Automated Music Generation for Sight Reading

Undergraduate Research Project

Spring Semester 2008

Developed and written by Kevin McKee

Supervised by Dr. Boaz Porat

Introduction

A major challenge that any beginning student faces when learning to play a musical instrument is sight reading. Whereas in some instruments each note can be played in only one way, in other instruments, most notably string instruments, each note (except for a few lowest and highest notes of the instrument's range) can be played on several strings. For example, on the guitar, the note C4 (middle C) can be played on the 1st fret of the 2nd string, or on the 5th fret of the 3rd string, or on the 10th fret of the 4th string, and so on. Consequently, when playing a piece of music on the guitar (or any string instrument), the player must continuously make decisions as for what strings and with which left-hand fingerings to play the upcoming notes. The ability to do this at sight and without pre-reading and preparation is called sight reading.

The project described in this report focuses on sight reading for the guitar, although it can be used equally well with any instrument that uses the treble clef. On the guitar, *position* is defined as the fret on which the index finger rests. When playing a simple piece of music (spanning no more than two octaves), the player can usually choose a position (depending on the range of notes) and stay on the same position throughout the piece. In more advanced music, the player may have to change positions as the music flows. The choice of position is also affected by the key. For example, in the key of C, the open position as well as positions 1, 3, 5, 7, 8 and 10 are all convenient (depending on the range of the piece); however, positions 2, 4, 6, 9 and 11 are less convenient. In summary, the aspiring guitar student should eventually be able to sight read any piece of music, of any key and any range of notes.

Beginning students spend most of their time playing tonal music; that is, music in a specific diatonic key (major or minor). However, more advanced students must be able to read music that is not strictly tonal (that is, including many accidentals) or even atonal.

Another aspect of sight reading concerns the ability to read rhythm. Beginning students concentrate on regular rhythms (such as 4/4 or 3/4) and relatively long notes (such as quarter notes and half notes); more advance students need to read short notes (eighth and sixteenth). Also, advanced sight reading requires reading of irregular rhythms and syncopations.

Whereas many guitar-learning books are available, few devote sufficient attention to extensive sight reading. Even the ones that exist cannot possibly cover all positions and all keys, with varying degrees of difficulty of rhythms and atonality, while providing numerous study pieces for each choice. Motivated by this, we set out to develop software that generates melodies "on demand", giving the user a wide range of choices, including keys, melody ranges, rhythmic irregularities, and levels of atonality. The melodies

generated by the software are random; that is, they follow statistical rules rather human artistry. Consequently, they are not necessarily pleasant to the ear. Their use is, as we have said, to serve as sight-reading material, not as performance pieces.

The software described in the report was written in Java. Our reasons for choosing this particular language were its portability and the availability of rich (and free) class libraries, in particular graphic libraries. Consequently, the software will run on any computer that supports Java Runtime Environment (JRE).

Melody Generator

The Melody Generator is a computer program designed to create a MIDI file that can be used to create sheet music for sight reading. The program uses a Markov Chain to calculate the next note based on the current note and the probabilities of the next notes. In some cases, the Markov Chain is expanded to a point where the next note is a function of not only the current note but also the previous note. For example, when the melody leaps in one direction, the next note has a higher probability of stepping in the other direction to compensate for this leap. The program creates a MIDI file, which can then be placed into a program such as MidiIllustrator, which converts the MIDI file into sheet music.

The user can define a broad array of parameters that affect the output of the MIDI file. If a user has a basic understanding of the different input parameters, he or she can create a piece of sheet music that is tailored to exactly what they need. These parameters are explained in further detail below in a "User Guide" for the Melody Generator.

Time Signature
Default: 4/4
The user may select the desired time signature. The options are 4/4, 3/4, 2/4 and 6/8.

Repeated Notes
Default: Yes
The user may specify to the program whether or not they would like it to allow the repetition of notes. If the user selects the default selection, "Yes", then the program is will be allowed to create a melody where the next note may be the same as the previous note.

Key Mode
Default: Major
This input selects the key mode for the song to be played. The options are Major, Natural Minor, Harmonic Minor and Jazz Melodic Minor.

Key Profile
Default: Scalar
The key profile can be either Scalar or Triadic. A Scalar key profile gives an equal probability to all notes in the scale. A Triadic key profile will emphasize the 1, 3 and 5 notes (the notes of the tonic triad).

Output File

Default: Output_File

The output file is the name the user would like to select for their song. When a song is created, it is written into a MIDI file with the name Output_File.mid. The user can change "Output_File" to any word, number, or combination of these that he or she wishes. If the user would like to create multiple different MIDI files, this parameter will have to be changed after each file is created; if it is not changed the new MIDI file will overwrite the previous file.

