NAME

midistats – program to summarize the statistical properties of a midi file

SYNOPSIS

This is a long manual because the program extracts many different parameters from the midi file depending upon the options selected. This manual attempts to describe these parameters and how they are computed. These parameters are formatted in a way so they can be read by other applications such midiexplorer.tcl, runstats.tcl, and numerous Python scripts.

Many of the options were designed specifically for analyzing the percussion track of the midi file. They are described in a separate section below (Other options).

DESCRIPTION

If you run midistats without any options other than the name of the midi input file, it will produce a table of values described here. Each line of output starts with the name of the variable or variable array and the associated values.

ntrks indicates the number of tracks in the midi file.

ppqn the number of midi pulses per quarter note.

keysig the key signature, followed by a major/minor flag, the number of sharps (positive) or flats (negative) in the key, and the beat number where the key signature was found.

trk is followed by the track number for which the following information applies.

program is followed by the channel number and the General Midi Program number.

trkinfo is an array of 19 numbers which indicates the statistical properties of the track of interest. The following data is given:

0 the channel number,
1 the first program assigned to this channel,
2 the number of notes for this channel counting any chords as one note,
3 the number of chord notes excluding the bass note,
4 the sum of the MIDI pitches for all the notes,
5 the sum of the note durations in MIDI pulse units,
6 the number of pitchbend messages,
7 the number of control parameter messages,
8 the number of pressure messages.
9 the number of distinct rhythm patterns for each channel
10 the number of pulses the channel was inactive
11 the minimum pitch value
12 the maximum pitch value
13 the minimum note length in pulses
14 the maximum note length in pulses
15 the number of gaps in the channel
16 the entropy of the pitch class histogram for that channel
17 the number of notes whose pitch were the same as the previous note
18 the number of notes whose pitch changed by less than 4 semitones
19 the number of notes whose pitch changed by 4 or more semitones
20 the average note velocity
21 the standard deviation of the note velocity

(In event of a chords the maximum pitches are compared.)

After processing all the individual tracks, the following information applies to the entire midi file.

npulses is the length of the longest midi track in midi pulse units

tempocmds specifies the number of times the tempo is changed in this file.

pitchbends specifies the total number of pitchbends in this file.

pitchbendin c n specifies the number of pitchbends n in channel c

progs is a list of all the midi programs addressed

progsact the amount of activity for each of the above midi programs. The activity is the sum of the note durations in midi pulse units.

progcolor: is a 17 dimensional vector where each component maps into a specific group of MIDI programs. Some of these groups are, keyboard instruments, brass instruments, wind instruments, and etc. More information can be found in the midiexplorer documentation.

drums is a list of all the percussion instruments (channel 9) that were used.

drumhits indicates the number of notes for each of the above percussion instruments.

pitches is a histogram for the 11 pitch classes (C, C#, D ...B) that occur in the midi file.

key indicates the key of the music, the number of sharps (positive) or flats (negative) in the key signature, and a measure of the confidence in this key signature. The key was estimated from the above pitch histogram by convolving with Craig Sapp's model. The peak of rmaj or rmin (below) indicates the key. A correlation less than 0.4 indicates that the pitch histogram does not follow the histogram of a major or minor scale. (It may be the result of a mixture of two key signatures.)

rmaj the cross correlation coefficients with Craig Sapp's major key model for each of the 11 keys (C, C#, D, ...,B).

rmaj the cross correlation coefficients with Craig Sapp's minor key model for each of the 11 keys (C, C#, D, ...,B).

pitchact is a similar histogram but is weighted by the length of the notes.

chanvol indicates the value of the control volume commands in the midi file for each of the 16 channels. The maximum value is 127. It scales the loudness of the notes (velocity) by its value.

chnact returns the amount of note activity in each channel.

trkact returns the number of notes in each track.

totalrhythmpatterns is the total number of bar rhythm patterns for all channels except the percussion channel.

collisions. Midistats counts the bar rhythm patterns using a hashing function. Presently collisions are ignored so occasionally two distinct rhythm patterns are counted as one.

Midistats prints a number of arrays which may be useful in determining where the music in the track is a melody line or chordal rhythmic support. These arrays indicate the properties for each of the 16 channels. (The percussion channel 9 contains zeros.) In the case same channel occurs in several tracks, these numbers are the totals for all track containing that channel. Here is a description of these properties.

programs: channel to midi program mapping

cnotes: the total number of notes in each channel

nnotes: the number of notes in each channel not including those playing in the same time interval.
nzeros: the number of notes whose previous note was the same pitch
nsteps: the number of notes whose pitch difference with the previous note was less than 4 semitones.
njumps: the number of notes whose pitch difference with the previous note was 4 or more semitones.
rpats: the number of rhythmpatterns for each channels. This is a duplication of data printed previously.
pavg: the average pitch of all the notes for each channel.
spread: the percentage of the track that each channel is active.

If some of the channels appear in more than one track, then some of the above values may be incorrect.

In addition the midistats may return other codes that describe other characteristics. They include

unquantized - the note onsets are not quantized
triplets - 3 notes played in the time of 2 notes are present
qnotes - the rhythm is basically simple
clean_quantization - the note onsets are quantized into 1/4, 1/8, 1/16 time units.
dithered_quantization - small variations in the quantized note onsets.
Lyrics - lyrics are present in the meta data
programcmd - there may be multiple program changes in a midi channel

Other options

It is recommended that you only select one of the options described here as the program was not designed to handle a multiple number of options.

