Simulating Ligeti
An Algorithmic Recreation of *Loop* in OpenMusic

I Introduction

Throughout the twentieth century there was a dramatic increase in musical works created through compositional process as opposed to compositional whim. This interest clearly starts with the combinatorial methods of the second Viennese school and runs through Boulez, Cage, Xenakis, Reich, and beyond. In order for compositional theory to be of any use we must move away from analyzing the results toward analyzing the process through which those results are achieved. By studying a musical work through statistical, algorithmic, and logical analysis we can better understand both the processes through which the work itself is created and the meaning buried beneath the notes. This paper is an attempt to do just that by studying the rhythmic parameter of *Loop* from György Ligeti's *Sonata for Solo Viola*.

After a description of the general construction of *Loop*, exploring its generative form and its relationship to chaotic systems, I will take the reader through various steps to creating a method of generating algorithmic simulations of *Loop*. This method will involve a combination of Markov Analysis and programming logic using a LISP Graphical User Interface for composition called OpenMusic (OM). By properly understanding the conditions which lead to successful simulations of *Loop* we are able to clearly distinguish the limits of its compositional process and the implementation of that process by an artistically sensitive composer such as Ligeti.

II General Construction

Loop is composed of a series of nine iterations of a 45-unit long string of dyads, preceded by a two bar introduction. The overall rhythmic scheme throughout *Loop* is divisible by a common denominator of one sixteenth note, with values ranging from 1-beat units to 8-beat units. Beginning with iteration one (heretofore referred to as $\mathbf{I}_1$) starting at bar 4, we have a rhythmically diverse presentation of the 45 dyads in a somewhat moderate rate. $\mathbf{I}_1$ uses almost the entire gamut of rhythmic units with 2- through 8-beat units all making an appearance. As the work moves from $\mathbf{I}_1$ to $\mathbf{I}_9$ the rhythmic pattern morphs from moderate and varied to fast and regular, with an almost constant stream of 1-beat units occurring in $\mathbf{I}_9$. Table II.1 shows the entire sequence of 16th note units as they move from $\mathbf{I}_1$ to $\mathbf{I}_9$ with their index (dyad #) in the top column.

What the data in Table II.1 show upon first inspection is the overall motion from rhythmic volatility in $\mathbf{I}_1$ to rhythmic standardization in $\mathbf{I}_9$. This motion is shown even more clearly when graphed as in Figure II.1, whereby each rhythmic value is plotted along the y-axis against its index in the x-axis. Graphed as such, the actual values are less interesting than the overall motion, we are able to see the trend from large variability and rhythmic value to flat, low value and almost completely even.

However, not only can we observe the data horizontally in iterative units, we can also see the transformation vertically of each dyad in the string. Thus a dyad such as number 14 occurs nine-times throughout *Loop* with the rhythmic sequence 3-3-2-2-2-1-1-1-1. When mapped individually, as in Figure II.2 for dyads 1 and 2, the generally rhythmic decay is still quite clear while the path with which that decay happens is more clearly represented. By comparing all 45 dyads mapped as such, two properties of the process become quite clear: 1) each dyad follows a *random* path of decay, and 2) each path of decay is *unique* to that dyad.

In general terms, we can say that *Loop* follows a generative process of decay whereby each atom (in our case any one of the 45 dyads) takes a *random* and *unique* path weighted towards 1-beat rhythmic units. The entire system, i.e. each iteration, moves from a *chaotic* state of high variability, entropy, and complexity to an almost totally *ordered* state of low variability, uniformity and simplicity. In nature this is the same general process of boulders becoming sand, mountains becoming plains, or snowflakes melting into water droplets.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
1	4	3	3	5	3	2	2	3	3	8	2	2	3	3	2
2	3	3	5	8	3	3	2	2	2	2	2	2	3	3	3
3	4	3	3	5	3	2	2	3	3	5	3	2	2	2	2
4	2	2	3	3	2	2	2	2	2	2	3	3	2	2	1
5	1	2	3	3	2	3	2	2	1	2	2	1	2	2	1
6	1	2	1	1	2	1	1	2	2	2	1	1	2	1	1
7	1	1	2	2	1	2	2	1	2	1	1	1	2	1	1
8	2	1	1	1	2	1	1	1	2	1	1	1	1	1	1
9	1	1	1	1	2	1	1	1	2	1	1	1	1	1	1

