Target-Based Rhythmic Pattern Generation and Variation with Genetic Algorithms

Cárthach Ó Nuanáin, Perfecto Herrera and Sergi Jordà
Music Technology Group
Universitat Pompeu Fabra
Barcelona, Spain
carthach.onuanain@upf.edu

ABSTRACT

Composing drum patterns and musically developing them through repetition and variation is a typical task in electronic music production. We propose a system that, given an input pattern, automatically creates related patterns using a genetic algorithm. Two distance measures (the Hamming distance and directed-swap distance) that relate to rhythmic similarity are shown to derive usable fitness functions for the algorithm. A software instrument in the *Max for Live* environment presents how this can be used in real musical applications. Finally, a user survey was carried out to examine and compare the effectiveness of the fitness metrics in determining rhythmic similarity as well as the usefulness of the instrument for musical creation.

1. INTRODUCTION

When composing drum tracks in electronic music, it is typical that a producer begins with a basic loop or pattern which is then iterated on. Depending on the nature of the music, the amount of variation that the producer imparts can range from very little to quite a considerable amount. Contrast for example, the subtle changes that occur in stylistically minimal techno to the constantly shifting, complex patterns prevalent in IDM (Intelligent Dance Music).

Finding ways of automating this kind of activity can be useful for producers. For example, it could help to quickly lay a rhythmic foundation for a work in progress track, allowing the producer to focus on the "bigger picture", or provide an intelligent agent that accompanies a laptop performer in live situations. To address this, we outline a method that, when given a target input drum pattern, generates patterns with increasing similarity to the input pattern by using genetic algorithms.

Genetic Algorithms are a class of algorithm for solving search problems inspired by the biological metaphor of evolution. The core operation of genetic algorithms entails the generation of populations of potential solutions by simulating the process of "natural selection". Fitter candidates are selectively paired together to spawn new offspring using crossover and mutation. This approach is useful in situations where the search space is large and prohibitively expensive to search exhaustively by more conventional techniques such as depth-first and breadth-first search.

Genetic algorithms have been shown to assist in computer music composition. *GenJam* by Al Biles is perhaps one of the most well-known early realisations of this, whereby solo phrases in the jazz idiom are generated continuously [1]. One of the attractive aspects of genetic algorithms is the possibility of a human interactively appraising the output of the algorithm as it progresses. This is done in hope of reconciling the domain agnostic "objective" operation of the algorithm with the subjective, artistic goals of the critic.

The problem with this, of course, is the complexity of analysing many candidates from the algorithm and, specifically for temporal domains like music, sequential analysis of those candidates. This is known as the "fitness bottleneck" [2]. Solutions to the fitness bottleneck are dealt with in myriad ways. Biles, for example, proposes the elimination of the role of the fitness function completely, just preserving the aspects of crossover and mutation [3].

In the approach we describe, we derive our fitness function formally and programmatically by evaluating the similarity of each candidate pattern with respect to a target pattern which the composer determines initially. To compare two rhythmic patterns to each other in terms of similarity, the algorithm needs a distance function that can produce a value accordingly. Based on a review of the literature dealing with perceptual rhythmic similarity, we chose and implemented two such measures, namely the Hamming distance and the directed-swap distance. In Section 2 these will be explained in more detail and the evaluation section discusses how the participants responded to them in the user survey. To our knowledge, this is the first reported research that integrates rhythmic similarity perception with genetic algorithms for pattern generation.

The structure of the paper is as follows. The next section will examine the state of the art in musical genetic algorithms and similarity measures for rhythmic patterns. The methodology section explains our approach and shows the operation of the software instrument designed. Results of the user evaluation present our findings regarding the efficacy of the similarity measures and the musical output.

2. EXISTING RESEARCH

2.1 Genetic Algorithms and Rhythm

A number of papers dealing with genetic algorithms and rhythm are presented here, and specific attention is directed to the derivation of a fitness function in each case.

Eigenfeldt [4] describes his Kinetic Engine: a software component that generates rhythms in general, not specific to drum sounds. His approach to fitness evaluation is perhaps a controversial one but not uncommon in musical applications. As in Al Biles' *GenJam* system, the role of fitness is simply eliminated. Thus the only real elements of genetic algorithms conserved are crossover and mutation. One may rightfully suspect that such a simplification renders the algorithm commensurate with a randomised search. Consequently a common solution used in such scenarios is to seed the initial population with a known dataset of good input. Another approach could be to embed some rules-based logic that restricts what type of candidates can be generated legally in the seeding process, as used by one of the authors of this paper in [5].

