

TIME AFTER TIME : SHORT-CIRCUITING THE EMOTIONAL DISTANCE BETWEEN ALGORITHM AND HUMAN IMPROVISORS

David Plans Casal
University of East Anglia
School of Music

ABSTRACT

In the experience of musical improvisation with an artificial improviser, there is a difference. As every musical intention is given by the human player to the live algorithm driving the artificial player (and viceversa), meaning can never be fully conveyed but for the opposition of other, differing musical intentions. Variety is the essential data from which one can build successful algorithmic designs, as without it, the data set is flat and useless. However, neither algorithm nor human can for now, successfully convey the emotional consequence of this difference to each other, and thus the human player is left to invest into and create a prosthetic emotional relationship on their own. This paper will summarise the previous design and creation of such an algorithm, outline the emotional problem space engendered by the author's interaction with it over a period of two years' musical performances, and explicate a brute-force solution designed to foreshorten the emotional distance between algorithm and human.

1. INTRODUCTION

Between 2005 and 2007, inspired by Peter Todd and Greg Werner's seminal paper, "Frankensteinian methods for evolutionary music composition" [14], I begun work, together with Davide Morelli, on a system that would use genetic co-evolution in the building of an artificial improviser. The background research and building process for this algorithm, whom we eventually called Frank (as an homage to Todd and Werner), were outlined in earlier ICMC proceedings [5]. In short, while Todd and Werner had implemented co-evolution using symbolic (MIDI) representations of musical knowledge, we used an existing technology intended for music retrieval (MPEG7 matching) as implemented by Michael Casey [6], and existing data clustering algorithms (k-means) to arrive at a musical gesture representation using the spectral domain. We then used this matching technique in combination with genetic co-evolution to create a system that could interact with a human player, by responding to her audio queries by evolving solutions to those and proposing them back as audio. An outline of this system is given in Figure 1. Over the last two years, both Davide Morelli and the author of this paper have given a number of performances using this system [4]. In live improvisation, my experiences performing with Frank have left me wanting to

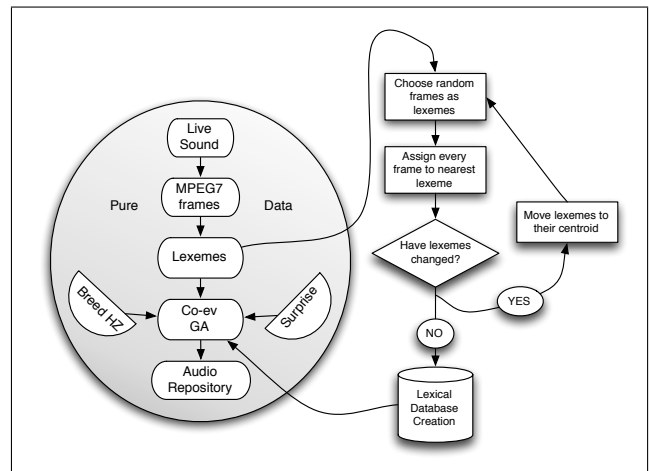


Figure 1. An overview of the Frank algorithm.

resolve a problem: while playing over a period of time, one comes to realise that while the algorithm is very efficient (and sometimes surprisingly so), at finding very musical responses to one's playing, it eventually becomes obvious that the effectiveness of the search for a match, while semi-organic (co-evolution produces good amounts of variety and surprise), is non-emotional and completely ambivalent. One cannot really feel empathy towards it. More explicitly, the problem is that a feeling of difference (understood here in the Stiegler interpretation of Derrida's concept) emerges in the player, whereby one wishes there was a more implicit, emotional connection between gestures proposed by the player, and the way the responses evolve in the algorithm. As an example, the capacity to sometimes arrive at the same place, at the same time, in the kind of spontaneous synchronicity that human players can have, is almost impossible with this algorithm. This is because it continuously moves away from the perfect solution at the time, to the next perfect solution, therefore ignoring possibly interesting, sustained direction: a frustrating and emotionally unsatisfying situation. Below, I propose a theoretical background to best understand the inherent problem, and a rough-and-ready solution to it, that nonetheless has satisfied the author musically.