Tonality Factor

Default: 1

The tonality factor is a variable that affects how tonal the melody is. The melody becomes more tonal as the value gets closer to one and becomes less tonal as the value gets closer to zero. A value of one is purely tonal.

Level Probabilities

Default: LP1 = 1, LP2 = 1, LP3 = 1, LP4 = 0.3, LP5 = 0

The level probabilities are the most straightforward way of setting the rhythm of the song. Each level probability is dependent on the time signature. The first level probability (LP1) represents the probability that there will be a note at the beginning of each measure for any time signature. A value of 1 means there will be a note at the beginning of 100% of measures. A value of 0.5 means there will be a note at the beginning of 50% of measures. The rest of the level probabilities are defined in the table below, based on the time signature.

Time Signature	LP2	LP3	LP4	LP5
4/4	Note on 3rd beat	All quarter notes	All 1/8th notes	All 1/16th notes
3/4	All quarter notes	All 1/8th notes	All 1/16th notes	N/A
2/4	All quarter notes	All 1/8th notes	All 1/16th notes	N/A
6/8	Note on 4th beat	All quarter notes	All 1/8th notes	All1/16th notes

Meter Factor

Default: 1

The meter factor is an indirect way to modify the level probabilities. As the meter factor is decreased, the level probabilities are decreased. The more the meter factor is decreased, the more irregular the rhythm becomes.

Triplet Probability

Default: 1/4 = 0.1, 1/8 = 0.1

These variables affect the probabilities that the program will generate triplets within the song. The program generates the original rhythm and then recognizes anywhere within the song where a triplet is possible. Given the default values, there is a 10% chance of a triplet at that point. These values can be anywhere from 0 to 100 percent. Here 1/4 refers to "quarter triplets" (three notes over a time interval of two quarters) and 1/8 refers to "eighth triplets" (three notes over a time interval of two eighths).

Number of Measures
Default: 48
This variable allows the user to select the number of measures, and thus the length, of the song. This must be an integer between 10 and 200. The program will not accept any numbers out of this range, real numbers or words. The default of 48 measures typically creates a song of about one minute and thirty seconds in length.

Openness Factor
Default: 5
This is a value between 1 and 12 with a default of 5. As the openness factor increases, the probability of the melody moving in larger steps increases.

Key Signature
Default: C major/A minor/0 sharps or flats
The user may select the key signature he or she would like to play in using the Key Signature selection. The user has the option of selecting any of 15 key signatures. The default key signature is C major/A minor. The key signatures are given as their names as well as the number of sharps or flats used in that key.
After the key signature is changed, this program requires the user to change the top and bottom notes before a MIDI file can be created. Every time the key signature is changed, the program re-initializes the top and bottom notes to E7, E#7 or Eb7 and E3, E#3 or Eb3, depending on the key signature and whether the "E" note should have a sharp or flat. The top and bottom notes must be moved to ensure that they are representative of the new key and the desired range of the melody.

Top and Bottom Note
Default Top Note: E7 (or E#7 or Eb7)
Default Bottom Note: E3 (or E#3 or Eb3)
The user can set the range of notes he or she would like to play with this input parameter. As the key signature changes, these notes change as well. This parameter also sets the initial note. The first note of the song is the lowest tonic note within the specified range. The default values were selected based on the range of notes of electric guitars.

Finish
Default: N/A
After the user has concluded setting all of the parameters, he or she need only click the "Create" button and a MIDI file within the set parameters will be created.

Harmony Generator

The Harmony Generator is a separate program that takes a different approach to generating music. In contrast to the Melody Generator that creates a melody without any harmony, the Harmony Generator creates a harmony without a melody. The Melody Generator is sufficient for creating sight reading exercises, but leaves much to be desired from a musical quality standpoint.

The main advantage of the Harmony generator is its ability to adhere to musical form. The melody generator is only aware of the previous one or two notes at any given time. Imagine writing a sentence if you could only know the previous one or two words at any given time. By this same principle, imagine writing a sentence if you had only a conception of what the following word should be, but none after that. It would be extremely difficult (though not impossible) to write a coherent sentence in this way. The same holds true with the Melody Generator; it is capable of writing a song that sounds pleasing, but it does not always accomplish this goal.