Bernardes et al. approach genetic drum pattern generation from the point of view of style emulation with statistical analysis of existing musical material [6]. Like Eigenfeldt, they do not incorporate a fitness function, but choose to perform prior analysis on user-supplied or preset MIDI files. This process gathers a probability distribution of weighted possible onset times in a 16-step pattern. The population is then seeded with candidates that have patterns generated according to this distribution.

A short paper by Horowitz suggests a multi-dimensional objective fitness function based on the combination of functional evaluations of syncopation, density, downbeat, beat repetition etc. [7]. Unfortunately the publication is rather scant on actual details regarding the implementation and evaluation of such a method.

Kaliakatsos–Papakostas et al. also use the notion of a a target pattern in evoDrummer [8], and define their fitness function by determining what they refer to as divergence in terms of "mean relative distance" to a base rhythm. The distance is computed based on a set of 40 features extracted from patterns, including descriptions of density, syncopation and loudness intensity. Indeed this is perhaps the most related work to our proposed system. However, to concentrate on the impact of distance measures between two simple patterns of onsets we have refrained from integrating such extensive feature vectors. The next section takes a look at various measures of rhythmic similarity in more detail.

2.2 Measures of Rhythmic Similarity

The most basic measure of similarity between two strings in general is the notion of edit distance. The edit distance defines the number of discrete insertion, deletion and substitution operations required to make one string match another [9]. Substitutions tend to be the most economical. For strings of the same length, and by restricting the operations to substitutions, this then corresponds to the Hamming distance, or simply the number of positions in which they differ [9]. Furthermore, if the strings are binary the distance can be computed quickly with the logical XOR operation. This is useful as chromosome representations in genetic algorithms are frequently represented in this manner, which we will see in more detail in the next section. Paiement et al. [10] have reported the distance to outperform Hidden Markov Models in distance modelling of rhythmic data, and suggest that derived models could be used for drum machines.

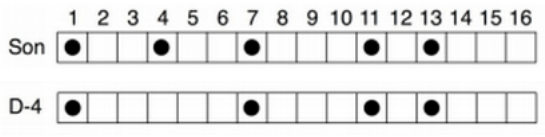

Figure 1. Son Pattern and a Variation [1]

Post and Toussaint have investigated the perceptual merit of the edit distance as a measure of rhythmic similarity [9]. It is compared to the swap distance, which considers the adjacent distance cost of mapping onsets between patterns. A swap cost of 1 is assigned for every adjacent movement an onset makes from its original position to a new position. An additional stipulation the swap distance model makes is that for two patterns with different number of onsets, every onset from the pattern with the greater number of onsets must be mapped onto an onset in the pattern with less onsets. For example, the cost of mapping the clave son (the "clave" is a fundamental rhythmic motif in Latin music) to the pattern D-4 (Fig. 1 above) would be 3, since onset 2 at position 4 needs to move 3 positions to onset 1 in position 1 or onset 2 in position 7 of D-4. Informally one would suppose that such a measure may better reflect the level of syncopation between two patterns.

Overall Toussaint concluded that the edit distance was a more robust measure of similarity that correlated quite well with listener judgements, when compared to the swap distance. This paper examines how these two measures can be applied to creating an automatic fitness function for a genetic algorithm that creates rhythm patterns. The next section will describe how this is implemented.

3. METHOD

This section introduces the tools we created in our research, namely *SimpleGA*: the genetic algorithm itself and *GenDrum*: the final *Max for Live* instrument. This is followed by a detailed discussion of the implementation of the distance measures and representational issues. Finally there is a description of the design of the experiment for evaluating the research.