2. TIME IS NOT A PHENOMENON

The main cause of *différance* between human and algorithm is the facile, but brutally disconnected relationship of the algorithm to time itself. Algorithms think of time computationally, connecting the dots between their typical UNIX-like genesis (January 1, 1971) and the current second. Of course, ‘counting’ in seconds or milliseconds is not nearly granular enough for a human musician. It is enough to watch the interaction between the players in a good, long-running string quartet, or the apparently spontaneous synchronisation between long collaborations in the improvised music world, such as the Redgate brothers, or perhaps members of AMM, to realise that no computational measure of distance between musical events is ever going to fully encapsulate the musical experience. Thus, a computer algorithm, made with layer upon layer of artificial (and delayed, or deferred) structure (from clock time, to low-level memory access, to high-level programming languages), has no real hope of coming close to a temporal relationship with a human that approaches the level of emotional relationship between human players. At least, not until we achieve feasible quantum computation (and even here using thermal time, not the quantum external clock), do away with time as a phenomena (and measure stick) for computational processes, and begin to emulate more directly what the human brain does when exhibiting musical behaviour. That is, exhibiting musical knowledge and interconnections at incredible speed, almost unconsciously. All we can do in the meantime is approximate, and build rough prostheses that through smoke and mirrors facilitate emotional relationships of some meaning with our algorithms. In his analysis of the space of interactive music, Jonathan Impett [9, 29] clarifies the idea that the “grip-slip relationship between vision and realisation” is crystallised in improvisation. The temporal difference between vision and realisation (imagining a piece of music and making it), which is divisible in composition, “...becomes the very object of aesthetic attention - tracked in real time.” It is this temporal difference, then, our tracking of it, and the *différance* engendered by the algorithm’s ability to track it in insightful and not just effective ways, that we need to model, resolve, or do away with altogether if we cannot model it at all.

2.1. Doing Away With Time Altogether

Time, in general relativity, is not absolute. There is no great cosmic clock. Instead, the observer ‘invokes’ the passage of time by their very observance. This is a useful definition of time for music-making, as the frames of reference between one observer and the other can be matched to a score and a digital clock (that by its very definition is just an aid to the perception of time). In quantum mechanics, however, time is a phenomenon that exists wholly outside of the universe, keeping track of every change in the microworld. This clock records time absolutely, in precisely the way Einstein said it could not. Julian Barbour’s work [3] in Machian dynamics led him to assert

that time simply must not exist in a quantum world; essentially, that it is not a phenomenon at all. Carlo Rovelli and his Quantum Gravity team have achieved the folding of multiple quantum events into a single event that can be described without any reference to time [8], where the electron and the measuring device (quantum mechanics’ externality) are described by a single wave function; in their case, time simply turns into a set of correlations between points or things that can be observed in space. In fact, Rovelli points out that one can simply describe the correlation in the observables as an adequate substitution for a description of which measurement happens *first in time*.

Thus, Rovelli points out that the dynamics of the Universe could be described as a network of correlations, rather than the evolution from big bang through the present day in time. This does not separate him completely from Relativity; in fact, he points out that in his opinion “...the result presented here reinforces the idea that quantum mechanics admits a very simple and straightforward generalization which is fully consistent with general relativity. And therefore that the contradictions between quantum theory and general relativity might be only apparent.” [8, pp. 16] He and mathematician Alain Connes have worked to try to define why our perception of time exists at all, in a reality where time does not. They argue that time emerges as a statistical effect, much in the same way temperature is a measure of the average amount of activity between groups of molecules. Because we cannot keep track of all the information generated by the different behaviours in the Universe as singular events, we perceive the average movement in a *statistical* way, and come up with an ongoing value, our perception of time passing. And so, “...it is not reality that has a time flow, it is our very approximate knowledge of reality that has a time flow. Time is the effect of our ignorance.” He calls this the thermal time hypothesis.

