Accompaniment Generation Report

In the lines below I will describe the flow of the work done.

Libraries

Two libraries were chosen for this work. ‘music21’ is for song analysis and ‘mido’ for generating output file of accompaniment.

Song Analysis

To generate fine chords we need to know the key of the song. I have two methods ‘generate_major_keys()’ and ‘generate_minor_keys()’ which generate tables of keys for both scales. Notes in these tables represented in midi notation of seven integers.

Note Numbers										
C	C#	D	D#	E	F	F#	G	G#	A	A#	B
0	1	2	3	4	5	6	7	8	9	10	11

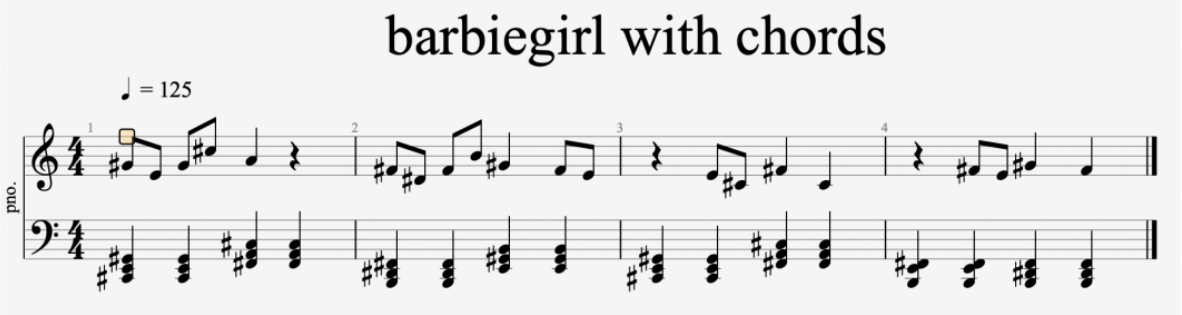
Major Keys	I	ii	iii	IV	V	vi	vii°
C	C	Dm	Em	F	G	Am	B°

So, C major looks like [0, 2, 4, 5, 7, 9, 11] in generated table.

In ‘define_key()’ method I invoke music21 functions which returns scale (major or minor) and tonic note. This is how I define key (line in one of the keys tablres) to work with.

Chords Generaton preparations

We want to generate chords for all quarters of the song. So, we need to know how much chords we need and where to paste them. ‘get_num_of_chords()’ method parses song file and counts messages which separate beats in song. In each beat we have four quarters. Now we know how much chords to generate.



For barbiegirl example we have 4 beats. It means we want to generate 16 chords. I decided to generate array of notes or rests from which each quarter starts. For this example ‘get_notes_on_quarters()’ method returns array of 16 integers (note in MIDI notation) or None values (rests). We want to play chords on rests either, so we need to do something with None values.

To generate chord we need to know the key of the song and the note to play chord with. So, to generate chords on rests we need to feel None values with note integers. ‘fill_rests()’ method replaces None values with adjacent note in the same beat or last note from left beat. Big rests (4 quarters as in the input3) are not filled.

Genetic Algorithm

My genetic algorithm gets note to which we want to generate chord and the key of the song. After roccessing it returns best generated chord.

Main function of genetic algorithm is ‘launch_genetic()’. Firtstly, it generates random population to start with. Secondly, main loop starts. We define best chords of the population with fitness function then I use crossover to generate children from chosen chords. After summing children and parents I apply mutation to add new features to the population. After needed number of generations algorithm returns the best found chord.

Work of the methods of genetic algorithm is described in comments.

Summing up

Let’s now analyse main method of the program: ‘generate_accompaniment()’. Firtsly it reads input file via path. Then it generates array of notes or rests for each qurater. Then if feels rests with

notes. Finally, there is for-loop with chord generation. For each note in notes array genetic algorithm is invoked. It gets note and key of the song as parameters and returns best fitted chord for this position which is appeneded to the new created track in our MIDI file.