Revisiting the Illiac Suite – a rule based approach to stochastic processes

Örjan Sandred
University of Manitoba
Studio FLAT

Mikael Laurson
Sibelius Academy
CMT

Mika Kuuskankare
Sibelius Academy
CMT

ABSTRACT

This article will first discuss the use of probability distribution in L.A. Hiller and L.M. Isaacson's string quartet the *Illiac Suite*. After pointing out some limitations with the technique used in the Illiac Suite, the use of stochastic rules in a constraint-based system will be introduced. Finally two possible versions of the beginning of the last movement in the Illiac Suite will be used to demonstrate the combination of stochastic and ordinary rules.

1. Introduction

Many early experiments in Computer Assisted Composition were inspired by the use of indeterminacy in contemporary music. John Cage and the group that grew up around him used chance to do away with the traditional control over the material. When composers began using computers, they had a tool that could take random decisions very quickly.

The first examples of random processes in computer music composition can be found in L.A. Hiller and L.M. Isaacson's string quartet the Illiac Suite (Hiller and Isaacson 1957, 1958). Hiller and Isaacson describe the Illiac Suite as a chronological record of experiments. The general idea is to use screening rules to accept or reject randomly generated pitches and rhythms. Probability distribution and Markov processes can also be found in the suite.

Around the time the Illiac Suite was composed, I. Xenakis established himself as the pioneer who explored stochastic techniques (with or without the aid of a computer). His music can serve as a catalogue of possible approaches (Xenakis 2001). Several composers have continued Hiller's and Xenakis' early work: J. Tenney's first instrumental work using Computer Assisted Composition, *Stochastic String Quartet* from 1963 (Tenney 1988), was inspired by both Xenakis and Hiller. A later example is T. DeLio's use of Markov Chains in his *Serenade* from 1976. C. Ames, a student of L.A. Hiller, has contributed both as a composer and a writer (Ames 1987, 1990).

Stochastic techniques are good for formalizing overall tendencies in music. The weakness of stochastic techniques is that the details are left to the randomness that characterizes the process. Some composers have dealt with this weakness by using larger units than pitches or durations, for example chords or phrases, within stochastic processes. However, stochastic techniques remain inaccurate for managing structural details.

The objective of our work described in this article has been to combine stochastic processes with the detailed control that is possible with constraint-based computing. Our intention has been to leave the definition of tendencies and rules to the composer of the music. Our interest has not been to research into machine learning techniques nor to recreate a musical style. We have used the 4th movement in the Illiac suite as a case study for this article.

We have implemented our examples in the PatchWork Musical Constraint (PWMC) system, a further development of the OpenMusic Rhythmical Constraint system (Sandred 2006). PWMC is an extension to the PatchWorkGL visual programming language (Laurson and Kuuskankare 2006) for Computer Assisted Composition. PWMC is a framework on top of the PMC constraints solver (Laurson 1996) that is part of the PWGLConstraints system and programmed in Lisp. PWMC was developed by Örjan Sandred at the University of Manitoba, Canada, and PWGLConstraints was developed by Mikael Laurson at the Sibelius Academy, Finland. There are currently several other approaches oriented towards musical search in addition to our constraint-based systems, such as Situation (Rueda et al., 1998), OMClouds (Truchet et al., 2001) and the recent Strasheela system based on the OZ programming language (Anders, 2006).

2. The 4th movement of the Illiac Suite

The pioneering work done in Computer Assisted Composition by L.A. Hiller and L.M. Isaacson in the 1950s shows several examples of random processes.

2.1. The Illiac Suite and probability tables

In the 4th movement of their string quartet the *Illiac Suite* Hiller and Isaacson used probability tables to control the distribution of melodic intervals in the four voices. The rhythm in the movement was set to an ostinato eight-note pulse; the computer was only used to decide what pitches to assign to the eight-notes.