The Harmony Generator is in the early stages of development. Consequently there are numerous disadvantages to the program. It does not currently generate a melody at all; only a harmony in the form of musical phrases. It also does not have some of the more basic musical inputs, such as key signature and time signature. The program assumes a 4/4 time signature and its output is only the roman numerals of the chord degrees. As a means of convention, capital letters mean a major chord (such as “V”) and lowercase letters designate a minor chord (such as “vi”). For example, in the key of C, I stands for C, ii stands for Dmin, iii stands for Emin, IV stands for F, V stands for G, vi stands for Amin, and vii-dim stands for Bdim (B half-diminished if the user so prefers). Thus, the user is assumed to be proficient in harmony to the extent of being able to translate chord degrees to actual chords for any desired key.

Form Generator

The Form Generator is the first step in the process of creating a harmony. Almost all musical compositions can be broken down into phrases. A musical phrase is similar to a sentence in spoken language. Most songs establish a phrase early in the song and then return to the same phrase (or a slightly modified version). Conventional musical structure defines the musical form with letters, such as A-B-A-B. A song with this form will begin on one phrase (denoted A), then move to a second different phrase (denoted B). After phrase B is completed, the song will repeat phrase A, followed by phrase B again.

The Harmony Generator allows the user to choose his or her own musical form. If the user does not want to choose, he or she can have the program choose one of the five musical forms at random. The musical forms available to the user are A-B-A-B, A-A-A-B, A-B-B-C, A-A-B-A, and A-B-A-B-C-B (also known as Popular musical form).

The Form Generator is also responsible for setting the cadence of each phrase within the song. The cadence refers to the last two chords at the end of a musical phrase. A chord can either create tension or resolution for the listener. The V, or dominant, chord creates tension, while the I, or tonic, chord creates resolution. When a listener hears a dominant chord, he or she expects it to be followed by the tonic to

resolve the tension that was created. When a phrase concludes with a V → I progression, it is called a closed (or perfect) cadence. Almost all songs (with some exceptions) end on a closed cadence. In this program, all songs end on a closed cadence.

While it is desirable to resolve tension at the end of a song, we want to build tension throughout the composition. Therefore, not all phrases will end with a perfect cadence. In addition to the perfect cadence, the Harmony Generator can set an open or deceptive cadence.

An open cadence refers to a musical phrase that ends on the dominant chord. As stated before, the dominant chord creates tension. It is desirable to create tension in the beginning or middle of a song, and the open cadence is used frequently to accomplish this task.

Another less common method of creating tension is called a deceptive cadence. A deceptive cadence holds true to its name, as it tricks the listener. The second to last chord of this phrase is the dominant chord, but it leads to any chord except the tonic. The listener hears the V and expects to hear the I, but is instead taken to a different chord.

The Form Generator is designed to finish each song with a closed cadence. The final phrase of every song is automatically set to "closed". All other phrases in the song are set to either "open" or "deceptive". Since open cadences are more common than deceptive cadences, the program will use an open cadence 80% of the time and a deceptive cadence the other 20% of the time.

Once a form is chosen, the program must generate phrases. After the phrases are generated, the Form Generator puts the phrases together into a song for the final output.

Progression Generator

The Progression Generator is responsible for setting the length and rhythm of a phrase based on input from the user. The user is allowed to choose how quickly he or she would like the chords to change in the song. The Harmony Generator allows three options; slow, medium and fast.

If the user chooses the slow chord progression, the phrases have a higher probability of being long phrases because a short phrase with a slow progression is a good recipe for a phrase with only one or two chord changes. On the other hand, a fast chord progression has a greater probability of creating shorter phrases so we do not generate phrases with too many chord changes. The length of the first phrase in a song is determined by the following probabilities:

If the change rate is…	the first phrase will be	with this probability.
Slow	8 measures long	60%
	4 measures long	40%
Medium	8 measures long	25%
	4 measures long	50%
	2 measures long	25%

Fast	4 measures long	60%
	2 measures long	40%

After the length of the first phrase is determined, the Harmony Generator sets the rhythm of the chord changes. A chord can change after two full measures, after one full measure, or after one half measure. The speed at which these chords change is a function of the user input. Remember that 8 quarter notes equals two measures, 4 quarter notes equals one measure and 2 quarter notes equals a half measure.

If the change rate is…	**… the chord will change after…**	**… with this probability.**
Slow	8 quarter notes	70%
	4 quarter notes	20%
	2 quarter notes	10%
Medium	8 quarter notes	15%
	4 quarter notes	70%
	2 quarter notes	15%
Fast	8 quarter notes	10%
	4 quarter notes	20%
	2 quarter notes	70%

Once the first phrase is set, the next phrase is a function of the previous phrase and the change rate. Phrases typically get longer as the song progresses, but this is not always the case. The following table shows the probabilities of the next phrase with respect to the change rate and the previous phrase length.