If you run midistats with the -CSV option, it will return the results in a form of comma separated values that can be loaded into a Python panda dataframe. Each line refers to one of the 16 midi channels. The following Python 3 code illustrates how you would load the midistats output into a dataframe.

```
import pandas as pd
import io
import subprocess
cmd = ("midistats", "-CSV" , inputmidifilepath)
process = subprocess.Popen(cmd, stdout=subprocess.PIPE)
csv =  io.StringIO(process.stdout.read().decode())
df = pd.read_csv(csv)
```

where inputmidifilepath is the path to the midi file that you are using. (eg. 'clean_midi/Zero/Chi sei.mid')

The MIDI file devotes channel 9 to the percussion instruments and over 60 percussion instruments are defined in the MIDI standard. Though there is a lot of diversity in the percussion track, for most MIDI files only the first 10 or so percussion instruments are important in defining the character of the track. The program Midiexplorer has various tools for exposing the percussion channel which are described in the documentation. The goal here is to find the essential characteristics of the percussion track which distinguishes the MIDI files. This is attempted in the program midistats. Here is a short description.

A number of experimental tools for analyzing the percussion channel (track) were introduced into midistats and are accessible through the runtime arguments. When these tools are used in a script which runs through a collection of midi files, you can build a database of percussion descriptors.

OPTIONS

-corestats
outputs a line with 5 numbers separated by tabs. eg
1 8 384 4057 375
It returns the number of tracks, the number of channels, the number of divisions per quarter note beat (ppqn), the number of note onsets in the midi file, and the maximum number of quarter note beats in midi file.

-pulseanalysis
counts the number of note onsets as a function of its onset time relative to a beat, grouping them into 12

intervals and returns the result as a discrete probability density function. Generally, the distribution consists of a couple of peaks corresponding to quarter notes or eigth notes. If the distribution is flat, it indicates that the times of the note occurrences have not been quantized into beats and fractions. Here is a sample output.
0.349,0.000,0.000,0.160,0.000,0.000,0.298,0.000,0.000,0.191,0.000,0.000

-panal
Counts the number of note onsets for each percussion instrument. The first number is the code (pitch) of the instrument, the second number is the number of occurrences. eg.
35 337 37 16 38 432 39 208 40 231 42 1088 46 384 49 42 54 1104 57 5 70 1040 85 16

-ppatfor n
where n is the code number of the percussion instrument. Each beat is represented by a 4 bit number where the position of the on-bit indicates the time in the beat when the drum onset occurs. The bits are ordered from left to right (higher order bits to lower order bits). This is the order of bits that you would expect in a time series. Thus 0 indicates that there was no note onset in that beat, 1 indicates a note onset at the end of the beat, 4 indicates a note onset in the middle of the beat, and etc. The function returns a string of numbers ranging from 0 to 7 indicating the presence of note onsets for the selected percussion instrument for the sequence of beats in the midi file. Here is a truncated sample of the output.
11 71 c1 1 11 51 f1 c1 1 11 51 e1 1 11 d1 c1 31 1 1 1 21 b1 d1 d1 51
51 51 51 51 41 71 c1 11 1 11 41 11 1 11 41 11 1 11 41 11 31 51 41 11

-ppat
midistats attempts to find two percussion instruments in the midi file which come closest to acting as the bass drum and snare drum. If it is unsuccessful, it returns a message of its failue. Otherwise, encodes the position of these drum onsets in a 8 bit byte for each quarter note beat in the midi file. The lower (right) 4 bits encode the bass drum and the higher (left) 4 bits encode the snare drum in the same manner as described above for -ppatfor. The integers are printed in hexadecimal.
02 88 20 02 a0 08 80 02 82 08 80 02 82 08 80 02 82 08 80 02 82 08
80 02 82 08 80 02 82 08 80 02 82 08 20 a0 0a 08 80 02 82 08 80 02

-ppathist
computes and displays the histogram of the values that would appear when running the -ppat. eg.
bass 35 337
snare 38 432
1 (0.1) 64 32 (2.0) 8 33 (2.1) 136 144 (9.0) 8 145 (9.1) 136
The bass percussion code, the number of onsets, and the snare percussion code and the number of onsets are given in the first two lines. In the next line the number of occurrences of each value in the -ppat listing is given. The number in parentheses splits the two 4-bit values with a period. Thus 33 = (2*16 + 1).

-pitchclass
Returns the pitch class distribution for the entire midi file.

-nseqfor n
Note sequence for channel n. This option produces a string of bytes indicating the presence of a note in a time unit corresponding to an eigth note. Thus each quarter note beat is represented by two bytes. The pitch class is represented by the line number on the staff, where 0 is C. Thus the notes on a scale are represented by 7 numbers, and sharps and flats are ignored. The line number is then converted to a bit position in the byte, so that the pitch classes are represented by the numbers 1,2,4,8, and etc. A chord of consisting of two note onsets would set two of the corresponding bits. If we were to represent the full chromatic scale consisting of 12 pitches, then we would require two-byte integers or twice of much memory.
Though the pitch resolution is not sufficient to distinguish major or minor chords, it should be sufficient to be identify some repeating patterns.

-nseq
Same as above except it is applied to all channels except the percussion channel.

-nseqtokens Returns the number of distinct sequence elements for each channel. The channel number and number of distinct elements separated by a comma is returned in a tab separated list for all active channels except the percussion channel. Here is an example.
2,3 3,4 4,11 5,6 6,3 7,3 8,6 9,3 11,2 12,1

-ver (version number)

AUTHOR

Seymour Shlien <fy733@ncf.ca>