The probability table with the distribution of melodic intervals was changed every second measure. The first table set the probability for repeated pitches to 100% and any other melodic interval to 0%. If the computer assigned the same pitch to two consecutive eight-notes they were automatically slurred to constitute a longer note value. Thus the movement starts with a sustained pitch for two measures (without any melodic movement) in all four voices.

Step-by-step, the vocabulary of melodic intervals expands to include octaves, fifths, fourths, major thirds, minor sixths, minor thirds, major sixth, major seconds, minor sevenths, minor seconds, major sevenths and tritons (in that order). In this way the linearity becomes increasingly more complex. Hiller and Isaacson always setup each probability table so that simpler intervals have higher probabilities (i.e. unison is more likely than an octave, and octave is more likely than a fifth, etc).

In an article about the Illiac Suite Hiller briefly mentions that they used more complex dependencies for the probabilities later on in the movement, such as letting the last choice of interval have an impact on the next choice.

2.2. The Illiac Suite – a critical listening to the aural result

A consequence of using probability or Markov tables in music composition is that the distribution can only be controlled in one dimension in a score. In Hiller's case the horizontal melodic contour was generated, but having a preference for what the vertical harmonies should be is not "part of the game". The opposite would have been equally possible.

Our critique of Hiller and Isaacson's experiment is that the aural effect of the music is very much colored by what is *not* defined in the probability tables. The lack of harmonic control is soon obvious; already when the third probability table is used (that includes repeated pitches, octaves and fifths as possible melodic intervals) in measure 5 – 6 it becomes clear that the harmonies are there as a side-effect of melodic movements. After another few measures the four voices walk over the full chromatic spectrum.

The movement begun without any consideration of the melodic context; the melodic intervals are distributed independently of each other. This allows the melodies to "walk away" to form very complex profiles. Hiller and Isaacson are conscious of this situation, and they experiment with different approaches where choices partly depend on the opening note of the passage, or on the preceding interval. It is not clear how this was done in detail, and judging from the aural result and the score, the impact is not always obvious.

3. Probability distribution as stochastic rules

We propose to implement probability distribution as a stochastic rule in a constraint-based system. Rules have the advantage of being modular – it is possible to add rules to restrict related dimensions (i.e. both the horizontal melodic contour and the vertical harmonic structure can be controlled by separate rules).

We have tried two different approaches to design probability rules. Both designs imitate the behavior of a stochastic process. Instead of using a probability table in a random process, the rule makes sure pitches (or rhythms) in a generated sequence are an acceptable representation of a probability table. The constraint system can be set to generate the sequence by proposing pitches (or rhythms) randomly, or systematically going through the domain of possible candidates. In the first case, the process resembles the stochastic process. In the latter case, the process is deterministic. In both cases the probability rule will filter out a sequence that follows the probability distribution.

Our first design uses a count value as a threshold for how many times a given pitch may exist in the sequence. For example if the constraint based system generates a sequence of 50 pitches, and the probability table states that 20% of these should be the pitch C4, then a maximum of 10 pitches can be a C4. As a result of the combination of all count values, an accurate representation of the probability table will exist when the sequence is complete (this is only true if the pitches that are not part of the probability table have the count value 0). For more detailed information see Laurson (1996).

Our second design checks that the probability table is an acceptable representation of the generated sequence of pitches (or rhythms) every time a new element is added to the sequence. Only pitches (or rhythms) that maintain an acceptable distribution at each step during the calculation are allowed to be added. The maximum accuracy that can be expected depends on the length of the sequence. For example if a sequence of pitches only contains two pitches, a pitch can not exist, exist once or exist twice in the sequence. Possible probabilities are thus 0%, 50% or 100%. If the table states that a pitch should have a 40% probability to occur in the sequence, a precise representation is not possible. Our implementation calculates the maximum deviation from the given probability table that might occur. The rule will always accept this deviation. Since the sequence gradually will become longer, the accuracy will increase during the generation of the sequence. In the given example the highest guaranteed accuracy will be +/- 50%.

$$a = \frac{100}{n}$$

a = highest guaranteed accuracy (percent)
n = length of sequence

The disadvantage of the first design is that the probabilities will not be accurate until the end of the sequence. The beginning of the sequence has little if any restrictions, while the end will be forced to strictly fit into the count values. In combination with other rules, time consuming backtracking might be necessary late in the search. The disadvantage of the second design is that the sequence might follow the probability table too closely, without allowing any fluctuations. The sequence will have the same distribution throughout its whole length. This might not be desired; for example unexpected events and surprises (that can be of great value in music) are if not impossible very unlikely.