	16	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**
1	2	2	2	4	6	3	2	3	4	6	2	2	2	2	2
2	2	3	2	5	3	2	2	2	2	4	2	2	2	3	2
3	2	3	2	3	2	2	2	2	2	3	2	2	1	2	2
4	1	2	2	2	3	1	2	2	1	2	3	2	2	1	2
5	1	2	2	2	3	2	1	2	3	3	2	2	1	2	2
6	2	2	1	3	2	1	1	1	2	1	2	1	1	1	2
7	1	2	1	2	2	1	2	1	2	2	1	2	2	2	1
8	1	1	1	2	1	1	1	1	1	2	1	1	1	1	1
9	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	31	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**	**43**	**44**	**45**
1	2	3	3	5	7	6	5	3	2	2	3	3	8	2	2
2	3	2	2	3	3	5	3	2	2	3	3	2	2	2	2
3	3	2	1	3	2	3	4	2	2	2	2	2	1	2	1
4	3	2	1	2	3	4	3	1	1	2	2	2	2	2	2
5	1	3	2	2	1	2	2	1	2	2	1	2	1	2	2
6	2	1	3	2	3	2	2	1	2	2	1	2	1	2	1
7	1	1	1	2	1	2	2	1	1	1	1	1	2	1	1
8	1	1	1	2	1	1	1	1	1	1	2	1	1	1	1
9	1	1	1	1	1	2	1	1	1	1	1	1	1	1	1

Table II.1: Rhythmic Pattern for *Loop*

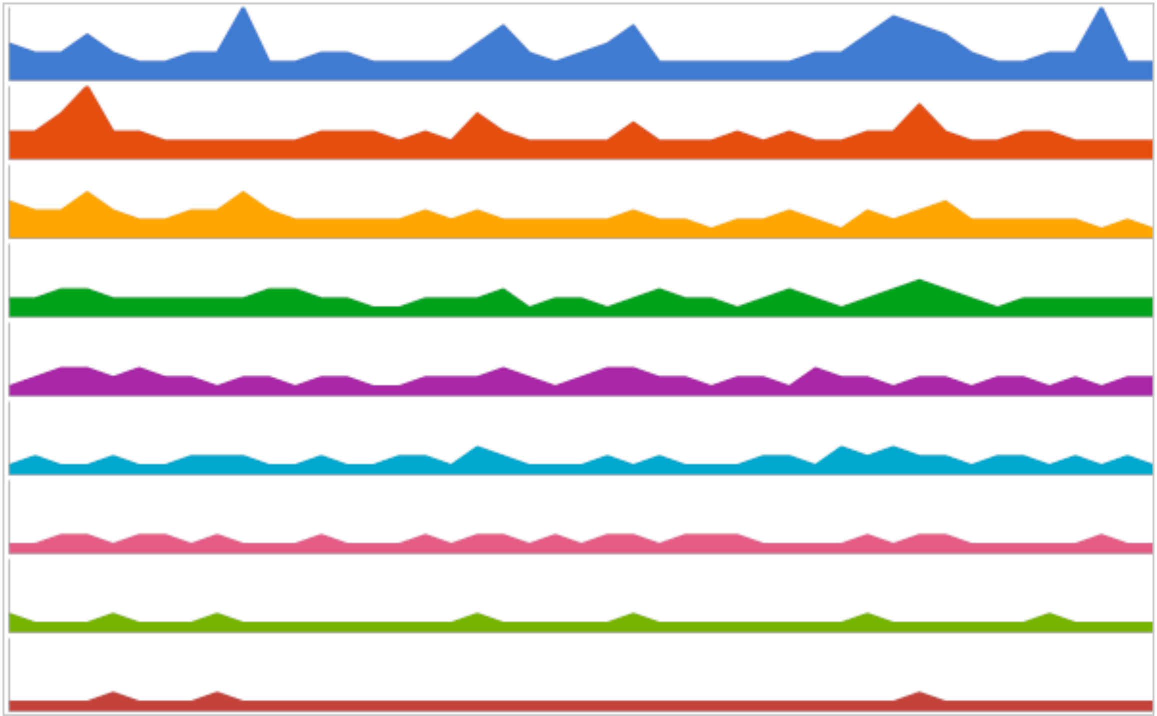

Figure II.1 General Rhythmic Transformation From $\mathbf{I}_1$ to $\mathbf{I}_9$