If the change rate is…	**… and the previous phrase was…**	**… the next phrase will be…**	**… with this probability.**
Slow	8 measures long	8 measures long	85%
	8 measures long	4 measures long	15%
	4 measures long	8 measures long	30%
	4 measures long	4 measures long	70%
Medium	8 measures long	8 measures long	75%
	8 measures long	4 measures long	25%
	4 measures long	8 measures long	30%
	4 measures long	4 measures long	60%
	4 measures long	2 measures long	10%
	2 measures long	8 measures long	10%
	2 measures long	4 measures long	30%
	2 measures long	2 measures long	60%
Fast	4 measures long	4 measures long	85%
	4 measures long	2 measures long	15%

	2 measures long	4 measures long	30%
	2 measures long	2 measures long	70%

There is one exception to the rule that phrases generally get longer as the song progresses. In the Pop form, the C phrase stands for the bridge. The bridge is typically shorter than the other phrases in the song, but is still dependent on the length of the previous phrase. It is also not dependent on the rate of change. The probabilities for the bridge are given below.

If the previous phrase is…	… the bridge will be	… with this probability.
8 measures long	8 measures long	20%
	4 measures long	80%
4 measures long	4 measures long	40%
	2 measures long	60%
2 measures long	4 measures long	20%
	2 measures long	80%

After the length and rhythm is set for each phrase, these phrases are ready to be completed with a harmony, which is handled by the Phrase Generator.

Phrase Generator

The Phrase Generator is the part of the program that generates the chord progression for each individual phrase. Once the Harmony Generator has completed running through the Phrase Generator, it will have created a complete musical phrase.

Beginning

The first step of the Phrase Generator is to set the first chord of the phrase to the tonic, or a “I”. The beginning of the phrase is set to the tonic for two reasons. First, the tonic can lead to any chord, making it an ideal starting point. Second, most musical phrases begin on the tonic, so it makes sense to start each phrase here.

Ending (Cadence)

The next step in building a phrase is to set the ending, or cadence. A cadence is a series of two chords that end a phrase. There are many different types of cadences in music. This program has functionality for three cadences; closed, open and deceptive. The differences in these cadences are explained in further detail in the Form Generator section.

If the cadence is closed, the second-to-last chord is set to “V” and the last chord is set to “I”.

If the cadence is open, there are three ways for the program to set the cadence, each with an equal 33.3% probability. In the first case, the last chord is set to "V" and the second-to-last chord is set by the typical rules that dictate which chord should lead to the V (see the "Middle" section for more details). The second case is similar to the first case except the last chord is replaced with a "V7". The third case sets the last chord to "V7" and the second-to-last chord to "V".

If the cadence is deceptive, the second-to-last chord is set to "V". The last chord is set to "IV" with 50% probability, "vi" with 25% probability and "ii" with 25% probability.

Note: When there are only two chord changes in the phrase, the second to last chord is also the first chord. In this situation, the first chord (which was set to a "I" when the beginning was set) will be overwritten by the second-to-last chord in the cadence and the phrase will not begin on the tonic.

Middle

After the beginning and ending are set, the program will attempt to set the rest of the chords in the middle of the phrase. The program will only attempt to set the middle chords if there are more than three chords in the phrase. If there are three or less chords in the phrase, the entire phrase will have already been completed when the beginning and ending were set.

When there are four or more chord changes in the phrase, the program will begin setting the chords in the middle. The program starts at the end of the phrase (just before the cadence) and works its way back to the beginning. Each chord is a function of the chord *that it leads to*. The PhraseGenerator has been programmed with a table of chord progressions and their corresponding probabilities. The complete table of probabilities is shown below.

Chord Progression	**Probability**	**Chord Progression**	**Probability**	**Chord Progression**	**Probability**
vi → ii	65%	IV → iii	35%	V →IV	40%
iii → ii	15%	V → iii	30%	vi → IV	30%
V → ii	15%	vii-dim → iii	25%	iii → IV	25%
IV → ii	5%	ii → iii	5%	ii → IV	5%
		vi → iii	5%		

Chord Progression	**Probability**	**Chord Progression**	**Probability**	**Chord Progression**	**Probability**
IV → V	55%	V → vi	60%	IV →vii-dim	55%
vi → V	22.5%	iii → vi	30%	ii → vii-dim	45%
ii → V	17.5%	ii → vi	5%		
iii → V	5%	IV → vi	5%		

As stated above, this table is used to find previous chords. Take, for example, a phrase with five chord changes and a closed cadence as shown below.

$$I \rightarrow \_\_ \rightarrow \_\_ \rightarrow V \rightarrow I$$

The Phrase Generator will set the third chord in this sequence next. If you consult the table, you will find that the third chord will be "IV" with a 55% probability, "vi" with a 22.5% probability, "ii" with a 17.5% probability, and "iii" with a 5% probability.