In both designs, we add a tolerance factor that will allow additional deviations from the given probability table. Especially when combining a probability rule with other rules, it might be necessary to allow a certain deviation to find a possible solution. Deviations also open up for unexpected events in the second design above. Pitches (or rhythms) that are not defined in the probability table are by default not allowed in the sequence (the tolerance factor will only affect the distribution of events that are defined in the probability table).

The text below will refer to our second design of the probability rule.

4. The implementation – PWMC

The experiments described in this article were done using the Patch Work Musical Constraints system (PWMC). PWMC is a system for generating musical structures based on a domain of pitches, rhythms and metric units. PWMC stores the generated score in a database. The database is gradually filled with candidates from the domain that fulfill rules defined by the user. In PWMC, pitch, rhythm and time signatures can all be unknown before the search is done, but it is also possible to set the sequence of pitches, rhythms or time signatures to a predefined sequence (as is the case with the sequence of rhythms in the example below). The rules typically constrain how pitch and rhythm can interact in relationship to their metric positions within a single voice as well as between two or more voices.

PWMC does not use the pattern-matching mechanism from PWGLConstraints. Instead PWMC searches its database to find the appropriate instances when rules are checked.

PWGLConstraints has a strictly defined search order for how variables in the generated sequence are assigned values. The search order in the PWMC system is more open. PWMC searches for a solution sequentially within each voice (starting from the beginning of the sequence). Beside this fundamental principal, the order for how voices are built is by default not known. The order between searching for pitches and rhythms within a voice is also not known. Typically the next assigned value will be in the voice with the shortest sequence. The values are typically alternating between durations and pitches in the solution.

It is outside the scope of this article to fully describe the PWMC system. The full power of PWMC was not used in the examples for this article: rhythm was only treated as a side effect of repeated pitches.