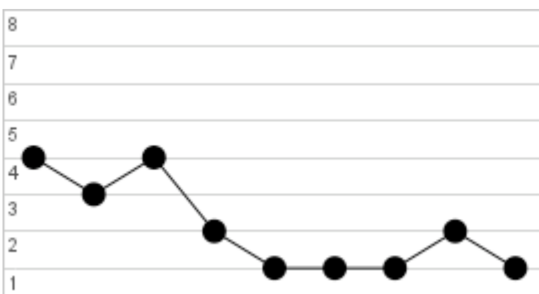

Figure II.2 A: Rhythmic Decay of Dyad 1

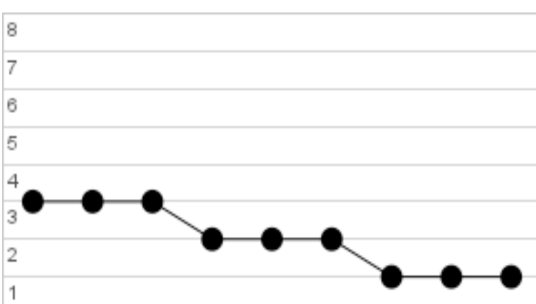

Figure II.2 B: Rhythmic Decay of Dyad 2

III General Algorithmic Process

In order to simulate the rhythmic decomposition of *Loop* we need to mechanize the derivation of each iteration. Exactly how this is done is a matter of both careful planning and taste; however, a general value that each simulation should be both varied from and similar to the original will be kept. This means that we would like to create an *algorithm*, or mechanized composition process, that can give us a variety of different solutions which all appear to be quite similar to the solution given by Ligeti. The general process of this algorithm is given in Figure III.1.

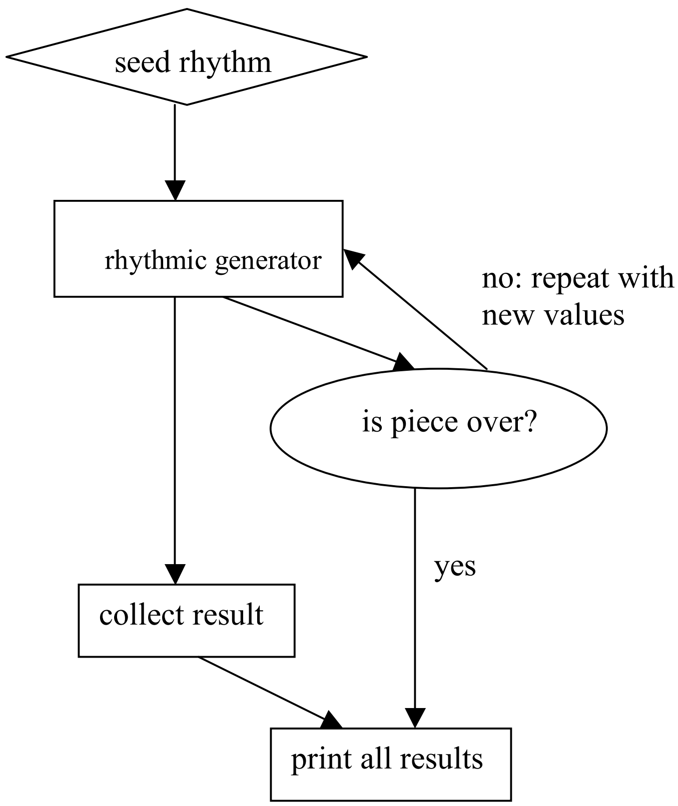

Figure III.1 General Algorithmic Process for *Loop* Simulations

In the schemata above, the seed rhythm is represented as a diamond which indicates a "fed value" – a value which the user inputs, the functions are given in rectangles and the logical operator "is piece over?" is represented in an oval. These three types of actions represent the basic building blocks of all algorithmic processes. "Fed values" can be either manually placed into the system or randomly derived; however, all algorithmic processes need some kind of data to work. Functions provide the basic transformations and/or actions taken on the data received and can range in complexity. Logical operators act by asking questions and can produce various results based on the answers to those questions. In the general algorithm presented in Figure III.1 one could set any kind of value to answer the question "is piece over?"; however, for our purposes we will follow the model of Ligeti by stating that the work is finished after eight repetitions of "rhythmic generator", taken with the seed value produces nine total iterations.