I find this theory the most feasible solution to the *différance* engendered by our relationship with an artificial algorithm. Since our perception of time is simply a statistical emergence notion, and it is possible to more accurately (and without referring to time at all) describe a world’s dynamics through the interconnectedness of the activity nodes within it (and their correlations), perhaps the design of this algorithm must simply focus on correlation, instead of defining the variety contained in the music by trying to describe its evolution through time? Would this help, in trying to achieve a more interpersonal relationship between artificial and human players? I believe it could, and that if successful, would eliminate the need for Impett’s real-time tracking. We would instead be looking at a single quantum event, containing the whole of the musical cultural heritage of the player, accessible as a random buffer without a reference to time, or its role in the division between vision and realisation.

Below, I will outline an attempt at a model for this solution computationally. But before I can do so, another important issue must be addressed: while time is inadequate both as a functional variable for the design of a musical algorithm (especially in improvisation) and as a perceptual measure of emotional value for the human player, we nonetheless understand our lives as an evolution through time. Remembrance and forgetting are both consequences of this. How can we then relate to a system which involves no invocation of time at all? We must have a story to refer to: some kind of map. To build this map while building a correlation-based solution, we must investigate the need for timelines more closely.

2.2. This Temporary Arrangement

We think of our lives as timelines. We refer back to points within this timeline, sometimes through image memory, sometimes auditory, olfactory, and sometimes simply in a largely unconscious, embodied way, such as an improviser will quote harmonic changes without specific reference or conscious intention, that nonetheless refer directly to a different improviser's (or composer) work.

This temporal, and temporary arrangement, our relationship with time and its fragile association with our memory, is cinematic in nature. That is, our relationship with our own story, our biographical timeline, both of the effect other things and people have had on us and the actions we ourselves have carried out throughout our lives, correspond to a granular, random buffer of associations and queued event-memories: Stiegler's cinematic consciousness [13]. But also, and more significantly for the purposes of discussing the origin of knowledge in an improvised act, which by its very nature lives for and by the present, we must look at the Socratic notion that all knowledge is remembrance. In Stiegler's explication of Socrates' recollection of the myth of Persephone to Meno [13, 95–100], whereby Socrates arrives at the question of being, or “What is being, the knowledge of being, that is, a true discourse?”, he arrives at an important point, which has influenced greatly the design of my improvisational methodology (and of the algorithms I design to help me). That is, that all knowledge is in fact the recollection of originary material [10]; that it is in fact, Kant's transcendental knowledge, preceding (and preceded) experience, which we are continually in the process of re-discovering, even when that discovery is new to every other soul currently alive. This is the biblical notion that “what has been is what will be, and what has been done is what will be done; there is nothing new under the sun.” (Ecclesiastes 1:9-14 NIV)

This view has led me to think of an algorithmic design in which the artificial improviser acts as a mediator between the implicit enquiry contained in improvisation (with an algorithm or a human), and the body of transcendental (musical) knowledge contained both in the

recorded output of every musical epigenesis, as well as the epiphylogenetical space contained in one's (the musician's) own practice. In fact, I need a map of the unconscious processes that inform musical improvisation that is made simply not of temporal but of significant connections. The notion of significance and musical meaning as understood in my algorithmic design therefore needs to be addressed. The most apt treatment of musical meaning and emotion (from which we most often derive significance) I can find to root my own understanding of it is Sloboda's contextualisation of ‘peak experiences’, or strong motive responses to music that go beyond evoking basic emotional responses such as sadness, happiness, etc. [12, pp.384-385]. Data that Sloboda points to (Gabrielsson, 2002) suggests that ‘...strong experiences arise in situations which often contain complex biographical and non-musical factors.’, and that strong emotive reactions triggered by music are not always triggered by the inherent structures in musical content. For the purposes of the design of this algorithm, I have always considered that ‘meaningful’ musical interaction between a human and the algorithm had to be based on peak experience, and not simple or basic emotions. As it became clear to me that the only feasible representation of time was a temperature-like degree of ‘timeness’, the only points in this kind of time worth noting as significant events had to be strong enough experiences to relate firmly to one another. Factually, Sloboda's definition of musical engagement as dependent on ‘if and how the listener believes that there is an intentional agent behind the music’ means essentially that, for this belief to be engendered by an algorithm in a human, the algorithm must directly provoke a peak experience in the human.