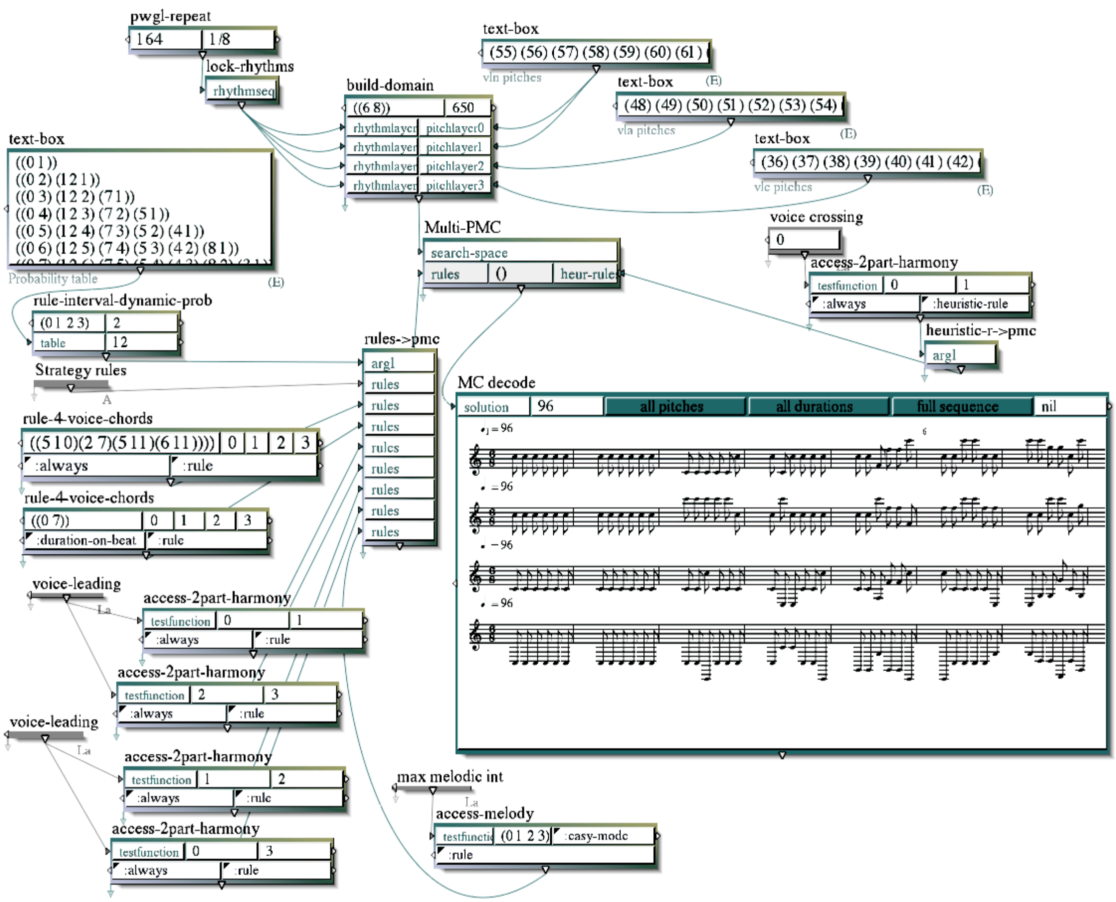

Fig.1. The PWMC patch used to generate the examples described in this article. The stochastic rule is called "rule-interval-dynamic-prob" in the patch, and the beginning of the probability table can be seen in the text box above the rule.

5. The 4th movement of the Illiac Suite; two new possible versions

Our objective here is to demonstrate how it is possible to have control over harmony and voice leading without violating Hiller and Isaacson's probability tables for the 4th movement of the Illiac Suite. We created a probability rule for melodic intervals based on the 2nd design described above. We used Hiller and Isaacson's original probability tables (changing the table every second measure) for generating pitches in four voices, and we allowed maximum 2% deviation from the table. The rhythm was set to a sequence of repeated eighth notes and the time signature was 6/8.

To this we added a rule for controlling the harmonies made up of the four voices. The rule would only allow certain chord structures between the voices. In our first experiment we did the obvious; we allowed only major and minor triads (and all their possible inversions). We also added a second rule for controlling voice leading. We did not allow parallel unisons, octaves or fifths between neighboring voices, neither between the outer voices. Finally we added a heuristic rule (i.e. a rule that is not strict, but gives the system a preference for certain types of solutions); the first voice is preferred to be above the second voice. Since the two violins have the same register, voice crossing would otherwise frequently occur for no reason. Despite this heuristic rule, the computer found reasons to have quite a few voice crossings.

The result is as expected a far more tonal-sounding Illiac movement than Hiller and Isaacson's original composition. Knowing the amount of possible variations that can be generate from the original probability tables alone, it is highly unlikely that Hiller and Isaacson would have come up with this version. At the same time, it is easy to prove that Hiller and Isaacson's probability tables are respected. Our version is therefore just as likely as any other version of the movement.

In a second experiment we changed the allowed chord structures to be based on either fourths, fifths or augmented fourths; a set of four allowed chord structures was given to the harmony rule (see fig.2). The rule would also allow all possible inversions of these chord structures, however we made a restriction: minor seconds were not allowed between neighboring voices (inversions of the last two chords in fig.2 could otherwise have created minor seconds in the harmony). At the first beat in each measure we made the rule stricter; at these positions only open fifths (and octaves) were allowed.

