Colin Sullivan
December 14, 2010

An Exploration of Algorithmic Composition via the Fibonacci Sequence

The attributes and implications of the simple and naturally-occurring pattern known as the Fibonacci sequence are especially intriguing to me and I have always been curious as to how this pattern can be utilized in art. My interest in music and technology naturally leads me to desire to understand how this sequence can be applied to sound and music. It is for this reason that I have attempted to utilize the Fibonacci sequence in a musical composition.

I initially experimented with the technical aspects of creating such a piece and developed a prototypical version of this composition. More recently, I have spent much more time on the theory behind my piece, researching what is possible with an algorithmic composition of this nature and understanding how the piece can mature in the future. My preliminary research into the history of algorithmic composition provided me with insight into the trajectory of this field and how it has changed over time. Before delving into the details of my composition, I will discuss a few highlights from the history of algorithmic composition that I found interesting.

The earliest account of algorithmic composition that I encountered was engraved on Babylonian clay tablets dating to 13th century B.C. which are reported to "describe algorithms for harmonization" (Baggi). It was at this point that I realized the magnitude of this field, and moved forward with a much expanded view of the history of algorithmic music. Guido of Arezzo (~991 - ~1033), who was a significant contributor to the development of musical notation, invented a method for converting text into melodic phrases where "Letters, syllables and components of a verse are mapped on tone pitches and melodic phrases (neumes), whereas groups of neumes are separated by caesurae [audible pause, period]" (Nierhaus 21). I found that the idea of using a human generated input to an algorithm is a prominent one throughout the future of algorithmic composition.

An interesting development that occurred a bit later (around 1300) was Philippe de Vitry's (1291 - 1361) experiments in "isorhythm". Isorhythm is where the placement of a fixed pattern of pitches is decided by a repeating rhythmic pattern. This method of systematic composition is interesting to me because of the constraints placed on the composer when implementing the isorhythm. In a way, this method is similar to the way I have structured a melody utilizing the Fibonacci number sequence in my piece. As will be discussed later, I used a fixed rhythmic pattern and mapped the (algorithmically determined) pitches into it.

It is also interesting to find the various contexts in which developments in algorithmic composition progressed. Giovanni Andrea Bontempi (1624 - 1705) developed a wheel with various layers that allows for constructing mixolydian mode voices in a composition. This wheel was designed by Bontempi with the intent that "one thoroughly ignorant of the art of music can begin to compose" (Cope 6). He reportedly also goes on to describe “how to calculate rhythms and cadences for more pleasing results” (Cope 6). I find it extremely interesting that Bontempi would develop a system to enable uneducated people to compose music. A similar example around the same time was developed by Kircher (1602 - 1680). Kircher created a system for musical composition via algorithms that required table lookups for pitch and rhythmic values in order to create a melody. Kircher also provided methods for developing melodies in terms of modes and counterpoint, all using his “Organum Mathematicum”, which was the “17th century equivalent of a laptop computer” (Burngardner 2).

Moving forward with the history of algorithmic composition, there seems to have been a surge in development with the idea of algorithmically combining smaller segments of a piece to develop a larger composition. The most popular examples of this are from Mozart (1756 - 1791) and Haydn (1732 - 1809) right around 1790. Mozart’s “The Dice Game” is a game in which the user throws dice and based on the result, chooses which measure to play next in the composition, and then which list to choose the next measure from when the dice are thrown again. This is interesting to me in terms of algorithmic composition because the chosen measure depends on which measure was last played, and therefore the “programmer” can set up these situations to his/her liking. The piece from Haydn entitled “Philharmonic Joke”, is much the same idea as “The Dice Game”, where Haydn developed “Tables according to which one can toss off minuets and trios” (Wager).