So, this map of significant, meaningful musical connections must be based on strong experiences, and must work on top of (mediating with) the original algorithm: Frank.

2.2.1. Wherefrom Structural Deferral

Building such a map is conceptually difficult, as we are trying to build a map based on significance, of events prompted by auto-noetic [15] memory: essentially a map of shifting sands. This is again, because of *différance*. According to Stiegler, in our effort to grasp the flux of our own consciousness, we encounter “...a structural deferral or [*italics*]différance because what is to be grasped (the flux itself) would always be already gone by at the moment of grasping.” [7] This temporal divide is ‘sutured’ by Husserl by shifting focus from the flux itself to the temporal object, and this in turn allows Stiegler the advantage of being able to focus on the technical conditions of the temporal object, thus introducing technicity, or what he calls tertiary memory. I venture that Stiegler's tertiary memory is homologous to Tulving's auto-noetic memory, in that both entail the storage and retrieval of the past in forms that can be “...assumed by consciousness in the present.”

I believe it is the capability we have as human beings to assume a collective past (even though we have not lived it) that allows us to experience the present as a large cinematic buffer of protentions and retentions; what Grand Granel calls a “large now”. Heidegger’s ‘thrownness’ perfectly illustrates the point-of-view through which we access this buffer. This meta-algorithm must be designed to be, then, no more (and no less) than a technicity mirror, in which the technicity of musical space is used to form and inform the world it knows, and which it presents back, in the form of an epiphylogenetic [13, 140] canvas: a large now.

However, I am in agreement with Hansen’s view [7] that Stiegler’s account of the technical contamination of retention (the effect of the prior audition of an object on a subsequent one belonging to tertiary, and not primary, memory) misses an important point: our hearing of a melody for the n th time is not simply a mere episodic recall of the melody as a technical object. It is, rather, impacted by “our largely unconscious, deeply embodied, and singular response to particular aspects of it.” [7, 601]

This embodiment of the musical technical object (away from the perhaps wrongly attributed cinematicism of Stiegler’s definition, but imbued with its sense of tertiary, autozoetic response), is in effect a dissection of our concept of time (it doesn’t exist, it is an imaginary buffer) into accessible noemata that can be re-cycled to form useful musical objects. These noemata, composed of lexical and morphological pieces, which our algorithm must suffer to evolve over millions of generations if it is to become a useful mirror for our musical unconscious, are in fact our map through time without time. They are our epiphylogenetic story, and therefore perhaps a good enough base for the improvement of the original algorithm.

Frank was originally engineered to immediately assume a currently happening musical event (a cluster of lexemes derived from MPEG7-described FFT frames) to be always and immediately ready for ingestion into the cultural pool; in this way, all musical events became always already known: we applied no symbolic filter to incoming events. The evolution of the algorithm’s cultural heritage from moment 0 is inevitably, always already part of the cultural heritage between improviser and algorithm. In this sense, the immediateness I ask of the algorithm is not an aesthetic choice, but simply an inevitable side-effect. My aesthetic choice lies in the non-use of symbolic processes and the absence of time as an evolutive measure. However, there is a problem. The genesis of technics is the genesis of temporality as such. As soon as I introduce the first musical event into the genetic pool in the algorithm, there is a way to track ‘back’ to it. Temporality inevitably ensues, and différance begins its inexorable emergence. The algorithm becomes cold and ineffectual in its search through time, and I notice and become estranged and emotionally detached from it. It, in turn, has no way of knowing I feel

that way.