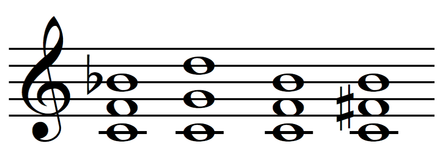

Fig. 2

We kept the voice-leading rule from our first experiment, but we added a link between the top two voices as well as between the bottom two voices. Each time there is a repeated pitch in one of the voices, the other voice in the pair must repeat its pitch as well. Since repeated pitches are automatically slurred (this is how Hiller and Isaacson created sustained pitches in the original version), the top two voices will have the same rhythm. The bottom two voices will have the same rhythm as well. We also added the heuristic rule from our first experiment to avoid voice crossing. Finally we restricted the maximum distance between the outer notes for two consecutive melodic intervals; maximum an octave was allowed. The purpose of this rule is to avoid melodies that walk over the whole register (for example this rule will force an octave to be followed by a repeated pitch or a contrary motion).

Each probability table in the Illiac Suite is valid for two measures. Within two measures each voice has 12 consecutive melodic intervals. The highest guaranteed accuracy will thus be 100 / 12 = 8.3%. With the tolerance set to 2% we can expect our score to show a probability distribution that differs up to +/- 10.3% from the given probability table.

Our second experiment shows many aural structures. Just as it would have been unlikely for Hiller and Isaacson to come up with anything that resembled our first experiment, our second experiment is equally unlikely without the added rules. However our second experiment respects Hiller and Isaacson's probability tables and is therefore a valid version.

Fig.3. The beginning of the score for our second experiment.

6. Conclusion

The concept of probability distribution can be useful in music. Development over longer time spans, as well as frequency of events, can be controlled. By implementing probability distribution as a rule in a constraint based system it is possible to combine this concept with other structural restrictions. The constraint solving systems of today (that were not available when the Illiac Suite was composed in 1956) make complex problems with dependencies between linear and vertical pitch dimensions possible to solve.

7. References

Ames, C. (1987). *Automated Composition in Retrospect: 1956-1986*. In Leonardo, Vol. 20, No. 2: 169-185.

Ames, C. (1990). *Statistics and Compositional Balance*. In Perspectives of New Music, Vol. 28, No. 1: 80-111.

Anders, T. (2007). *Composing Music by Composing Rules: Design and Usage of a Generic Music Constraint System*. PhD Dissertation, Queen's University, Belfast.

Hiller, L. and Isaacson, L. (1958). *Musical composition with a high-speed digital computer*. In Journal of Audio Engineering Society 6: 154-60.

Hiller, L. and Isaacson, L. (1957). *Illiac Suite*. Score, Theodore Presser Co, New York, USA.

Laurson, M. (1996). *PATCHWORK: A Visual Programming Language and Some Musical Applications*. Ph.D. Dissertation, Sibelius Academy, Helsinki, Finland.

Laurson, M. and Kuuskankare M. (2006). *Recent trends in PWGL*. In Proc. ICMC 2006, New Orleans, USA.

Rueda C., Lindberg M., Laurson M., Bloch G. & Assayag G. (1998). *Integrating Constraint Programming in Visual Musical Composition Languages*. ECAI 98 Workshop on Constraints for Artistic Applications. Brighton, England.

Sandred, Ö. (2006). *Kalejdoskop for clarinet, viola and piano*. In The OM Composer's Book - Vol.1, Editions Delatour France / IRCAM. 223-235.

Tenney, J. (1988). *Stochastic String Quartet*. Score, Smith Publications/Sonic Art Editions, Baltimore, USA.

Truchet, C., Assayag, G. & Codognet, P. (2001). *Visual and Adaptive Constraint Programming in Music. Proceedings of the International Computer Music Conference*. Havana, Cuba. 346-352.

Xenakis, Iannis. (2001). *Formalized Music: Thought and Mathematics in Composition.* (Harmonologia Series No.6). Hillsdale, NY: Pendragon Press.