The next phase of these historical highlights that I will discuss progresses into the realm of electronic computers, starting with the “Illiac Suite”, which is considered by many to be the first computer-generated composition. The “Illiac Suite” was developed in 1956 by Lejaren Hiller and Leonard Issacson at the University of Illinois at Urbana-Champaign using a computer, and included four distinct experiments in computer music composition. The first was an experiment in “Cantus Firmi”, which is a way to express “melody in its purest form” (WattsUrizen). The second experiment was an exercise in creating four-voice segments with various counterpoint rules. The third composition experiments with algorithmically manipulating rhythm, dynamics, playing instructions, etc., and the fourth utilizes probabilities for generative grammars (using Markov chains). Typically I feel indifferent about compositions that are developed just to showcase a technology, but in this case I cannot blame these researchers for developing a purely “proof of concept” collection of compositions, seeing as they were literally the first ones to use a computer in this vein.

Shortly thereafter, in 1963, Lejaren Hiller worked with Robert Baker to develop the “MUSICOMP” programming language. MUSICOMP was developed to provided basic abstracted techniques for generating musical scores. The language included three distinct feature sets, as Hiller describes:

> We have divided the source decks into three basic parts: “System Regulatory Routines”, “Compositional and Analytical Subroutines”, and “Sound Synthesis Routines” The last are not actual synthesis programs but rather routines that prepare and organize data that serve as input to sound-generating programs... (Hiller 72)

Much like how the Illiac Suite is widely considered to be the first computer-generated composition, MUSICOMP is considered to be the “original music software environment” (Roads 848). Using this tool, its developers wrote a “Computer Cantata”, which incorporated stochastic choice processes, and other operations made accessible by the MUSICOMP environment.

Examples of computer music compositions continued to emerge at a rapid pace such as “Harmonization of the Unfigured Bass” from 1974, written using the LISP programming language. This composition attempted to produce a more “human” sounding composition by requiring a human composed bass line as input to the software. Based on this human defined input, the upper three voices were constructed. More recent developments include “Experiments in Musical Intelligence”[1] from David Cope in the 1980’s, or Roman Klinger who has developed Java-based “automatic composition” software called “MusiCom”[2]. These developments yield seemingly more complex results than all of the previous research, and the developers take all that has been discovered by Hiller and Mozart for granted so that they are able to work at a much higher level of abstraction. This concept of abstraction is obvious in modern music programming tools such as Max/MSP, which includes simple functionality for utilizing the power of Markov chains in a musical context. This abstraction allows a composer to utilize these constructs without entirely understanding how they are built underneath, and thus allows him/her to focus more on the composition itself. The power of abstraction is a fundamental concept in Computer Science, and provides the means for huge leaps in experimental technologies, such as in the field of computer music.

As I mentioned earlier, I have been developing an algorithmic composition that is based on the Fibonacci number sequence. Tools such as Max/MSP helped me to accomplish this task much more effectively by allowing me to focus on the development of the composition rather than the technical details behind the implementation. I have written another report of the implementation details of my piece which can be found on my website[3].

---

[1] http://artsites.ucsc.edu/faculty/cope/experiments.htm

[2] http://home.arcor.de/romanklinger/publications/pdf/Klinger.pdf

[3] http://colin-sullivan.com/main/archives/944

For this composition, I feel that it is important to incorporate the Fibonacci sequence on the macro scale, as well as the micro. It is for this reason that the piece contains elements such as a central melody that is heavily based on the Fibonacci numbers, as well as an overall structure that is also determined by the sequence.

When considering the structure of the piece, I wanted to incorporate similar ideas that Bartok did in "Music for Strings, Percussion and Celesta", where the structure of the piece is divided into sections that have durations based on the Fibonacci numbers. The climax of my piece is at 61.8%[4] of the total duration, similar to the Bartok piece. With this in mind, I was able to structure the sections leading up to and down from this climax based on the Fibonacci sequence as well. In my piece, there is a concept of a "global Fibonacci number" which increases up to the climax, then decreases back down afterwards. For each smaller section within the build up to (or down from) the climax, the global Fibonacci number determines the duration of the section, and determines the pace of the composition.