3.1 Software Implementation

SimpleGA is a basic, general purpose genetic algorithm external object we developed for the *Pure Data* [11] and *Max/MSP* environments in C++ using the *Flext* framework [12]. It can handle binary, numerical and alphanumerical

[1] Image and example from [9]

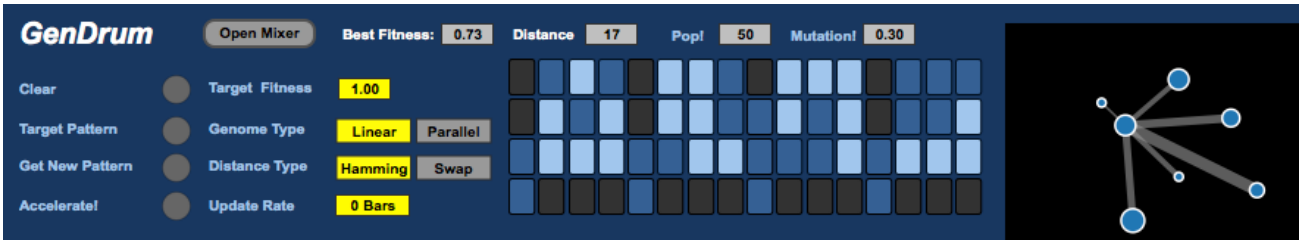

Figure 2. GenDrum Interface

strings. A target string is supplied to the object and fitness is determined by measuring its distance to the generated strings using the Hamming or directed-swap distance, which will be explained in more detail later. Bang messages to the leftmost "hot" inlet causes the genetic algorithm to undergo a generational stage of evolution then output the fittest individual from the pool.

Next we created the *GenDrum* instrument, a *Max for Live* device that uses the *SimpleGA* genetic algorithm to create polyphonic drum patterns of synthesised kick, snare, hi-hat and crash/clap type sounds (Fig. 2). These four sounds are mapped onto a single 16-step row to create a 4x16 drum pattern matrix as is evident in the figure. The operation of the instrument is intended to be as simple as possible. The user inputs a desired pattern into the pattern matrix and assigns this as the target pattern. New target patterns can then be generated by clicking on "Get New Pattern". This assigns the best candidate in the current population to the grid and reports its fitness in the number box. In the yellow number box a target fitness can be issued to the genetic algorithm and clicking "Accelerate!" will increase the speed of the algorithm until it reaches this target. Patterns that are close to perfect fitness now emerge from the algorithm, repeating and contrasting with the target pattern.

The instrument also attempts to provide some visual feedback on the evolutionary process using a type of force-directed graph. The fittest individual from each population is represented as a node with its fitness determining the size and distance in relation to the target individual.

3.2 Algorithm Details

3.2.1 Linear vs. Parallel Operation

As the previous section explains, the instrument is intended as a fully functional drum machine, and hence is polyphonic with the possibility of multiple sounds concurring in time. As seen in the literature, similarity studies are predominantly focused with examining monophonic patterns such as the single sound clave son. How we deal with the polyphonic implications in our research is outlined here.

Our first naive implementation of the genetic algorithm converts the 4x16 drum pattern matrix into a single "linear" 64 digit binary string. Evidently this is a simplistic "brute force" approach that does not explicitly take into account any musical or perceptual aspect of the application. Specifically it does not embed any knowledge about the constituent sounds and the separation between them in the genome string. Essentially, we're treating it as a single sound 64-step pattern, with crossover and mutation happening at the halfway point between the first 16 bits of the kick and snare pattern and the second 16 bits of the cymbals/crashes.

Another approach is to force some kind of logical separation between the different polyphonic timbres in the pattern. By assigning a separate instance of the genetic algorithm to each timbre an overall mean fitness can be derived from the individual outputs. This we implemented and labelled as "parallel" mode. Since the genome pattern length is now split from one single 64-bit string to four concurrent 8-bit strings, the time it takes for the genetic algorithms to reach the target fitness is considerably reduced. Some tweaking of the parameters is required to reduce this convergence time and maintain a healthy level of diversity. Setting the population size parameter to 30 and increasing the mutation rate parameter to between 20% to 40% has been found to work well in our experience. Next we turn our attention to the distance measures used to derive the fitness function of these genome patterns.

3.2.2 Hamming Distance

In the review of the state of the art, we referenced how Post and Toussaint have surveyed the effectiveness of the edit distance in determining rhythmic similarity between two binary patterns [9]. Recall that the edit distance allows for insertion deletion and substitution of symbols within the string. A simplification can be made if operations are restricted to substitutions only, and string lengths are equal (as is always the case in our representations) in which case it becomes the Hamming distance.

