Rolling the jChing:

A Java-Based Stochastic Compositional System

Robert Hamilton (rob@roberthamilton.org)

Abstract: The gamut-based compositional techniques utilized by composer John Cage in works such as *Music of Changes* defined a compositional framework which serves as the model for a compositional software application capable of transforming musical data cells using both chance-based and stochastic algorithmic functions. Written entirely in Java, the jChing makes use of the MusicXML data format to output transformed musical data. The jChing was designed to be used by composers to create dynamic reinterpretations of their own composed musical materials. This article outlines the functional model and technical specifications for the application and provides a basic example of the jChing workflow.

1 Introduction

While many composers find the application of algorithmic processes and models upon musical composition to be an effective and evocative method of work, the inherent calculation and mapping of computational data to musical form can be an extremely time-consuming and unnecessarily complex task. In *Music of Changes*, John Cage created a chance-based work by manually flipping coins to select musical data cells, which were in turn mapped to squares of the I-Ching. Cage's system was simple in design yet time-consuming in implementation. And while the methods the composer employed to select pitch and performance data were chance-based, the resultant composition remained fixed; the static notation of the piece removed any further element of chance from future performances.

The initial concept behind the jChing project was to replicate Cage's compositional process in software, automating both the pitch-cell selection and score-rendering processes, while at the same time creating a method where the transformations effected upon the original musical material could be easily and quickly replicated. In this manner, the resultant composition could itself be regenerated before every performance or even in real-time during a performance.

Using a class framework written in Java, it was relatively easy to create a basic model of Cage's Gamut structures and coin-flipping selection processes. The MusicXML (Recordare) data format provided a complete data output solution, whereby the functionalities of existing and prevalent music notation softwares such as Finale (Coda Systems) and Sibelius (Sibelius Group) could be leveraged in the presentation and further editing of output scores.

Once the basic system architecture was in place, it became apparent that the jChing system could be augmented to make use of a number of stochastic and probabilistic algorithms, further extending the range of compositional transformations available to a composer/user. The introduction of a basic system of weighting allowed for the possibility that algorithmically generated probability curves could be applied to aspects of the piece, including overall note density, note duration and note dynamics

1.1 Gamut-Based Composition

To create a chance-based compositional form from pre-defined musical phrases, Cage used a set of matrices of musical data cells mapped to cells of the I-Ching. These matrices are referred to as 'Gamuts' and the individual cells of each Gamut are referred to as 'Gamut Squares'. Separate Gamuts are used to hold note-phrase data (groupings of musical pitches and durations), dynamic level markings, and performance technique articulations. By randomly selecting Gamut Square cells, Cage sought to escape the inherent determinism he saw in traditional processes of composition. It was through chance selection that Cage realized a method of creating an overreaching compositional form without relying on his own compositional biases or preferences.

2 Using the jChing

Composers wishing to apply probabilistic algorithmic processes to their materials can enable weighting values for Gamut Squares to replicate more stochastic processes. After creating note-phrase cells through traditional compositional processes, composers enter those cells into the jChing input data file. Standard dynamic markings such as ***pp** (pianissimo)* and ***f** (forte)* are already defined in the system and can be given a weighting percentage to set the probability that those dynamics will be attached to any given note. Other system settings allow the composer to randomly or algorithmically assign various pitch transpositions and rhythmic expansions or diminutions to the cells. Currently the system is run by modifying and executing the Main.cls java class.

3 Object Data Structures

The highest hierarchically ordered object within the jChing object model is the Piece object. The Piece object acts primarily as a container for the Gamut object, the principle data-storage container for input data, and the Staff object, the principle data-storage container for output data. Piece objects provide logical data storage for Piece-level data elements (such as “Composer Name” and “Piece Title”). Both the Gamut object and the Staff object store Gamut Squares with the Gamut object reading Gamut squares note-by-note from the input data file and the Staff object ordering, transforming, and outputting Gamut Squares into a MusicXML formatted output file.

3.1 Data Input Structures

Contained within the Piece object, the Gamut and Gamut Square objects serve as key grouping data structures for the jChing data input process. Each Piece has one Gamut object, which in turn is made up of any number of Gamut Squares - cells of Note objects upon which transformations can be later performed.

As data is read from the input data file, individual Gamut Square objects are created for each successive Gamut Square. Within each Gamut Square, individual Note objects are created containing pitch and duration data for each note found in the input file. By the time the data input file has been fully processed, the Gamut object is populated with a set of Gamut Square objects, each respectively populated with a series of one or more Note objects. It should be noted that a musical rest is also considered a Note in this context and is stored as such.

3.2 Data Output Structures

During the algorithmic selection of Gamut Squares and their subsequent transformation, Staff objects are used both as containers for processed Gamut Square objects as well as representative staves within a multi-voice musical piece. When calculating the sequence of Gamut Squares for output, Gamut Squares are selected from the Gamut object based on the desired chance-based or probabilistic selection criteria and placed into a new order within the Staff object. In a multi-staved musical piece, all Gamut Squares for one Staff object are selected sequentially. Only after all squares for one staff have been selected will squares for the next staff be selected.