More specifically, before the climax, the global Fibonacci number determines the amount of time (leading up to the climax) to devote to each section, while after the climax this number determines the amount of time remaining in the piece to devote to each section. I decided that the Fibonacci number should increase from 1 to 21, and then back down again. I decided to stop the sections at 21 because I found that when dealing with time (and the amount of notes in a phrase, as I will discuss later), it was much easier to comprehend a section using the number 21, and after this (in the Fibonacci sequence), the numbers were just getting too large to help with structure. Since 21 is the 7th Fibonacci number, there are 7 sections leading up to the climax. The method that I used to determine the duration of each section is a reverse sequence of the Fibonacci numbers. I chose to use a reverse sequence because I wanted the piece to climax at 21, and I found it easier to increase tension if the duration of each section is decreasing on its way to the climax. So, when the global Fibonacci number is at 1, the section duration is based on the number 21, and when the global Fibonacci number is 2, the section duration is based on the number 13. The duration of each section is based heavily on the composition of the "central melody" in the piece, which is the bass-like voice that plays prominently throughout. Therefore, in order to understand exactly how the durations of each section are determined, the composition's central melody must be analyzed.

The central melody of the piece utilizes the Fibonacci sequence in the timing *and* pitch structure of the phrases. For the rhythm of the central melody, the amount of 1/8th notes that are played is initially equal the current global Fibonacci number. I call this number of notes the "phrase Fibonacci number", as during one global Fibonacci

[4] 1.618 is the "golden ratio", the limit which the ratio between any two consecutive Fibonacci numbers approaches

number, the phrase Fibonacci number may increase many times. More precisely, when a new global Fibonacci number is generated, the phrase Fibonacci number is initialized to this new global Fibonacci number and the central melody plays this same amount of 1/8th notes for the section of the phrase at that time. Once these 1/8th notes are played, the melody then pauses for a phrase Fibonacci number amount of 1/4 notes. This arbitrary decision was made simply because I have not yet created a more complex way to generate the rhythm for this melody, and thus these 1/4 notes provide a systematic breath between the faster portions of the phrase.

After the central melody has paused for these 1/4 notes, the phrase Fibonacci number is increased to the next Fibonacci number in the sequence, and the corresponding amount of 8th notes and 1/4 note pauses is played again. Below is a graphical representation of a portion of the rhythm as described.

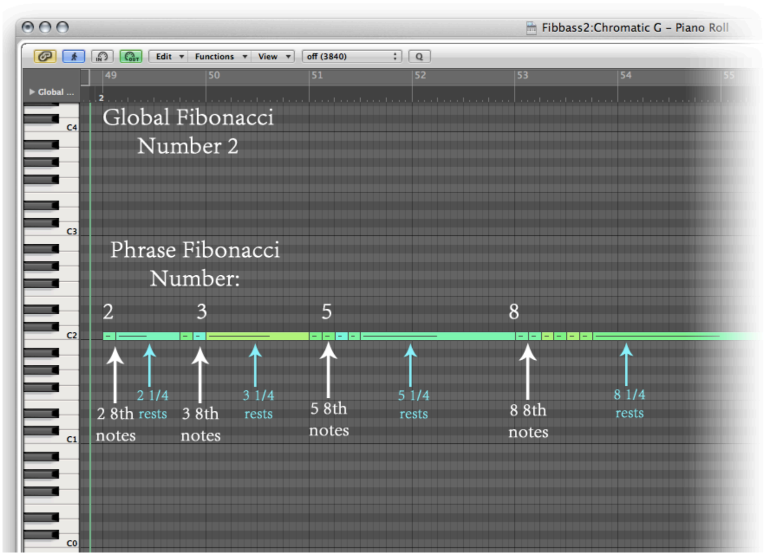

Figure 1. A graphical representation of the central melody rhythm. At different points in the phrase, a new "phrase Fibonacci number" is generated, and the corresponding amount of 1/8th notes are played followed by the same amount of 1/4 note rests.