If a and b are two strings, a fitness function can be derived using the Hamming distance by counting the number of positions where the two strings match then dividing by the total string length (64 for the linear string and 8 for each string in the parallel implementation). This can be seen in the formula below.

$$F = \frac{1}{N} \sum_{n=1}^{n} a_n \oplus b_n \tag{1}$$

3.2.3 Swap Distance

The Hamming distance takes into account the correct positional scores between two patterns, but it does not give any indicator as to how different a pattern is in terms of the horizontal displacement. Intuitively one would think this horizontal displacement would reflect the important phenomenon of rhythmic syncopation. For example there

may exist two different patterns with equal distance but one pattern has more onsets aligned closer to the original pattern. Is there a measure that can give this "closer" pattern a higher score based on its horizontal distance? Indeed, the swap distance may be a suitable measure in this instance.

As alluded to previously, the swap distance assigns a score depending on the amount of swaps needed to convert one binary pattern from one to another. Computationally speaking, Díaz-Báñez et al. point out that actually performing the swaps to derive the score is expensive and redundant [13]. It is better to create a new vector for each of the patterns working with the offset distances of the onsets instead. For example the pattern {0, 0, 0, 1, 0, 1, 0, 0} would reduce to the offset vector {3, 5}. In the case of patterns with the same number of onsets the computation is straightforward: you simply sum the differences at each index of both vectors.

$$D = \sum_{n=1}^{N} |a_n - b_n| \tag{2}$$

The genetic algorithm can generate many possible string configurations and critically, strings with different number of onsets to the target string. This poses a problem in calculating the distance as the previous equation no longer applies: how to map one string optimally to the other? In fact, this issue has occurred in the similarity analysis of standard flamenco rhythms where the number of onsets frequently differ, as Guastavino et al. point out in [14].

To tackle this, Toussaint proposes an extension to the definition of the swap distance known as the directed-swap distance. It stipulates that a) every onset in the shorter string must receive at least one onset from the longer string and b) every onset in the longer string must go to some onset in the shorter string. Extending the genetic algorithm object for Max and Pd, we implemented an algorithm proposed by Colannino [15] for computing the distance in $O(n^2)$.

The algorithm essentially treats the problem as a minimum surjection between two sets. A weighted directed graph is constructed and the optimal distance between the two strings is extracted by gathering the shortest path, which was done using an implementation of Dijkstra's Algorithm in the Boost Graph library [16]

3.3 Evaluation Design

When evaluating this research and its resulting tools, our goals were to investigate:

1. The overall correlation of distance with perceived experience.
2. Comparing the impact of the Hamming distance versus the directed-swap distance.
3. Comparing the impact of the linear versus parallel string representation.
4. The more informal, subjective issue of the musical "interestingness" of the rhythmic patterns created.

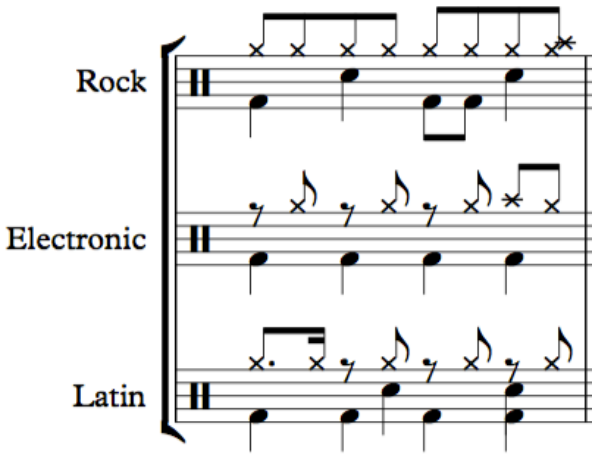

Figure 3. Target Patterns

We decided to conduct a simple listening survey to get listener feedback regarding on these aspects. The survey was web-based and unsupervised; participants were sent a link with instructions on how to complete it.

The listening portion of the survey was divided into two parts. The first part examined similarity ratings by presenting the user with the target pattern and the algorithmically generated patterns. Participants were then asked to rate the perceived similarity on a five-point Likert ranging from "Highly Dissimilar" to "Highly Simmilar".

The second part then examined the "interestingness" of generated patterns. To enable the participant to ascertain this, the patterns were arranged in a soundfile as TPx2, GPx2, TPx2, GPx2 (TP=Target Pattern, GP=Generated Pattern) i.e. a two-bar loop of the target pattern is followed by a two-bar loop of a generated pattern and the whole sequence is repeated. This choice of configuration was quite arbitrary, but it was reasoned that in order to get a sense of the interplay between the target pattern and the generated pattern it was necessary to repeat the sequence at least once.

