US (1942) [34].
2 These included the use of the I Ching as a source of aleatoricism in Music of Changes (1953) [29, pp. 78-88]. "Found systems" such as "folded paper templates" in Music for Carillon No. I (1952) [29, p. 92] and the "paper improvisation technique" in Music for Piano (1952-6) [29, p. 94]. Cage's use of Astronomical maps as "found systems" dates from Music for Carillon No. A (1961) [29, p. 211] and was incorporated in Variations I-V (1965).
3 Cage's exploration of indeterminate notation began in Music for Piano (1953) and culminated in 1958 with the magnum opus Concert for Piano and Orchestra, [29, p. 199], [33, p. 132].
4 Examples are recomposition of phrases of Satie's Socrate (1910) in Cheap Imitation (1969) [4]. "Subtraction" of material from anthems and congregational tunes Apartment House 1776 (1976) and "rubbing" of Satie Chorales in Song Book (song 85) [27].
There is relatively strong documentation of the evolving non-digital performance practice of the Variations as performed by David Tudor (Variations II-1961 [30], [31], [22]), John Cage (Variations II-1963 [28]), John Cage, Merce Cunningham et al (Variations V-1965[26], [19]), David Miller (Variations I and II-2003 [24]),...
A reading of the work is created by measuring the distance from each point to each of the five lines to generate a composite of parameters that define each event with the following attributes: number of sounds (1-4+), frequency, duration, amplitude, timbre and point of occurrence. These attributes are relative with the continuum upon which the parameter is measured defined by the performer. For example: the point of occurrence of each event is relative to the total duration of the work (which is not defined by Cage). Figure 2 illustrates the measurement process required to define one event [16].

This procedure results in a mixture of determinate, permutable and indeterminate variables in Variations I. The number and position of the points and lines is fixed and there is a finite number of possible combinations and orientations of the transparencies, however the range of the continuum upon which each parameter is plotted is indeterminate. Table 4 illustrates the determinate, permutable and indeterminate factors involved in the generation of an instantiation of the work.

Variations II uses a similar system of dots and points, with some small but significant differences. There are six transparencies each with a single line and five transparencies each with a single point. The sixth line determines the structure of the musical event, whether it is a single sound, an aggregate or a constellation of sounds, the function that had been determined by the size of the points in Variations I.

The orientation of the lines and points is therefore completely open, meaning that there are an infinite set of potential configurations of the score. A performance consists of any combination of configurations and therefore in theory Variations II may describe any possible musical work [24 p. 42]. In this sense it “represents the most flexible composition tool that Cage ever invented” [29 p.136].

Table 4: Determinate and indeterminate qualities of Variations I.

<table>
<thead>
<tr>
<th>Indeterminate</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency/Overtone/Amplitude Range</td>
<td>open</td>
</tr>
<tr>
<td>Instrumentation</td>
<td>open</td>
</tr>
<tr>
<td>Total Duration/Event occurrence</td>
<td>open</td>
</tr>
</tbody>
</table>

Performance of Variations I and II has traditionally involved one of three methods: “simply observing” [5] the resulting score, annotating an instantiation of the score [5][21] or transcribing the detailed measurements of an instantiation into a “performance score” [24 p. 22]. Figure 3 shows violinist Patricia Kopatchinskaja’s annotation of the score of Variations I [21].
The principal issue associated with “simply observing” or annotating the score, as can be observed in Figure 3, is that the notation on Cage’s transparencies is two-dimensional as opposed to traditional one-dimensional linear musical notation. To preserve the order of note occurrence, the transparencies must be read “two-dimensionally” in arcs emanating from the line that determines “point of occurrence” as shown in Figure 4. The distances to the other four lines and calculation of their parametrical value must occur simultaneously.

Although David Tudor’s realisation of Variations I relied on “careful definition of measurement scales and a precise performance score” [30 p. 2], James Pritchett shows that Tudor’s version of Variations II reduced Cage’s prescribed measurements to binary values: simple and complex. Figure 5 shows Tudor’s transcription of two events from the work. Tudor’s score overcomes the issue of reading multiple axes (the 50 events he used were aligned in rows), however its transformation of the multi-parametrical notation into single- or double-bordered squares with intersecting lines and circled or plain points is nearly as enigmatic looking as the original.

