Artificial Intelligence Component

The artificial intelligence component of the project has six jobs in order to produce its music. The first job is to develop the backbone of the rhythm for both the melody and the accompaniment. Depending on what it chooses, the computer will develop its bass the same way it develops the backbone of the melody; it will only develop the bass this way if the bass it chooses is ostinato since the walking bass and the chord progression already have a set rhythm. For the chord progression it will develop the chords using basic theory on every major scale. After it creates the basic chords it will develop first and second inversion chords to add diversity to the bass. It will choose the chords at random since all of the chords are close together and all have the same possibility of being chosen. After, the rest of the measure will be filled up with xs in order to simulate the chord being elongated. The walking bass also has a set rhythm since all it does is set up as many beats as there are in the top number of the measure; for example in 3/4 time there will be a walking bass of 3 quarter notes in each measure of the bass. The notes of the bass will be chosen at random from any of the notes in the scale in order to make it sound more like in the key. Since the notes in the scale are all close together, it will make the bass simple to play and therefore no note is better to pick than any other note. In order to create the backbone it takes in all of the values for each note, looks at how much room it has in the measure and chooses any of the four notes that it can fit in there at that time. If there is more than one note then it will choose on at random. Once it does that it will subtract that value from the amount in the measure and continue on from there and do it again until the amount in the measure is zero. For the ostinato bass, that is where the rhythm stops but since the melody is more important than the bass, the melody will go through a fitness function will which see if the rhythm it created is good enough for the time signature that it has. If it is good then it passes on to the melody function; otherwise it will bring the rhythm to a fitness function which will increase the size of the rhythm so it is more likely to be chosen next time. The second job is borrowed from a function from a paper I read [1] which takes a random note from the melody and splits it into two different notes (for example a quarter not will split up into two eighth notes). This will increase the size of the melody slightly and hopefully next it be will big enough to be chosen.

The third job of the agent is to take the rhythms of the melody and the bass and create harmony between them in order to eliminate any chances of dissonance (instability between notes) to make them sound better. To do this, the function will loop through the bass and find the first note, which is always the first space in the vector. After that it stores both the value of the note and where it is and loops through the melody until it reaches that space in the vector. If there is no note in that space then it skips it and goes back to the bass and continues through the vector until it reaches another note; otherwise it will take the note from the bass and form consonant, or “nice-sounding notes” to choose from which correspond with the note. Once it does that it will choose one at random and continue on with looping through the bass. When looping through the melody if it comes across a note that does not correspond with a note in the bass in its spot then it brings that note to a function and develops a note based off of the previous note. It will take that note a create two random numbers; one that dictates whether or not in stays at that note, creates a higher pitch or creates a lower pitch and the other number dictates how many half-steps, or adds or subtracts from the previous note, it should move. For example if the previous note was a C, or in midi ‘0’, and the number generator creates a 2 and a 5 then, respectively it creates a note that moves down and add 5 to the note and creates a ‘5’, or an F. This will create a tone which passes from one note to the other without something else being in the bass. It will continue this until both vectors have been looped through and then repeat this 2,000 times in order to use every consonant note available and to create the best melody it can.

Once it is finished creating five-thousand melodies, it will score each of them based on three basic aspects of music composition: how much it uses the tonic note, how many of the notes are in the key signature and does the melody contain many skips. The tonic note of a piece of music is the one note that it should always come back to and is the center of the piece of music (for example in the key of C major the tonic note is C). Therefore any melody that uses the tonic note many times it objectively a better melody than one that doesn’t. The same goes for melodies that use notes in the key. Even though many composers use notes outside of the key, most of the time you will see pieces of music with the majority of notes being in the key signature. For example the key signature of C major is C D E F G A B and a song in C major will use these notes consistently; although some composers will use C major as a way to use notes outside of the key (B flat or D sharp as a couple of examples). Finally the last aspect of music that it will be scoring on is the number of skips in the melody. A skip in a melody is anything greater than a step. For example, a C (‘0’) to a D (‘2’) is a step but a C to a F (‘5’) is a skip. Some skips in a melody are fine however constant use of them do not sound good and it makes it difficult for players/singers to play/sing the melody. Therefore a melody with fewer skips than another melody is better melody. The function ignores intervals from 0 to 3 since those are fine but anything bigger than that will be subtracted from the score. After everything is scored then the deque which contains all scored five-thousand melodies will be looped through and from there it will choose the melody with the highest score. If any melody contains the same score, which is unlikely but still possible, will be skipped in favor of the first melody with the same score.

Overall, I believe that this melody composition AI is slightly better than the AI’s in the papers I have read. This is due to the evaluations and functions I give it which will objectively make melodies sound and look better than some of the melodies created in the papers. The Klinger paper [1] for example focuses more on melodies that contain many notes which makes look nice but many notes do not always create a good melody. The Oliwa paper [2] also seemed to be more focused on having more impressive notes than trying to create harmony with the other instruments which resulted in dissonance, especially in the piano duet. Since my AI focuses more on creating harmony and the best melody possible using the key, this is why I think my AI is slightly better in regard to that. However both of those AI’s are better at creating melodies with more notes than mine is.

[1] Klinger, R., & Rudolph, G. (2006). Evolutionary composition of music with learned melody evaluation. In Proceedings of the WSEAS International Conference on Computational Intelligence, Man-Machine Systems and Cybernetics, pp. 234–239, Stevens Point, Wisconsin, USA.

[2] OLIWA, T. M. Genetic algorithms and the abc music notation language for rock music composition. In GECCO ’08: Proceedings of the 10th annual conference on Genetic and evolutionary computation (2008), pp. 1603–1610.