Once again a five-point Likert scale graded the ratings, this time with labels ranging from "highly disinteresting" to "highly interesting". Regarding the subjective interpretation of "interestingness", we instructed the participants to consider how the target pattern and generated pattern "flows" from one to another, and how the generated pattern "develops" on the target pattern in terms of introducing stimulating variation.

Three target patterns were used for the purposes of the test: a standard straight 8 rock pattern, a four-on-the-floor electronic pattern and a son-based latin pattern (Fig. 3). Table 1 summarises all the variables under consideration for the evaluation. This resulted in a total of 72 WAV files with 3 fitness levels (inversely corresponding to the distance scores).

Question	Measure	String	Pattern	Fitness
Similarity	Hamming	Linear	Rock	Low
Interesting	Swap	Parallel	Latin	Med
			Elec	High

Table 1: Variable Summary

Twenty-two participants took part in the survey, mostly drawn from music students and researchers. All of the participants confirmed that they played an instrument, 7 of whom specified a percussive instrument. Eighteen out of the 22 participants reported the ability to read music. It took approximately 15 minutes to complete.

4. RESULTS

Before carrying out the statistical analysis the responses were summarised by computing the mode of the Likert scores for each stimulus. Two "interestingness" stimuli out of the total 72 (36 for similarity, 36 for interestingess) were removed due to high divergence of opinion (50% or more of the responses deviated by 2 or more Likert scale values from the mode).

4.1 Similarity Ratings

Our first task when looking at the data was to confirm whether inverse pattern distance and the fitness of the genetic algorithm correlates with the perceived similarity as determined by the participants.

Indeed the data seems to confirm this hypothesis. Table 1 presents the Spearman ranked correlation matrix of fitness, distance and the mode scores received for each stimulus. There is a clear, strong negative correlation coefficient (ρ = -0.71, $p < 0.05$) between the distance measure and the perceived similarity to the target. This correlation is not as clear with the fitness function, which can be attributed to the fact that fitness as a function of distance is evaluated differently for the two distance measures.

	Distance	Fitness	Score
Distance	1.0000000	-0.4145087	-0.7134277
Fitness	-0.4145087	1.0000000	0.4353168
Score	-0.7134277	0.4353168	1.0000000

Table 2: Overall Similarity Correlation Matrix

Fig. 4 shows the separated distance and fitness correlations against the mode scores for the Hamming and directed-swap distance measures. The fitness correlation values are 0.784 and 0.625 respectively and the distance correlation values are -0.784 and -0.716 respectively ($p < 0.05$). It can be seen that the Hamming distance has slightly better correlation.

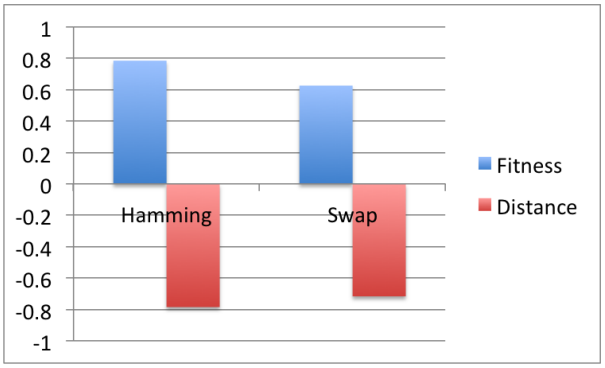

Figure 4. Distance Measure Comparison

Fig. 5 shows the separated distance and fitness correlations against the scores when we discriminate between linear and parallel pattern strings. The fitness correlation values are 0.525 and 0.480 respectively and the distance correlation values are -0.852 and -0.576 respectively ($p < 0.05$). The linear representation scheme appears to correlate better with human judgement.

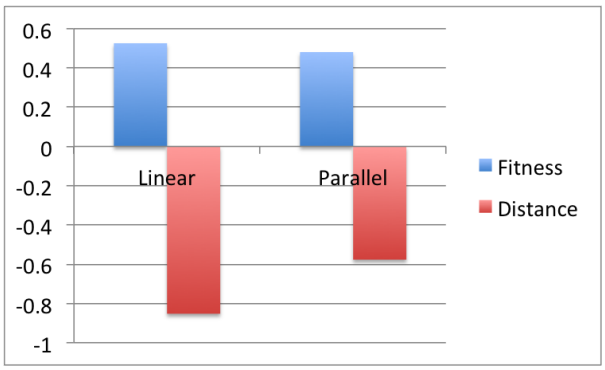

Figure 5. Representation Comparison

We can draw some tentative conclusions based on this data. Firstly, the strong correlation between the overall distance and similarity ratings suggest that even for polyphonic string representations of drum patterns, distances such as the Hamming and directed-swap are useful measures of perceived similarity. Secondly, splitting and recombining the bit strings by timbre, as carried out in the parallel operation, does not seem to offer improvement over the simplistic, long bit string representation. Finally, the more complex directed-swap algorithm, with its ability to capture the "horizontal" displacement between two drum patterns, does not appear to reflect an increase in similarity scores in our survey, confirming Toussaint's finding but also extending it to the case of polyphonic patterns.

