Journal Report 8

10/28/19-11/11/19
Irina Lee
Computer Systems Research Lab
Period 1, White

Daily Log

Monday October 28

I wrote a method to ”prepare” the motif string for the ’finger piece’ method by splitting the string into an array based on commas and turning it into a matrix capable of being parsed by the latter method. The motif can now be successfully fingered by the ’finger piece’ method.

Tuesday October 29

”Believer” is a type of piece that stays in relatively the same hand position most of the way through. I wrote an algorithm that takes advantage of this to expedite the process of fingering. When fingering the motif, the algorithm notes the hand position used to finger the motif and searches for notes in the rest of the piece that can be reached by the hand position dictated by the motif. Using this method, all the notes in ”Believer” are able to be fingered except for the G in the A-G-A section that forms the crossover between the first and second motif appearances. Something to be cautious of is that this code assumes that the hand stays in the same hand position throughout though this will not be true in most cases.

Thursday October 31

My original idea for the most efficient fingering was centered around the idea of minimizing crossovers, but I have realized that certain cases of crossovers provide greater comfort and stability even though they may not be strictly necessary. I wrote code that scans through the music sections looking for potential crossover sections. Potential crossover sections are defined by two ”ones” separated by a single note that is lower in value. My algorithm has now completed an alternate method of fingering for ”Believer.”

Wednesday November 6

I debugged the code for finding motifs and found a new way of parsing out a motif. When attempting to finger arpeggios, I realized that using my current motif wouldn’t work because it tried to find exact matches in the notes while the arpeggio spanned multiple octaves. I came up with a different method of finding the motif: creating an array that stored the differences between a pair of notes for every single pair, and running that array through the regex. This way, the a motif that might occur in different octaves or in a different key can still be found.

Tuesday November 7

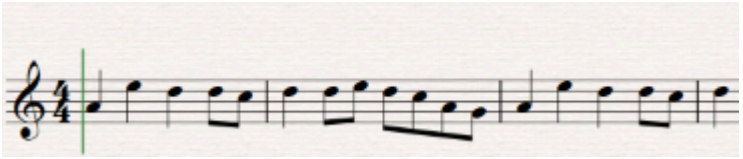
I created an array to keep track of the differences and ran it through my regex to find a match. I converted the match from a string into an array, took the length of the array and used that to find the motif from another array that keeps track of all the notes in the piece.

Timeline

Date	Goal	Met
October 25th	Be able to finger all 12 major arpeggios	The algorithm can correctly identify all indices at which the motif appears
November 1st	Be able to finger the notes between the motif and transition sections correctly	Yes, my algorithm fingers the notes between the motif and transition sections correctly
November 8th	Be able to finger the 7 ”1-2-3” arpeggios	I improved the motif finder algorithm.
November 15th	Be able to finger all 12 arpeggios	–
November 22nd	Expand the current fingering to include the left hand.	–

Reflection

These two weeks brought an interesting departure from the original strategy. Fingering centered around musical motifs provides an alternate strategy that could produce even better results. The excerpt from ”Believer” is one such case.



Motif: ’069’, ’076’, ’074’, ’074’, ’072’, ’074’, ’074’, ’076’, ’074’, ’072’, ’069’, ’067

Indices at which motif appears: [0, 12]

Original Fingering: [[256, 69, 1], [256, 76, 5], [256, 74, 4], [128, 74, 4], [128, 72, 3], [256, 74, 4], [128, 74, 4]] [[128, 76, 5], [128, 74, 4], [128, 72, 3], [128, 69, 2], [128, 67, 1]] [[256, 69, 2], [256, 76, 5], [256, 74, 4], [128, 74, 4], [128, 72, 3], [256, 74, 4]]

Alternative Fingering: [[256, 69, 1], [256, 76, 5], [256, 74, 4], [128, 74, 4], [128, 72, 3], [256, 74, 4], [128, 74, 4]] [[128, 76, 5], [128, 74, 4], [128, 72, 3], [128, 69, 1], [128, 67, 2]] [[256, 69, 1], [256, 76, 5], [256, 74, 4], [128, 74, 4], [128, 72, 3], [256, 74, 4]]