3. THE SCORE-READER FOR VARIATIONS I AND II

The imperative of generating performance materials that are easily and intuitively read, led Decibel to a decision to transcribe the data created in Variations I and II, into proportionally notated graphical scores. In Decibel’s realisations of Variations I and II the parametrical data derived from measuring perpendicular distances is evaluated and then used to generate a scrolling, proportionally notated screen-score. The score moves from right to left with the point of occurrence of each event, rendered as a horizontal rectangle, indicated by its point of contact with a vertical line or “play-head” on the left of the screen. In this way the score moves “towards” the performer from the right in the same direction as a traditional paper score.

Duration is represented proportionally by the length of the rectangle. The vertical position of the rectangle indicates its frequency, thickness indicates volume and shade indicates timbre. The number of sounds in each event is specified by a number attached to each rectangle. A portion of such a realisation is shown in Figure 6. The notation draws on conventions established in works by Cage and his colleagues Earle Brown and Christian Wolff, as illustrated in Figure 7.
This transcription enables a faster reading for performers using familiar symbols. In addition the graphs appear ahead of the 'playback', giving them time to consider their approach to the graphics. The score can be read by a single performer or multiple. The determination of continuums on their own.

This transcription enables a faster reading for indicated time to consider their approach to the graphics. The versions of the score can be generated for a group of performers networked to a master computer. A control panel (shown in Figure 8) allows for generation of a group of performers networked to a master computer.

The evaluation of the data to generate the scores of Variations I, II and III and a component of the score player were written in Java and embedded within the Max/MSP patch. The Java code for Variations I and II and the score player mechanism were written by Stuart James, and the Java code for Variations III was written by Aaron Weeks.

There were several advantages for re-implementing these processes in Java. One of these was the ability to access the same memory space that Max/MSP is pointing to, namely Jitter matrices, by utilizing the Java Jitter API. The mapping of both Java and Jitter processing proved to be an efficient way of accomplishing, storing, and sorting tables of values and referencing a stored Jitter matrix of note values determining the note polyphony within a designated time frame.

```java
int z = 0;
for (x = 0; x < coord.length; x++) {
    gB[z] = x;
    storage.setCell[gB, 0, coord(x)];
}
```

Here we see the score player mechanism referencing a stored Jitter matrix of note values determining the note polyphony within a designated time frame:

```java
for (int i = 0; i < timescale; i++) {
    if (value < timescale)
        %
    else if (value > timescale)
        %
}
```

There were several advantages for re-implementing these processes in Java. One of these was the ability to access the same memory space that Max/MSP is pointing to, namely Jitter matrices, by utilizing the Java Jitter API. This mapping of both Java and Jitter processing proved to be an efficient way of accessing, storing, and sorting tables of values and referencing a stored Jitter matrix of note values determining the note polyphony within a designated time frame.

4. VARIATIONS III

In Variations III, Cage moved to a significantly different score paradigm. Here the composer's focus was on actions rather than sounds. The score is created by distributing 40 circles (printed on individual transparencies) onto a surface and then removing all but the largest group of circles that are in direct contact with one another. According to Fitchett, Cage's aim was to "enable free and direct action in the performance - one would simply do things and count the actions and variables in performance." [29 p. 149].

The digital score for Variations III mimics this procedure: first randomly distributing circles on the screen, then calculating the distances between them and finding out all but the largest group of overlapping circles.

The score can be read by a single performer or multiple.

Figure 8: The master control panel for Dechel's realisation of the Cage Variations.
For this purpose there were advantages for implementation in a procedural language like Java by making use of recursive function calls. This proved to be significantly easier to program the way than using a scheduled message environment like Max. In this realization the adaptation of the two-dimensional layout of the score in Java it was possible for example to “undo” the circles along an infinitely decremental axis, while retaining the points of interaction with other circles in order to create a linear horizontal score. However, whereas the materials of Variations 1 and 2 give rise to a linear series of events and are therefore most appropriately rendered as a linear score, Variations III specifically avoids the linearity of the two-dimensional score itself. Cage instead asks the performer to “start at any circle” and “move on to any circle” only requiring the performer to “observe the number of circles which overlap it” [16]. This realization then, simply provides the means to (very slowly) regenerate the score until they are content with the resulting graphic while retaining the indeterminate finals and the Cage specify the performer in the original score.

