Machine Discovery Final Project - Report

Team Members

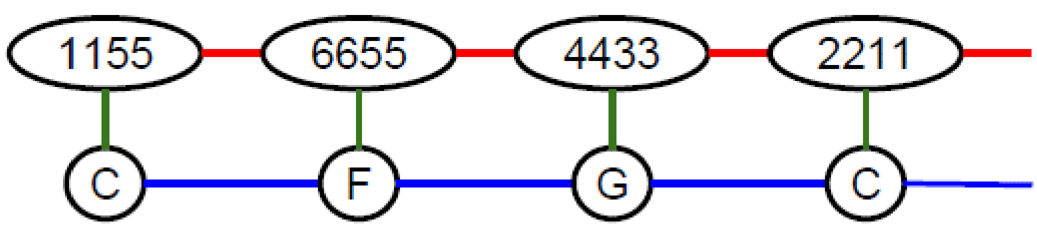
- B03902010 耿宗揚
  - model design, data conversion, model training, algorithm optimization
- B03902015 簡瑋德
  - data conversion, model training, report composition
- B03902086 李鈺昇
  - algorithm implementation

Goal

Specific Piano-Sonatina-Like Music Period Composition Using MRF

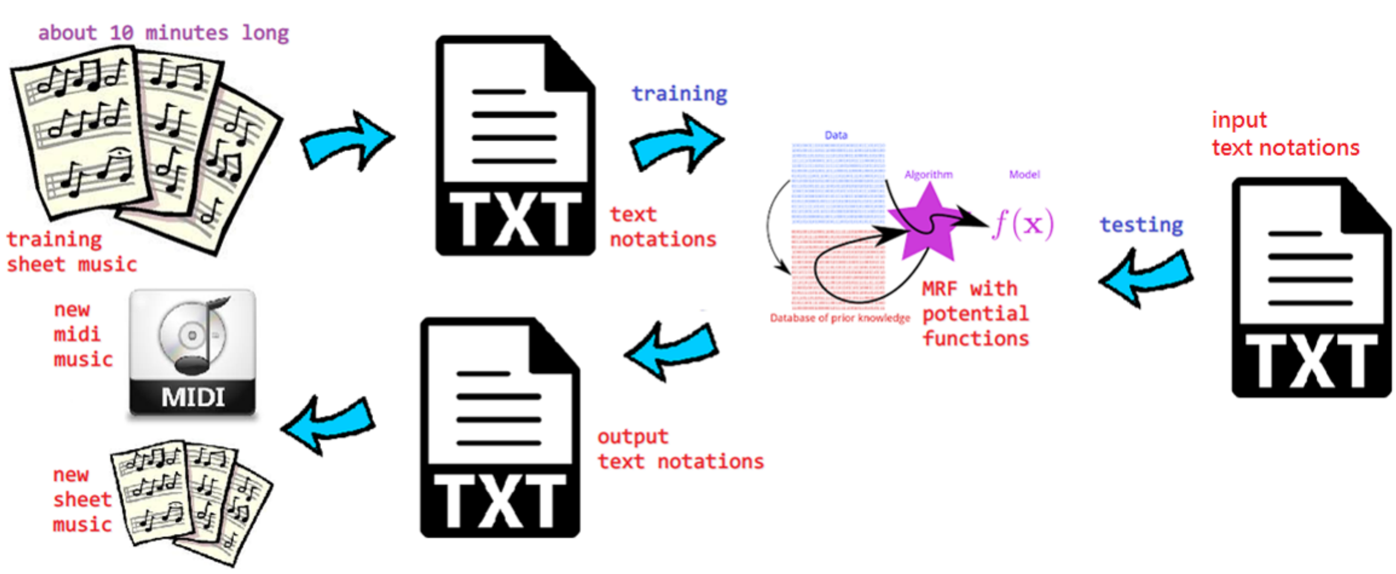
- Machine completes a music period given fixed melody fragments and chords at some beats.