4 Gamut Square Transformations

The jChing is designed to allow users to effect transformations, either chance-based or probabilistic in nature, upon three core attributes of note sets within a piece: the ordering of Gamut Squares found in the piece, the relative amplitudes or dynamics for each note of the piece, and the absolute temporal durations of each note in the piece. Based on the differences in structure between Gamut Square objects and individual note objects, the data structures and methodologies by which the transformations are implemented are different. But whether algorithmic value selections occur in a hash-table of Gamut Square objects or in an array of amplitudes, the functionalities of these transformations remain essentially the same.

4.1 Gamut Square Ordering

The most basic form of transformation applicable to musical cells in a Gamut-based system is the relative order in time that each cell will be performed; essentially a virtual shuffling of Gamut Squares. By selecting cells at random a timeline of cells can be created to form a chance-based compositional structure. Such a transformation is wholly chance-based as it is equally probable that each cell will be chosen as any other cell.

By applying simple weighting values to each data cell, specifying the relative likelihood of each cell’s selection within the Gamut, a simple stochastic system can be created. Generating the weights of relative cells based on probabilistic functions such as Gaussian distributions can impose a more ordered stochastic form upon the cells, giving the composer even more control over what many would still consider a relatively “random” compositional form.

Whether a chance-based or probabilistic algorithm is used to select Gamut Squares from the Gamut, selected squares are placed in their new order into a Staff object. If the piece being generated contains multiple voices or parts, multiple Staff objects are populated with Gamut Squares until a pre-set limit of either Gamut Squares per Staff or a total length of beats is reached, at which time the next Staff object is populated.

4.2 Musical Dynamic Weightings

Just as a Gamut structure can be used to facilitate algorithmic selections of note cells, so too can a simple table structure be used to apply occurrence-weightings to different musical dynamics. Composers can set weighting values for each desired level of dynamic from ***pppp*** to ***ffff***. A table of values in the input data file matches dynamic markings to percentages (from a combined total of 100 for the entire piece). For instance, if the ***p*** dynamic is to appear more frequently than the ***f*** dynamic, the composer can set the ***p*** to have a weight of 20, while the ***f*** could be set to 1; therefore the probability of selecting ***p*** is twenty times greater than ***f*** and will most likely appear significantly more frequently in the piece

On a note-by-note basis, dynamics can be selected from this table and applied to the Note object currently being processed by jChing. Dynamics can be selected using either a strictly chance-based selection or more algorithmic processes. Functionally, the table with added indices, more accurately realized as a multi-dimensional array, acts in a manner similar to a Gamut.

4.3 Gamut Square/Note Scalings

Composers wishing to effect duration changes to Gamut Squares or Notes can enable an option whereby a percentage value entered in the input file will determine the probability that any given Gamut Square or Note object will have its durational value multiplied by a multiplier value. Multipliers are entered as a range of numbers and a divisor; as cells are selected for scaling, a value derived from the range (in increments of the divisor) will be used to multiply the cell's duration. For logistical reasons, the cell can be either a Gamut Square (whereby all durations of notes contained within will be scaled) or an individual Note.

5 Input Data Formatting

In the early stages of development, it became clear that input data needed to be organized in a manner that was both comprehensible to the human user as well as formatted in a robust and logical way to facilitate easy data parsing and processing. After considering a number of possible data formatting solutions, a modified version of the SCORE (L.Smith, 1987) music data format was chosen for its simplicity and comprehensive coverage of musical expressions and data types. The SCORE 4.0 Music Data Entry Reference Manual (Sapp, 2002) stands as the standard formatting model for all musical expressions of jChing data input.

5.1 Header Data Declarations

Data representing elements such as the Piece-Name and Composer-Name, as well as structural characteristics such as the number of staff-systems, respective system clefs, and the overall size of the excerpt are entered into the Header Declaration of the .gam input file. Data from the header declaration will be used to populate Piece-level data in the jChing object model. From the following header declaration for the choral work *Diane Sumus In Fide* for SATB chorus, we can see that each data-value is prefaced by a data-tag in capital letters. Multi-part data such as the four values for SYSTEMCLEFS are separated using "/" marks.

```
PIECENAME Dianae Sumus In Fide
COMPOSER Robert Hamilton
POET Catullus
RIGHTS Copyright 2003, CDS. Publishing
SOFTWARE jChing
ENCODINGDATE February 14, 2004
SYSTEMS 4
SYSTEMCLEFS T/T/A/B/
PARTS Soprano/Alto/Tenor/Bass/
SIZE 18
```

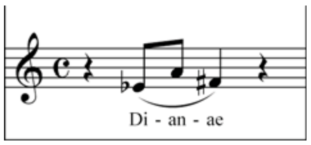
5.2 Dynamic Weighting Declaration

Following the header data the user can define relative weightings for the range of dynamic values to be used in the piece. In the following partial definition of dynamic levels, the "NULL" weighting is used to define a cell with no specific dynamic marking.

```
DYNAMICS
NULL    45
PPPP          0
PPP           0
PP   10
…
```