These probabilities are created because the program generates a random decimal number between 0 and 1 and compares it against an array of numbers defined within the program. If that random number is 0.55 or lower, the third chord will be set to "IV". If the number is between 0.55 and 0.775, the third chord is set to "vi". If the number is between 0.775 and 0.95 the third chord is set to "ii". Finally, if the number is between 0.95 and 1, the third chord will be set to "iii". Let us continue this example and assume that the randomly generated number was below 0.55. The program will set the third chord…

$$I \rightarrow \_\_ \rightarrow IV \rightarrow V \rightarrow I$$

and restart the same process using the probability for chords leading to IV. Once the Phrase Generator sets the second chord, it will recognize that it is at the beginning of the phrase and stop generating progressions.

Chord Substitutions

There are multiple scenarios where musical theory dictates that one chord can be substituted for another related chord. This program does not have functionality for chord replacement, except for one scenario. In the case of iii → vi, iii becomes III with a 10% probability and it becomes III7 with a 10% probability. The reason is that III is the dominant chord of vi and III7 is its dominant seventh. It is thus common to denote the progression III → vi as V/vi → vi; however, we will use the simpler notation III → vi.

Rare Chord Progressions

A rare chord progression is defined as any chord progression with a 5% probability of happening. A complete list of rare progressions can be found below.

Rare Chord Progressions (5% chance of occurring)	
VI → ii	ii → iii
vi → iii	ii → IV
iii → V	ii → vi
IV → vi	

These progressions are atypical in music and may sound awkward to the ear. If the user is looking for a more traditional sounding harmony, he or she may choose to disallow rare chord progressions. If the user

sets the program to disallow rare chord progressions, the probability of these progressions will be set to zero and the probabilities of the rest of the chords will be modified slightly. This is best explained through an example. Let us assume the program is looking to fill the following progression:

__ → V

If the user has decided to allow rare chord progressions, this could be completed with the iii → V progression. If you refer back to the example in the "Middle" section you will find that this progression would be chosen if the randomly generated number was between 0.95 and 1. However, when the user decides to disallow rare progressions, a random number between 0.95 and 1 should no longer dictate a iii → V progression. The program accounts for this by multiplying the random number by 0.95 (thus giving us a new number between 0 and 0.95) and then comparing this new number against the probabilities for the non-rare progressions. When this happens, the probabilities for all progressions are affected slightly.

Chord Progression	Original Probability	Chord Progression	New Probability
IV → V	55%	IV → V	57.89%
vi → V	22.5%	vi → V	23.68%
ii → V	17.5%	ii → V	18.42%
iii → V	5%	iii → V	0%

Uncommon Probabilities

If the user decides to allow uncommon chord progressions, they may also choose to increase the probabilities of the program using those rare progressions. A rare progression is defined in this program as a progression with a default probability of 5%. The user has the ability to increase this probability to as much as 10%, thus increasing the chances of seeing this chord progression by a factor of two. This modification is similar to the one described in the "Rare Chord Progressions" section. Let us again use the example of a chord leading to a V, as shown below:

__ → V

We will assume that the user has decided to allow rare progressions and has decided to increase uncommon probabilities by 100%. The Phrase Generator will convert that percentage into a decimal and multiply it by 0.0556. This number will then be added to the upper bound of the … In this example, the program will choose the rare progression of iii → V when the random number is between 0.95 and 1.0556. The original random number is multiplied by this new upper bound (thus giving us a random number between 0 and 1.0556) and is compared to the new table with updated probabilities.

Chord Progression	Original Probability	Chord Progression	New Probability

IV → V	55%	IV → V	52.10%
vi → V	22.5%	vi → V	21.31%
ii → V	17.5%	ii → V	16.58%
iii → V	5%	iii → V	10%

The probability increase given as a percentage in the program is only an approximation. For example if the user set the program to increase uncommon probabilities by 10%, the user would expect the probability of an uncommon progression to be changed to 5.50% However, the new probability would actually be 5.53% The difference of 0.03% is minimal but warrants mentioning.

Conclusion

The Harmony Generator by itself is a useful program for generating a Harmony. However it is musically very basic. It should be expanded by adding functionality for chord substitutions and extended chords. Once this is complete, the next step will be to lay a melody over the harmony. With the basis of a solid harmony that takes musical form into account, this program has potential to create more pleasant sounding music than the Melody Generator itself.