6 CONCLUSION

Rendering these works digitally allows two dramatically opposed arguments often raised against indeterminate indeterminate works such as the Variations I and II. On one hand, since the audience always hears the works in a linear fashion sequentially in time there is always the question that the indeterminacy in somehow “loses” that the performance managed it before hand. In addition such works sometimes presented in the audience the notion that the performances are themselves “making it up” because “the only way to determine whether they are accurately reading the score.

The precision provided by the computer player for Variations I and II arguably lends legitimacy to the performances, because the score that is created is both “accurate” to a reasonable degree and easily read by the performer in a verifiable manner.

On the other hand, such works are sometimes criticized on the grounds that potential variations of other versions implies that the particular one that is being performed might not be the best example of the work. The ability to almost instantaneously generate multiple versions of the work, as demonstrated in Variations II, provides the opportunity to choose interesting and promising variations of the work.

We have attempted to be as authentic as possible to the specifications Cage prescribed in those three works, using technology to provide a platform that is precise and accurate in its rendition while still leaving open the human element of interaction with the score. As Milner expresses in regards to “authenticity” in the performance of those works:

Cage’s formal constraints [...] should be taken as points of departure and of periodic return in the course of developing mechanisms. They are the documents that express, however contingently at times, the works’ potentials and possibilities [25 p. 61].
REFERENCES

COMPUTING WITH CHORD SPACES
Donya Quick, Paul Hudak
Yale University
Department of Computer Science
New Haven, CT USA
donya.quick@yale.edu, paul.hudak@yale.edu

ABSTRACT
The number of solutions involved in many algorithmic composition problems is too large to be tractable without simplification. Given this, it is critical that composition algorithms be able to move through different levels of abstraction while maintaining a well-organized solution space. In this paper we present the following contributions: (1) extended formalizations and proofs needed to implement the chord spaces defined by Tymoczko [11] and Callender et al. [2], (2) a generalized framework for moving between levels of abstraction using quotient spaces that can easily be integrated with existing algorithmic composition algorithms, and (3) an application of both to voice-leading assignment.

1. INTRODUCTION
A major problem in the area of algorithmic composition is the need for organized and easily traversable sets of solutions, also referred to as solution spaces, which are tractable in terms of both runtime and memory requirements. Many music-theoretic ideas are also not formalized to the degree necessary to ensure correct implementation of algorithms and accompanying data structures. In this paper we address both of these problems by presenting a general framework for organizing and traversing harmony-related solution spaces. Our work builds on that of of Tymoczko [11] and Callender et al. [2], adding an additional layer of formalization necessary to create a generalized and extensible implementation. We then apply our framework to the task of voice-leading assignment, a common problem in music composition.

Consider the following situation: given a sequence of chords intended for a soprano, tenor, and baritone, rewrite the same chords for three tenors while factoring in additional constraints about each performer – perhaps one of the performers is a beginner, requiring smooth voice-leadings. This paper presents a set of algorithms and supporting proofs to automate algorithmic composition and arranging tasks such as above. Our approach utilizes two important concepts: chord spaces [2, 11] and musical predicates.

There are many other chord spaces that relate chords in different ways. These can also be used with our algorithm to perform variations on the voice-leading assignment task, allowing the algorithm a greater degree of control over musical features are generated. By simply changing the chord space, our voice-leading assignment algorithm can be generalized to make choices about pitch classes and octaves.

Data-driven algorithms such as Markov chains have been used to learn voice-leading behavior from collections of examples [3, 12]. Markov chains suffer from state explosion when addressing low-level features in music while still capturing structure. Variable-length Markov models [1] and probabilistic suffix trees [10] attempt to address this problem, but are still prone to the same problem with the large alphabets involved in musical problems. Chord spaces [2, 11] can help with this, since they allow generative problems to be broken into multiple steps, each at a different level of abstraction.

Chord spaces, however, present a number of repre-