To succeed in fighting this problem, I had to find a compositional/improvisational methodology that invoked time not as “clock time”, or the epistemological “a priori” time that an archival memory consists of, but as what it is: a figment of our imagination. In effect, I must observe the thermal time hypothesis, but design a metasystem that will nonetheless manage the temporal correlations between musical event nodes, without literally referring to (looking up) time *at all*.

3. BRUTE-FORCE EMOTIONAL RENDEZ-VOUS

“Technology adds nothing to art. Two thousand years ago, I could tell you a story, and at any point during the story I could stop, and ask, Now do you want the hero to be kidnapped, or not? But that would, of course, have ruined the story. Part of the experience of being entertained is sitting back and plugging into someone else’s vision.”, Penn Jillette, Interview in WIRED magazine, 1993 - US magician & showman (1955 -)

It is an inherent part of my ambition that I must be able to not just use algorithmic design to aid in amplifying my vision during the act of improvisation; but that I must be able to actually plug into the algorithm’s own vision, emotionally. This informs the design of the meta-algorithm below, in a way a band-aid to Frank’s design, to get the human improviser as close as possible to the inner life of the algorithm. This means that I have purposely chosen the simplest, and not the best technically, solution to my technical ambition. I am simply more interested in rough musical result, than computing efficiency, and thus I am willing to ignore computational design inefficiencies and simply throw more computing power at it if needed. As an example, the choice of binary tree I make below is less efficient in terms of CPU use for processing the acoustic lexeme arrays than a traditional binary search tree would be.

3.0.2. The Splay Tree

In the original algorithm, Frank, we gave our system a division of tasks: the male population in our genetic algorithm produces many answers to the musical space question (an incoming query by way of real-time audio, such as a piano chord, in the form of a stack of MPEG7-described FFT frames). The female population criticises those answers, isolates winners, and breeds with them. Just as in Todd and Werner’s idea, this process is about generating answers, testing those against some criteria and repeating the process. Our objective was for those criteria (the fitness function) to evolve in real-time, and not be set by the system maker (which would move us towards the design of a composition, not an improvisational, system). We saw that using MPEG7 vectors (provided by

Casey’s Soundspotter methods), essentially frames in the musical spectra of ongoing real-time audio derived from FFT analysis, could provide us both with an ongoing influence and set of criteria, but also with a genotypical unit with which to start the process of evolution. We could assign a number of incoming MPEG7 frames to be our female genotype, which would trigger imitation (from the males), and let co-evolution take over from there. The ability of co-evolution to generate synchronic diversity through the process of sexual selection (speciation) would then save our system from eradicating diversity and reaching a ‘perfect’ solution, which would be musically uninteresting. To reduce the data set (the FFT frames came in 86-dimensional space), we clustered MPEG7 frames using the k-means algorithm, and the center of each cluster became one of our lexemes: the genotype. These lexemes are the data structures on top of which I built the meta-algorithm I describe below.

In order to find a solution to the problem outlined above, we must find a way to represent computationally, the correlations between nodes of musical interest. Since we already have a database of clustered information through the k-means cluster achieved by Frank, a possible way to represent the relationship between each cluster center might be a graph, in which each node would have an ‘importance’ value. However, this does not give us a flexible way to search the graph, and does not allow for the interconnectedness between nodes to be searched efficiently. Binary search trees can perhaps offer a solution.

I needed a binary search tree that would gather its values from the clustered centers of the lexemes database, then proceeded to construct a search algorithm to sort the most accessed values in the tree nearer to the root, therefore allowing for a running order on what musical nodes or lexical gestures were accessed last (and for these to be accessed quickest). There is, in fact, a type of binary search tree that does just that. The Splay Tree, first used and designed by Daniel Sleator [11]. A further advantage of the Splay Tree is that, while more traditional binary search trees such as balanced or optimum search trees are “...designed to reduce the worst-case time per operation” [11, pp.653], splay trees use what Sleator calls “amortised” time to work on a *sequence* of operations, by using a self-adjusting data structure. By allowing the structure to exist in an arbitrary state, and then applying a simple restructuring of the tree to improve the efficiency of future operations, splay trees achieve a very efficient path-compression of incoming information in very little computational space. They can require more processing power, and could be a problem in real-time operations, but my experiments so far have added an average of 7% processing overhead to the existing CPU power used by Frank, and therefore this is not a concern.