As far as rhythm and structure is concerned, it is somewhat clear that the possibilities associated with utilizing the Fibonacci sequence are endless. The sequence does grow very quickly, but structures and rhythms that are based on the numbers in the sequence seem natural and familiar, providing pleasing results when just utilizing the first few values, or when reversing or combining different transformations of the sequence. For example, in Bartok's piece (mentioned above), there is an 89[5] measure-long segment which includes a climax 55 measures in. In the time leading up to the climax, the mutes are removed from the strings 34 measures in, leaving 21 measures before the climax actually occurs. Also, this is just for the structure of the piece, he is reported to have used the sequence in various other ways throughout. It turns out that the Fibonacci sequence works very well when used to

[5] 89 is the 10th Fibonacci number

develop structure and rhythm, but when attempting to develop pitch, it becomes much more difficult to incorporate the numbers in a meaningful way.

Based on my somewhat brief attempts, I found it very difficult to apply the Fibonacci number sequence to pitch in a non-arbitrary manner. From an innocent point of view, since the sequence rises so quickly, simply mapping the Fibonacci sequence to frequency will result in frequencies climbing out of the range of human hearing extremely quickly. Therefore, it is necessary to do some sort of "folding" with something like modulus arithmetic in order to be able to utilize more than just a few values of the sequence. A computer music student, Casey Reynolds[6], has published his extensive research in the field of Fibonacci pitch sets. He discusses that when using modular arithmetic, the "actual Fibonacci spacing is lost on the large scale, but the Fibonacci property that each succeeding pitch class is the sum of two previous pitch classes is preserved" (Mongoven 1). He goes on to discuss the patterns that emerge in the sequence as it is passed through the modulus, using modulo 12 as an example. In particular, the sequence begins to repeat after 24 values, and when considering all possible dyads (two consecutive values) in all possible inversions of the sequence, the collection of sets "bears some resemblance to the all-interval set in twelve-tone music" (Mongoven 3).

These conclusions are among the most basic that Mongoven discusses, as he moves on to many other topics including harmonies and invariances associated with these dyads. Although I attempted to take advantage of some of these concepts in the development of the pitch space for my central melody, I was not happy with the results. Systematically determining the pitch set based on these sequences often created a chromatic "white-noise" effect, where it seemed like so many random pitches are playing that no pattern was discernible. This effect was amplified when multiple voices were playing the systematically generated pitch sets at the same time. These renderings were not pleasing for me to listen to, and I decided instead to map some voices to more familiar pitch sets so that the focus of the piece could remain on the systematically determined aspects, and no indiscernible melody would detract from the overall mood and progression of the piece.

With that being said, we can continue the discussion of the construction of the central melody in my piece. For the pitch of the central melody, I went through many iterations in attempt to find something that was satisfying systematically and still sounded somewhat pleasing to me. First, I will discuss the algorithm that generates the melody, which is entirely independent of the pitch class that it gets mapped into. This "central melody algorithm" starts off at the current phrase Fibonacci number, and defines a new number: the "note Fibonacci number". This

[6] http://www.caseymongoven.com/bio/

note Fibonacci number is initially set to the phrase Fibonacci number, and will determine which pitches are chosen as the melody plays. This works by simply using the note Fibonacci number to index a 2-octave (maximum) pitch set with modulus. The indexing works as follows: First, the base note is played, this pitch is determined by the drone (which will be discussed later) or the note where the last central melody phrase left off. From there, the current note Fibonacci number is used to index the pitch set (relative to the current pitch, *descending*) to determine the next note to be played. Once this note is played, the next note Fibonacci number is generated, and the next pitch is chosen from the pitch set (relative to the current pitch, *ascending*). This process repeats (descending, then ascending) for the amount of notes in each section of the phrase. Below is a diagram that may help clarify. In this case, our pitch set is the list of increasing integers from 0 - 23 (representing a 2-octave chromatic scale).