IV Markov Analysis

Given the rhythmic data provided by an iteration $\mathbf{I}_n$ of *Loop* it is possible to derive a very rough imitation by creating a simple probability distribution of all possible rhythmic events occurring. This is done by counting up the occurrences of each rhythmic unit and dividing by the total. For instance, $\mathbf{I}_2$ has 24 occurrences of 2-beat units, 16 occurrences of 3-beat units, 1 occurrence of a 4-beat unit, 3 occurrences of 5-beat units, and one occurrence of an 8-beat unit. Using this material we can create the probability distribution of $\mathbf{I}_2$ found in Table II.1.

	2	**3**	**4**	**5**	**8**
P(x)	24/45	16/45	1/45	3/45	1/45

Table II.1: Probability Distribution for Rhythmic States in $\mathbf{I}_2$

Using the "ChoixMultiple" function in OMAlea, we can randomly select any number of events following a given probability distribution (Figure II.1). We present the function (c) with a probability vector (a) representing the distribution from Table II.1 and a list of states (b) indexed to our probability vector. The function then randomly draws a state based on the probabilities of (a).

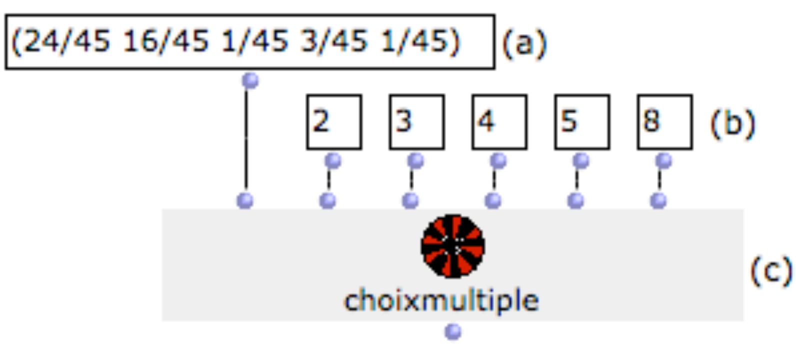

Figure II.1: Using "choixmultiple" To Generate States

By repeating this method 45 times we can create a somewhat satisfactory imitation of $\mathbf{I}_2$ (Figure II.2, compared with the original). However successful this method is in synthesizing an imitation of a given $\mathbf{I}_n$ it fails miserably to produce a satisfactory imitation of the entire *Loop* process. This is because each state in a given iteration is not randomly drawn from a probability distribution but is *generated* from the equivalent state in the previous iteration. Each rhythmic state after $\mathbf{I}_1$ is imbued with a memory of its previous state in earlier iterations, the entire transformation from $\mathbf{I}_1$ to $\mathbf{I}_9$ is thus a *Generative Process* as opposed to a *Random Process*.

In order to deal with generative processes such as *Loop* we need a different approach to probability theory, one that takes into account the formulation “given *State A* what is the probability that *State B* will occur”. The method through which generative probability can be explored is known as “Markov Analysis”. As an analytical tool Markov analysis is similar to the method of probability distribution above; however, it

Imitation

Ligeti

Imitation (continued)

Ligeti (continued)

Figure II.2: $\mathbf{I}_2$ imitated with a simple probability distribution compared to original

takes the set of probabilities of all initial states becoming resultant states. For *Loop* this means measuring not the occurrences of states in an iteration $\mathbf{I}_n$ but the transformations of states across two iterations (i.e. the probability of a X-beat unit from $\mathbf{I}_{n-1}$ becoming a Y-beat unit in $\mathbf{I}_n$).

The most common method for presenting the resultant data is in what is known as a “Transition Table” whereby the initial states are listed vertically in the left-most column and the new states are listed horizontally on the top-most row. The probability of *State A* becoming *State B* is then filled in for each cell of the table. Table II.2 presents the transition table of a 1st Order Markov Analysis of $\mathbf{I}_2$. By looking at an initial state from $\mathbf{I}_1$ we can look up the probability that a given state will occur at the same dyad in $\mathbf{I}_2$.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	-	-	-	-	-	-	-	-
2	-	13/19	6/19	-	-	-	-	-
3	-	4/7	5/14	-	1/14	-	-	-
4	-	1/3	1/3	-	1/3	-	-	-
5	-	-	2/3	-	-	-	-	1/3