Figure 2 gives a graphical overview of the Splay Tree as used within this design.

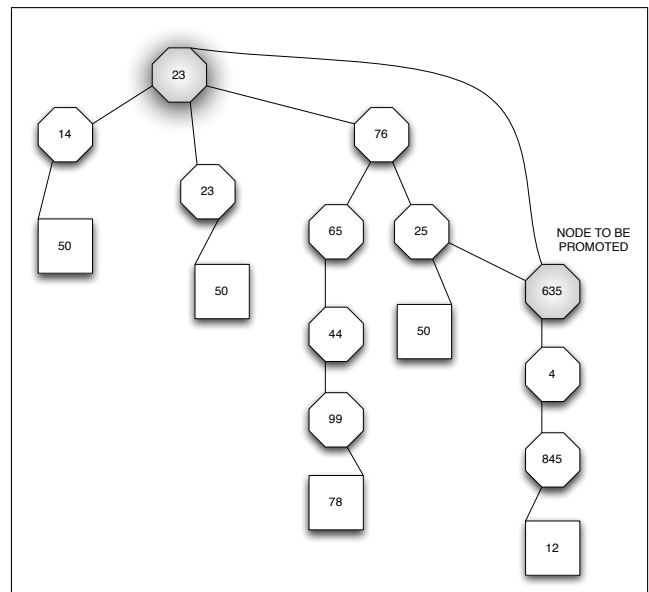


Figure 2. An acoustic lexeme Splay Tree with terminal nodes as squares.

3.0.3. Meta-Algorithm Implementation

Because Frank is designed to publicise its last winning cluster of lexemes as a result of its genetic co-evolution computations, each generation will output its winning cluster of lexemes as a numerical ID that can be input into the Splay Tree. Since Frank was written the C++ Flex framework for Puredata (and Max/MSP), and there is a Python layer that can easily access Flex objects from within Puredata, I used a Python implementation of the Splay Tree freely available as Open Source code [2]. From within this implementation, I can simply pass winning lexemes as well as lexemes in the query form (human) to the python function as follows:

```
def insert(self, acousticLexeme):
    if (self.root == None):
        self.root = Node(acousticLexeme)
        return

    self.splay(acousticLexeme)
    if self.root.acousticLexeme == acousticLexeme:
        return

    n = Node(acousticLexeme)
    if acousticLexeme < self.root.acousticLexeme:
        n.left = self.root.left
        n.right = self.root
        self.root.left = None
    else:
        n.right = self.root.right
        n.left = self.root
        self.root.right = None
    self.root = n
```

Note that if the lexeme has been already entered into the tree, we do nothing. This is on purpose, to signify that the musical knowledge in the tree as represented by connections between musical nodes should always already be

there: we are only trying to rediscover and quicken the connections between them.

Now that incoming lexemes are being inserted into the tree, I can then use the remove, findMin and findMax functions to construct the rest of the solution.