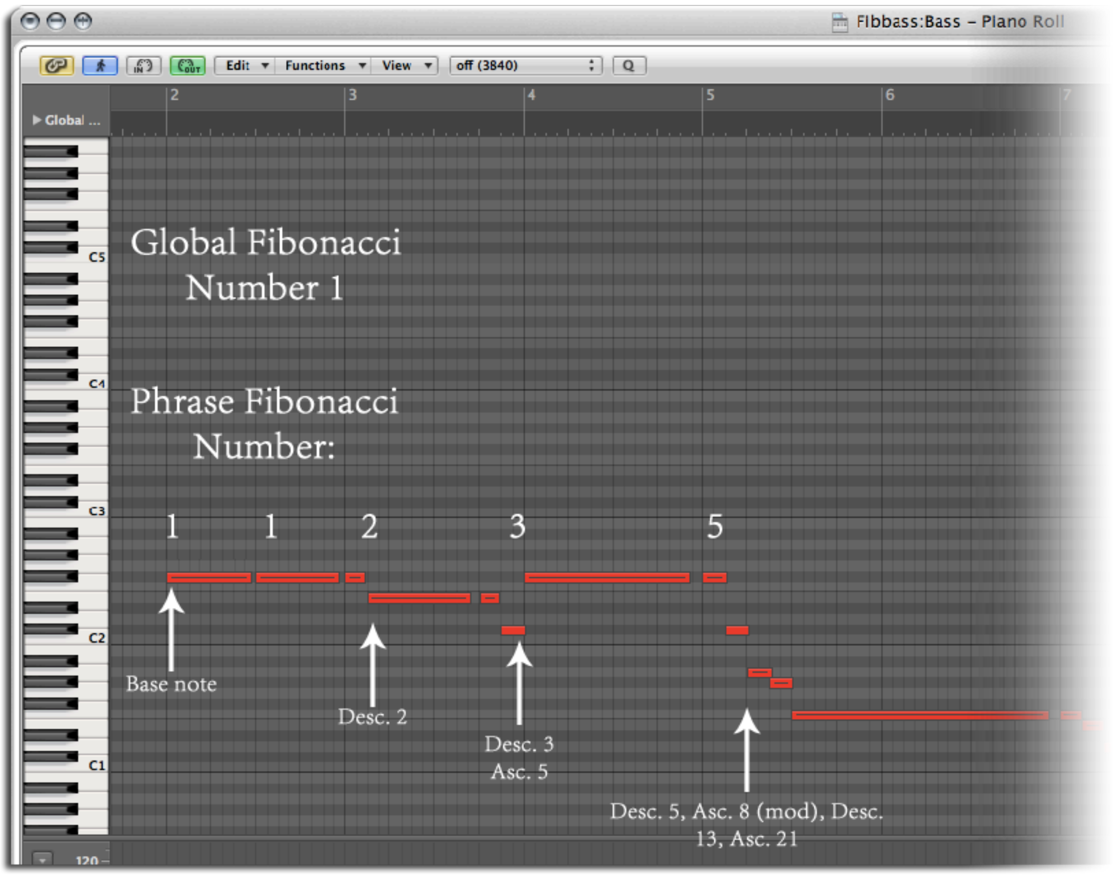

Figure 2. Pitch selection algorithm for central melody. At each new "phrase Fibonacci number", the pitches are selected based on an alternating (ascending/descending) selection process, where the pitch class is indexed by the increasing "note Fibonacci number" each time a note is played. Here the chromatic pitch class is used for clarity.

When developing the melody, since I discovered that generating pitch classes would require significant effort, I left this pitch selection algorithm alone, and decided to focus my efforts on the construction of the pitch classes instead. I began to map this pitch selection algorithm to various familiar pitch classes, including the whole tone scale, blues, pentatonic, harmonic minor, etc. in interest of hearing the algorithm in action over these sets. I liked how the chromatic pitch set sounded when the piece started, and the transition into the whole tone set after that. I began to place the rest of the results somewhat arbitrarily in a way that helped emphasize my climax in a dramatic way, but I found this extremely unsatisfying for a few reasons.