4.2 Subjective "Interestingness" Evaluation

Table 4 presents the correlation matrix corresponding to the results of the users' impression regarding the "interestingness" of the patterns when heard in a sequence with the target. Curiously the distance and fitness correlation coefficients are both positive, despite the fact that distance is inversely proportional to the fitness of the genetic algo-

rithm but the p-values are so high (0.211 and 0.1887 respectively) this data is not reliable. It is impossible to draw some meaningful or significant conclusions based on the disparity and inconsistency across subjects.

	Distance	**Fitness**	**Score**
Distance	1.0000000	-0.5419222	0.2201162
Fitness	-0.5419222	1.0000000	0.2310225
Score	0.2201162	0.2310225	1.0000000

Table 3: "Interestingness" Correlation Matrix

Disregarding the difficulty in appraising musically subjective output, the reason for this problematic data is largely attributable to the way in which we considered the notion of "interestingness" and how the question was formulated. Asking the participants to rate two essentially diametrically opposing qualities - i.e. variation (related to dissimilarity) and repetition (related to similarity) was a flawed approach that caused confusion. This became immediately apparent from some of the user feedback at the end of each survey session. For example:

> "I noticed that I somehow prefer rhythms that are a natural evolution of the previous pattern, instead of being totally different. But if the similarity with the previous pattern is too high, the result is still uninteresting to me, because the resulting pattern is too predictable and loses every appeal."

To complete the survey then, users often reverted to their own rules to determine their ratings, as evident from these comments:

> " 'Interestingness' was hard for me to evaluate. In the end I rated with a 'good' interesting 'bad' interesting system: if it's weird but i like it, it's in the first case, if not, in the second case."

> ".. There are some times on experiment 2 that the new rhythm might not be interesting for a complete section but that might be useful as a bridge or as a temporal loop marking the end of a section."

Another participant made the point that the pattern sequence may have forced some "expectation" regarding the concept of "interestingness":

> "... I have also the feeling that having the pattern repeated (ie AB twice), and therefore, having to come back to A again after having been in B, conditions very much the results."

Despite the apparent issues with our method of evaluation, the data and informal feedback does suggest that the genetic algorithm creates "interesting" musical output. Nineteen stimuli out of the 34 analysed registered a mode score value higher than 4 as seen by the 56% green positive region in Fig. 6 (there were two responses for 'Strongly Disagree', but these were the two that were disregarded due to high divergence of opinion). The task ahead is to review the evaluation strategy in order to quantify and explain this aspect in a more coherent and predictable manner.

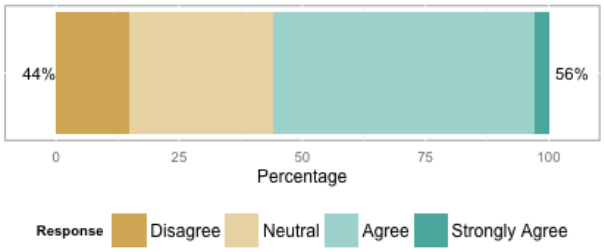

Figure 6. Distribution of Responses for "Interestingness"

5. CONCLUSIONS AND FUTURE WORK

In this paper we presented a way of generating drum patterns automatically using genetic algorithms. Rather than rely on the commonly used interactive fitness function, our method was shown to use the notion of a "target pattern", with fitness derived from the distance of the generated patterns to the target.

Following a review of various approaches to establishing the distance between rhythms as present in the literature, we demonstrated the implementation and incorporation of two such measures - the Hamming distance and the directed-swap distance - into a genetic algorithm instrument for polyphonic drum pattern creation. We believe this paper contributes the first integration of perceptual research in rhythmic pattern generation with genetic algorithms.