If the most recent match against the database (as incoming from the human player) matches one of the lexemes currently in the tree (the tree only keeps around 10 to 15 nodes at a maximum), that node is found and therefore promoted to the near the root of the tree. In addition, it is then inserted into the male population of the genetic co-evolution process, causing a small event horizon in the relationship between this population and the ‘critic’ population (female), as it represents a very strong phenotypical expression (in effect, a dominant gene). Within Frank, I then give the female population the ability to insert a highly successful individual from the immediately following generation (i.e. 70% success) back into the tree. This is simply reciprocal, but highly complex process (because of the incoming variation produced by males and subsequent female criticisms), produces an intense self-referencing between Frank and this meta-algorithm, in which the human player is much more closely involved than the original; it is a sort of brute-force rendez-vous between the inner process of co-evolution and the human player’s process of continuous lexeme autooensis. However, it doesn’t give the human player (or the algorithm) a chance to let each other be aware of possible close emotional relationships: it still relies on emergent relationships. To this effect, I introduced another way in which both players could trigger cataclysmic musical events. Frank was designed to be fairly self-sufficient, but we gave it trap doors: a couple of variables called BreedHZ and Surprise. This alters the frequency at which generations breed and the extent to which they will attempt to surprise the female population: just how wild the mutation will be. These two variables are easily accessible through messages passed to Frank, so I pre-empted two cases, using the findMin and findMax functions, in which if the incoming lexeme (from either the human, male or female population of the algorithm) matches the maximum depth in the tree, we accelerate breeding close to the fastest it could possibly go. And, if the incoming lexeme matches the minimum depth of the tree (closest to root without matching it), we set surprise close to maximum. This essentially means that, if the human and algorithm, through the process of splaying, matching and re-inserting alpha individuals, were to come really close to each other lexically, the evolution process would experience the equivalent of a revolution: it speeds up evolution in excitement (if findMax), and tries to be more surprising if the human is catching up (if findMin). Finally, to ensure that the meta (or mediating) algorithm doesn’t get stuck in a ‘perfect’ organisation of itself, if the incoming Cluster ID is exactly that of the root of the tree, we remove it.

Figure 3 gives an overview of the whole system (with the Lexeme database creation process omitted this time),

with the splay tree algorithm as a mediator between Frank and the human improviser.

Input: Sets of Lexemes as Queries to the Tree

Output: Promoted Lexemes into Male Population, and BreedHZ/Surprise values

```
foreach Cluster ID do
  while clusterID != Root do
    if clusterID = x then
      Promote(Splay) x;
      Insert x into malePopulation;
    end
    if NodeMatchDepth = findMax then
      Set BreedHZ = 90;
    end
    if NodeMatchDepth = findMin then
      Set Surprise = 90;
    end
    remove ClusterID;
  end
end
```

Algorithm 1: The Splay Tree Meta Algorithm

This, when represented visually through the aid of a programmable touch screen such as the Lemur [1], which can respond to OSC messages given by Puredata, and accurately represent the shape of the tree in real-time, can give the human player a very quick, graphical notion of how close the algorithm and her are in terms of the material they are referencing, and whether they are close to experiencing some kind of synchronicity. In fact, it can also show (from experience) how ‘excited’ either player is about the *possibility* of such a synchronicity, as human players tend to listen more deeply and stay motivically stationary when they feel they are engaged in a similar, synchronous musical idea with other improvising players. This displays in the tree in the form of rapid re-organisation, and then stasis. Because the Lemur can send OSC messages back to PD, it is possible to let the human player *manually* promote a node up the tree, thus let the algorithm know that it is excited about a particular gesture.

It is a brute solution to the problem of representing a process which is invariably contained in the passage of time (autooensis, and the pushing forward and waiting between improvisors), without using time as a reference. We can display, use, and subvert this cinematic buffer of musical gestures and the possibly synchronous events between players simply in terms of emotional closeness (excitement at the possibility of a musically interesting, ensemble gesture).

More importantly, however, this solution catalyses the existing genetic co-evolution and MPEG7 matching algorithm into a much more responsive system. Using it in performance, I have experienced much less detachment from its process and a far closer, human-like error and closeness, arriving together at synchronous events without meaning to. This is, in effect, a simple but effective solution to the difference problem posed at the beginning

of this paper. It allows the human (or at least, the author) to have a better, closer, and emotionally more accessible relationship with the algorithm.