5.3 Gamut Square Declaration

Figure 1 depicts a composer-defined musical cell in both standard musical notation as well as in the jChing input data format. Pitches are declared as a note name and accidental followed by an octave number.



```
START
NUMBER 1
FLIPSET 666666
METER /4 4/
CLEF TR
KEYSIGNATURE K1S
DURATIONS Q/E/E/Q/Q/
NOTES R/Eb4/AN4/F#4/R/
WEIGHTING 1
        END
```

Figure 1. Individual Data Cell

The .gam data file lists each Gamut Square cell used in a piece, complete with individual numbers, meters, clefs, key signatures, note pitches, and note durations. Each cell is prefaced by a START data-tag and ended with an END data-tag. For Gamut Square 1, we can see that there is a Treble-clef in a key with no flats or sharps and four notes with respective durations of one quarter-note, one-eighth note, one-eighth note, one quarter-note and one quarter-note. Both Note durations and Note pitch values are entered in "/"-delineated lists, with the order of values in each list corresponding to the order of notes found in the particular Gamut Square.

The METER data field shows that the cell can be interpreted to have a time signature of 4/4. The FLIPSET data field has been included for implementations of Cage's coin-flipping scheme that wish to mimic the composer's techniques exactly. Additionally, the cell is given a WEIGHTING value of "1", meaning it is given no additional weight in probabilistic computations.

6 Data Output Formatting

Scores output by jChing make use of the MusicXML data format as defined in the MusicXML 1.0 Tutorial (Good, 2002) available from the Recordare.com website. According to the MusicXML data definition, score data can be grouped either in a "partwise" manner, where each score part or instrument is hierarchically superior to the measures contained within, or in a "timewise" manner, where each measure is considered one by one, grouping each individual part as belonging to the particular measure. By using XSLT stylesheets, MusicXML scores formatted in a "partwise" manner can be converted to a "timewise" manner, and vice-versa. Currently, scores created using jChing are formatted using the "partwise" definition. The cell-based design of the Gamut Squares and the linear nature of jChing's transformation processing seems more in keeping with the linearity of the "partwise" definition. For an in-depth description of data formatting using MusicXML visit the Recordare.com website.

6.1 Staff-based Output Ordering

Just as the MusicXML partwise data-definition sets up a macro-structure whereby individual measures are grouped together within parts, jChing orders data for output by selecting and grouping individual Gamut Square objects within Staff objects. Each Staff object acts as a container for all Gamut Squares that will make up a specific voice or part within the score; all notes from these Gamut Squares will be placed on the same staff on the written score. When transformations upon Gamut Squares are performed, the Staff objects monitor overall length and pitch ranges for the entire part and if need be can apply constraints to prevent the staff from growing too long or the range of the individual notes from exceeding the range possible for the staff's chosen instrument.

After outputting header information, the jChing processing steps through individual Staff objects and renders each Staff object as a separate part. Parts consist of a number of measures, each defined individually with both a set of measure-wide attributes and a set of note values.

6.2 Measure Calculation

One of the most important aspects of properly presenting scaled Gamut Squares on a written staff is the calculation of individual measure durations within each staff. When Gamut Squares are ordered in partwise fashion, each staff is ordered without consideration of events happening on other staves at the same time. One immediate issue is how to properly size measures and/or divide and tie notes together to preserve the intended rhythmic values of notes within Gamut Squares on a written staff.

Since the modification of time signatures for individual measures can have subtle differences in meaning for performers, the clearest solution of this issue is to effect a rule-based system of note splitting and tying. This system will be able to fit groupings of notes into a locked measure time signature and create subdivisions of tied notes (e.g. two-tied quarter notes instead of one half-note) to span measure lines. Such a system is currently under development.

6.3 Score Rendering

By applying either chance-based or probabilistic manipulations to the input score data, virtually any number of interpretations of the musical data could be made, resulting in an infinite number of possible Gamut Square orderings. Following the processing of data, musical scores can be created by simply importing the resultant score .xml file into a music notation software such as Finale or Sibelius.

7 Conclusion

The primary goal of the jChing project is to incorporate the chance-based and stochastic compositional processes utilized by composers such as John Cage and Iannis Xenakis into a score-generating compositional tool. By automating the mechanics of calculation for composers, the jChing can act as a valuable composition tool. In its current state, only basic system functionality is supported by the jChing, however the previously mentioned system enhancements all fit easily into the existing system framework. Additional development for the project is ongoing to incorporate more features and streamline existing processes. The scope of enhancements will include not only an increased number of stochastic and deterministic algorithms but also greater support for data input and output of MusicXML, a graphical-user interface for more user-friendly operation, and compatibility with the Max/MSP 4.5 implementation of Java for real-time use.

References

[Sapp 2002] Sapp, C., (2002). "SCORE 4.0 Music Data Entry Reference Manual", http://ccrma.Stanford.edu/~craig/score/input.

[Good 2002] Good, M., (2002). "MusicXML 1.0 Tutorial", http://www.recordare.com.