To evaluate the research carried out, we conducted a listening survey to determine participants reaction to the generated patterns in terms of the similarity and "interestingess" related to the target pattern. It was shown that the distance and thus fitness correlates strongly with user perception in terms of similarity. Crucially we showed that the Hamming distance alone is a worthwhile quantifier of rhythmic similarity even in the case of polyphonic patterns. Our approach to gauging users' response to the concept of "interestingness" however, needs review and presents a complex challenge for future work.

Links

Code is available to download at:-
`http://www.github.com/carthach/GenDrum`

Acknowledgments

This research has been partially supported by the EU-funded GiantSteps project (FP7-ICT-2013-10 Grant agreement nr 610591). [2]

[2] http://www.giantsteps-project.eu/

6. REFERENCES

[1] J. A. Biles, "GenJam : A Genetic Algorithm for Generating Jazz Solos," in *International Computer Music Conference*, 1994, pp. 131 – 137.

[2] E. R. Miranda and A. J. Biles, *Evolutionary Computer Music*. Springer-Verlag New York, Inc., 2007.

[3] J. Biles, "Autonomous GenJam: eliminating the fitness bottleneck by eliminating fitness," *Proceedings of the 2001 Genetic and Evolutionary Computation Conference*, 2001. [Online]. Available: http://www.ist.rit.edu/~jab/GECCO01/

[4] A. Eigenfeldt, "Kinetic Engine: Toward an Intelligent Improvising Instrument," *Proceedings of the Sound and Music Computing Conference*, pp. 97–100, 2006.

[5] C. O. Nuanáin and L. O. Sullivan, "Real-time Algorithmic Composition with a Tabletop Musical Interface - A First Prototype and Performance," in *AM '14 Proceedings of the 9th Audio Mostly: A Conference on Interaction With Sound*, 2014.

[6] G. Bernardes, "Style Emulation of Drum Patterns by Means of Evolutionary Methods and Statistical Analysis," *Proceedings of the Sound and Music Computing Conference*, pp. 1–4, 2010. [Online]. Available: http://smcnetwork.org/files/proceedings/2010/26.pdf

[7] D. Horowitz, "Generating Rhythms with Genetic Algorithms," in *The Twelfth National Conference on Artificial Intelligence*, 2004.

[8] M. a. Kaliakatsos-Papakostas, A. Floros, and M. N. Vrahatis, "evoDrummer: Deriving rhythmic patterns through interactive genetic algorithms," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 7834 LNCS, 2013, pp. 25–36.

[9] O. Post and G. Toussaint, "The Edit Distance as a Measure of Perceived Rhythmic Similarity," *Empirical Musicology Review*, vol. 6, no. 3, pp. 164–179, 2011. [Online]. Available: https://kb.osu.edu/dspace/handle/1811/52811

[10] Y. Grandvalet and D. Eck, "A Generative Model for Rhythms," *Neural Information Processing Systems, Workshop on Brain, Music and Cognition*, pp. 1–8, 2007.

[11] M. Puckette, "Pure Data : another integrated computer music environment," *Proceedings, Second Intercollege Computer Music Concerts*, pp. 37–41, 1997.

[12] T. Grill, "C++ layer for Pure Data & Max/MSP externals," in *2nd International Linux Audio Conference*, 2004.

[13] J. Díaz-Báñez and G. Farigu, "El compás flamenco: a phylogenetic analysis," *Proceedings of BRIDGES: Mathematical Connections in Art, Music and Science*, 2004. [Online]. Available: http://archive.bridgesmathart.org/2004/bridges2004-61.html

[14] C. Guastavino, G. Toussaint, F. Gómez, F. Marandola, and R. Absar, "Rhythmic similarity in Flamenco music: Comparing psychological and mathematical measures," *Proceedings of the fourth Conference on Interdisciplinary Musicology*, p. 76, 2008. [Online]. Available: http://mil.mcgill.ca/docs/GuastavinoToussaintCIM2008.pdfhttp://cim08.web.auth.gr/cim08_abstracts/CIM08AbstractsProceedings.pdf#page=76

[15] J. Colannino and G. Toussaint, "An algorithm for computing the restriction scaffold assignment problem in computational biology," *Information Processing Letters*, vol. 95, pp. 466–471, 2005.

[16] J. G. Siek, L.-Q. Lee, and A. Lumsdaine, *The Boost Graph Library*, 2002. [Online]. Available: http://www.informit.com/store/product.aspx?isbn=0201729148