6	-	-	1/3	1/3	1/3	-	-	-
7	-	-	1/1	-	-	-	-	-
8	-	1/1	-	-	-	-	-	-

Table II.2: Transition Table $\mathbf{I}_1$ to $\mathbf{I}_2$ (1^{st} order Markov Analysis)

This method can be expanded for all transitions $\mathbf{I}_1$ through $\mathbf{I}_9$ ddd

Another way to present the results of a Markov analysis is in a map known commonly as a Markov Chain which shows the probability that any given state will become a resultant state. This method better suits an alternative approach that focuses on the transformation of each individual dyad as opposed to the total transformation from iteration to iteration of all dyads as in the 1^{st} order markov analysis above. Figure II.3 shows the Markov Chain of dyad 25 from $\mathbf{I}_1$ to $\mathbf{I}_9$ in *Loop*. The usefulness of this diagram lies in exposing the probability of possible paths of rhythmic decay given the information at hand and allowing the synthesizer to recreate an alternative path based on those data. Starting with an initial rhythmic value of 6 for $\mathbf{I}_1$, every simulation of dyad 25 will jump down to a rhythmic value of 4 by $\mathbf{I}_2$ and 3 by $\mathbf{I}_3$ due to the 100% probability of the transitions represented in the Markov Chain. From there the possibilities using 1^{st} order Markov Analysis double, the rhythmic value of $\mathbf{I}_4$ could be either 1 or 2 with equal probability. In this way, the original rhythmic value of 6 in $\mathbf{I}_1$ can randomly shifts until it reaches the end state (naturally, after passing through a rhythmic value of 1).

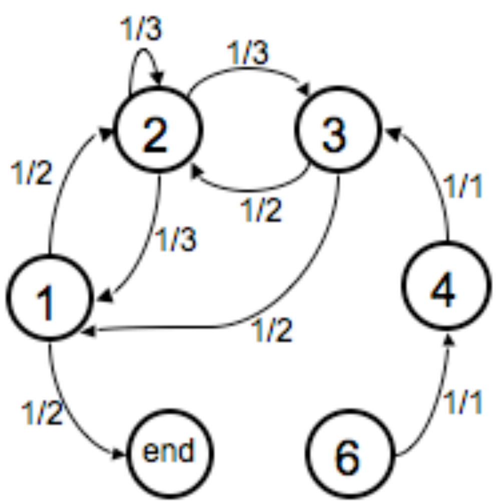

Figure II.3

	1	2	3	4	5	8	Total
1	-	-	-	-	-	-	0
2	1/6	2/3	1/8	-	1/24	-	24
3	-	9/16	5/16	1/8	-	-	16
4	-	-	1/1	-	-	-	1
5	-	-	1/1	-	-	-	3
8	-	-	-	-	1/1	-	1

Table 2: Transition Table $\mathbf{I}_2$ to $\mathbf{I}_3$ (1^{st} order)

	1	2	3	4	5	Total
1	1/4	3/4	-	-	-	4
2	6/25	3/5	4/25	-	-	25
3	-	2/3	1/4	1/12	-	12
4	-	1/2	1/2	-	-	2
5	-	1/2	1/2	-	-	2

Table 3: Transition Table $\mathbf{I}_3$ to $\mathbf{I}_4$ (1^{st} order)

	1	2	3	4	Total
1	2/7	4/7	1/7	-	7
2	7/28	18/28	3/28	-	28
3	1/3	1/3	1/3	-	9
4	-	1/1	-	-	1

Table 4: Transition Table $\mathbf{I}_4$ to $\mathbf{I}_5$ (1^{st} order)

	1	2	3	Total
1	8/11	2/11	1/11	11
2	10/27	15/27	2/27	27
3	5/7	2/7	-	7

Table 5: Transition Table $\mathbf{I}_5$ to $\mathbf{I}_6$ (1^{st} order)

	1	2	Total
1	6/11	5/11	22
2	3/5	2/5	20
3	2/3	1/3	3

Table 6: Transition Table $\mathbf{I}_6$ to $\mathbf{I}_7$ (1^{st} order)

	1	2	Total
1	23/26	3/26	26
2	15/19	4/19	19

Table 7: Transition Table $\mathbf{I}_7$ to $\mathbf{I}_8$ (1^{st} order)

	1	**2**	**Total**
1	37/38	1/38	38
2	5/7	2/7	7

Table 8: Transition Table $\mathbf{I}_8$ to $\mathbf{I}_9$ (1^{st} order)