4. REFERENCES

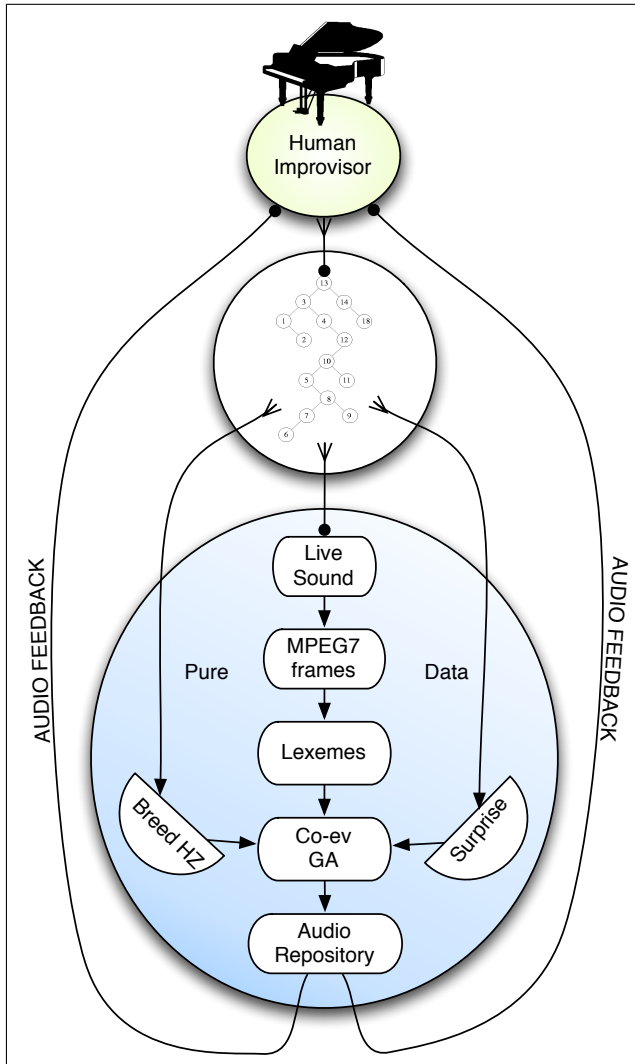


Figure 3. An overview of the original system with addition of Splay Tree algorithm.

- [1] The lemur. http://www.jazzmutant.com/lemur_overview.php.
- [2] Pysplay - splay tree implementation for python. <http://www.koders.com/>.
- [3] BARBOUR, J., FOSTER, B., AND MURCHADHA, N. O. Relativity without relativity. *Classical and Quantum Gravity* 19 (2002), 3217.
- [4] CASAL, D. P. David plans casal performances. <http://www.davidcasal.com/research>, 2005, 2006, 2007.
- [5] CASAL, D. P., AND MORELLI, D. Remembering the future : and overview of co-evolution in musical improvisation. In *International Computer Music Conference* (August 2007), vol. 1, The International Computer Music Association, pp. 200–206.
- [6] CASEY, M. Acoustic lexemes for organizing internet audio. *Contemporary Music Review* (2005).
- [7] HANSEN, M. The time of affect, or bearing witness to life. *Critical Inquiry* 30, 3 (2004).
- [8] HELLMANN, F., MONDRAGON, M., PEREZ, A., AND ROVELLI, C. Multiple-event probability in general-relativistic quantum mechanics, 2006.
- [9] IMPETT, J. Situating the invention in interactive music. *Organised Sound* 5, 01 (2001), 27–34.
- [10] PLATO. *The Collected Dialogues*, vol. LXX of *Bollingen*. Princeton University Press, 1961.
- [11] SLEATOR, D. D., AND TARJAN, R. E. Self-adjusting binary search trees. *J. ACM* 32, 3 (1985), 652–686.
- [12] SLOBODA, J. *Exploring the musical mind*. Oxford University Press, 2005.
- [13] STIEGLER, B. *Technics and Time*. Stanford University Press, 1998.
- [14] TODD, P., AND WERNER, G. Frankensteinian methods for evolutionary music composition. 313–339.
- [15] TULVING, E. *Elements of Episodic Memory* (*Oxford Psychology*). Oxford University Press, USA, 1983-01-01 1983.