Providing a "tour" through the results of this algorithm is an interesting concept, but is not what I want to achieve with this piece. It was for this reason that I looked a bit longer at why the whole tone pitch class sounded natural after the chromatic to me. I realized that the whole tone class sounds natural after the chromatic because is just every 2nd pitch removed from the chromatic class. Since I was using the whole tone class on global Fibonacci number 2, I decided to see how the melody would sound if on global Fibonacci number 3, I removed every *3rd* note in the pitch set, and did this for all global Fibonacci numbers. This worked out wonderfully for now; it sounds consistent enough to not sound arbitrary, and is systematic enough to satisfy me. Although the results still sound very chromatic, I decided to move on and begin to accentuate aspects of the piece with other layers of sound.

The prominent voice that is "echoing" the central melody is systematically delayed from the central melody based on the current global Fibonacci number. The purpose of this voice is to emphasize the central melody, and to add some variation in pitch space. The voice plays behind the central melody at an amount of 1/4 notes equal to the current global Fibonacci number. Because this value changes throughout the piece, the variation in the delay time adds to the overall complexity and in my opinion, makes the central melody itself sound much more interesting. The other aspect of this voice that is changing throughout the piece is the range of pitches that are played. Based on the global Fibonacci number, the range of pitches available to the voice expands such that during the climax, this voice is playing extremely high and low notes, with nothing much in between. This helps to build tension in the piece as well, since with each iteration of a new global Fibonacci number, the pitches are migrating away from where they started.

One of the first voices that I developed was the "buzzing drone" that can be heard throughout the piece. This drone is fairly constant, and becomes increasingly distorted and warped as the piece progresses toward the climax, and then eases these attributes back down afterwards. The drone defines the base pitch from which all of the other voices are transposed. For example, in the generation of the central melody, if the decided value of the pitch is 0, then the central melody will play a note that is the same as the drone. The pitch of the drone is based on a simple descending pitch scheme that starts on an arbitrary pitch (G), and descends the global Fibonacci number amount of semitones each time the global Fibonacci number changes. Immediately following the climax, the drone instead ascends with each global Fibonacci number. Again, these decisions are somewhat arbitrary, and I would like to spend more time developing them in the future.

Another prominent timbre is the distorted "click" that occurs throughout the piece and increases frequency as the piece progresses. This click sound occurs each time the global Fibonacci number is changed, and then uses the global Fibonacci number to determine how many times a click is played during that time period. Since these clicks are distributed evenly among the section, the rhythm of the click often goes out of sync with the rhythm of the central melody. My intention is to make the tempo of the central melody (and everything else) gradually change to match these clicks, but some technical limitations prevented me from experimenting with this at the moment.

Possibly my favorite timbre that I have produced for this piece is the voice that sounds somewhat distant because it is played through a large amount of reverb. I will refer to this voice as the "minor accompaniment". My intention with this voice was to add to the tension of the piece before the climax by making the melody sound very dramatic. I attempted to add to the tension by decreasing clarity in the timbre with a large amount of reverb on the voice, which is slowly reduced as the climax approaches. As mentioned above, I spent quite some time looking into systematically modifying pitch classes to make them sound interesting, but my attempts did not yield favorable results. In the end, I've decided to simply map this voice to the harmonic Minor scale, as I found it was an easy way to make the voice sound dramatic. To generate this melody, I began with the algorithm for the central melody (described above), and modified it slightly. The algorithm for this voice uses the global Fibonacci number to both decide wether or not to play a note, and what duration to play (or not play) for each note. This decision is made at each tempo tick (1/480th of a 1/4 note), so that the phrase can sound much different each time. The way the phrase generation works is fairly straightforward. Each time a global Fibonacci number is generated, the probability that a note will play is updated. The probability that a note will play for Fibonacci number 0 is zero (this is currently not included in the piece), and the probability that a note will play for Fibonacci number 1 is 0.161803399. Whenever a new global Fibonacci number is generated, a new play probability is generated the same way a Fibonacci number is generated (by adding the previous two values). This ensures that the probability that a note will play for this voice is very infrequent in the beginning, and increases at a "Fibonacci-like rate". After a few Fibonacci numbers, this value is greater than 1 so it has no effect as a percentile, but still works in the piece because by that time, we are headed to the climax and these frequent notes help to build the energy of the piece. The durations for the notes (or pauses) in this phrase are also determined based on the global Fibonacci number. This works by first defining a base length (in this case I am using an 8th note), and multiplying that duration by the current global Fibonacci number each time a new one is generated. Whenever a new global Fibonacci number is generated, the new probability is added to the list, and when a note is about to be played, a duration is chosen from this list at random. Assigning probabilities to each of these durations would be ideal, and would allow the notes that this voice is playing to get longer and longer as the piece progresses, building tension by a sort of "super division", while most other timbres are sub dividing to create tension.

The remaining percussive elements in the piece are simply accentuating the global Fibonacci number by playing the corresponding number of beats when the global Fibonacci number changes, or by randomly playing a percussive timbre based on a similar probability scheme to the one described above. The heartbeat-like pulsing that increases pace leading up to the climax is simply playing at a rate relative to the current global Fibonacci number.

There are a few elements in the beginning of the composition that were placed somewhat arbitrarily, to ease the listener into the space of the piece. Initially, I had the piece starting at global Fibonacci number 1, but have since added a bit of cushion before the central melody starts. This time allows for the drone and heartbeat to establish the space in which the rest of the timbres will enter. The first few bars of the central melody were also arbitrarily manipulated, to sort of provide a similar effect of "easing" the listener into the melodic ideas of the piece.

In the end, this piece is still in a fundamental stage. Coming from an extremely limited music theory background, I enjoyed integrating a bit of a mathematical approach to the composition, as well as looking at various components from a brute-force perspective by utilizing the computer as a rapid development environment. I feel accomplished because I challenged myself to consider the systematic modification of all aspects of the piece, and gave most aspects a reasonable attempt (considering my time frame). The historical research helped me to grasp the magnitude of what it means to build an algorithmic composition utilizing the Fibonacci numbers, and out of this research I feel that I've begun to develop some very interesting concepts, but still have much more to accomplish before I can be satisfied with my articulation of this idea.

Works Cited

Ariza, Christopher. "Navigating the Landscape of Computer Aided Algorithmic Composition Systems: a Definition, Seven Descriptors, and a Lexicon of Systems and Research." *International Computer Music Conference Proceedings* (2005). Web. 25 Sept. 2010. <http://quod.lib.umich.edu/>.

Baggi, Denis L. "The Role of Computer Technology in Music and Musicology." *LIM | Laboratorio Di Informatica Musicale*. 9 Dec. 1998. Web. 26 Nov. 2010. <http://www.lim.dico.unimi.it/events/ctama/baggi.htm>.

Burngardner, Jim. *Kircher's Mechanical Composer: A Software Implementation*. Rep. Web. 26 Nov. 2010. <krazydad.com>.

Cope, David. *The Algorithmic Composer*. Madison, WI: A-R Editions, 2000. Print.

Hiller, Lejaren. "Compositional Techniques Involving Computers." *Music by Computers*. Ed. Foerster Heinz Von and James W. Beauchamp. New York: J. Wiley, 1969. Print.

Mongoven, Casey. "Fibonacci Pitch Sets." *Caseymongoven.com*. 2000. Web. 26 Nov. 2010. <http://www.caseymongoven.com/writings/>.

Nierhaus, Gerhard. *Algorithmic Composition: Paradigms of Automated Music Generation*. Wien: Springer, 2009. *Google Books*. Web. 25 Sept. 2010. <http://books.google.com/books?id=jaowAtnXsDQC>.

Roads, Curtis. *The Computer Music Tutorial*. Cambridge, Mass. [u.a.: MIT, 2007. Print.

Wager, Willis. "Mozart's Musical Dice Game from Carousel Publications Ltd." *World Music Instruments: Strings, Music Publications and World In Tune (Integrative Curriculum)*. 2000. Web. 02 Oct. 2010. <*http://www.carousel-music.com/shooters.html*>.

WattsUrizen. "Species Counterpoint: Cantus Firmi." *Harmony Central Forums*. 10 Nov. 2002. Web. 02 Oct. 2010. <http://acapella.harmony-central.com/showthread.php?t=162345>.