- We use MRF to represent the period, for example



- Converted goal: assign values (output) to variables in a 2-by-L grid MRF, given some variables (input) fixed

Framework



1. **Manually** convert the piano sheets into pre-defined text format
2. Derive potential functions from fractional counts on data - `composer.py`
3. Parse the input file - `composer.py`
4. Use Viterbi algorithm to complete the music period given by the input and then generate the output - `viterbiOn2ByLMRF.py`
5. Generate the pdf and midi file, given the output text notations - `Data2Sheet.java` and `txt2Midi.py`

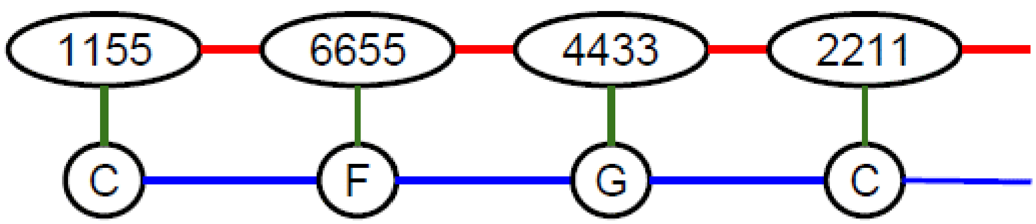
Training Data

- About 1400 units from Clementi op36
- Source: $\text{imslp. org/wiki/12\_Sonatinas\_(Clementi, \_Muzio)}$
- For example, the following period is converted to the notations


```
1  @ x2
2  C c6 - e6 c6
3  C g5 - g5 -
4  C c6 - e6 c6
5  C g5 - g6 -
6  C f6 e6 d6 c6
7  C b5 c6 b5 c6
```

  - Each line in the text file is can be viewed as a unit, which is defined to be 4 basic notes plus a chord.
  - In the example, `C` is the chord of the first unit.
  - `c6` is the first note of the first unit, while `e6` is the third note of the first unit.

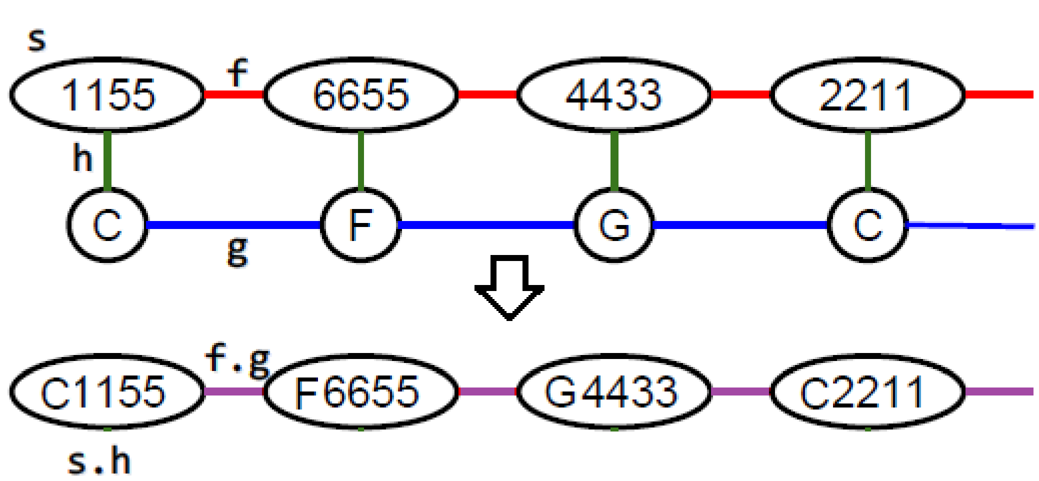
Potential Functions



- First, we define $m_{i,j}$ as the j^{th} note of the i^{th} unit
- Potential between Melody nodes: $\phi(M_i, M_{i+1}) = \mathbb{P}(m_{i,3}, m_{i+1,0})$
- Potential between Chord nodes: $\phi(C_i, C_{i+1}) = \mathbb{P}(C_i, C_{i+1})$
- Potential between Melody and Chord
  $\phi(M_i, C_i) = \mathbb{P}(M_i, C_i) = \mathbb{P}(C_i)\mathbb{P}(M_i|C_i) = \mathbb{P}(C_i)\prod_{j=0}^{3}\mathbb{P}(m_{i,j}|C_i)$
- Potential within the Molody: $\phi(M_i) = \prod_{j=0}^{2}\mathbb{P}(m_{i,j}, m_{i,j+1})$
- All the probabilities are derived from fractional counts on data.

Viterbi Algorithm

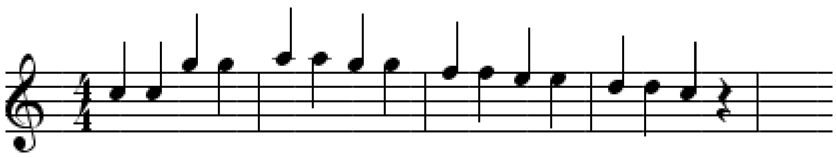
- Given a music preiod with some empty nodes, we would like to fill those nodes based on the potential functions.
- We simplify the 2-by-L grid MRF by combining the potentials.



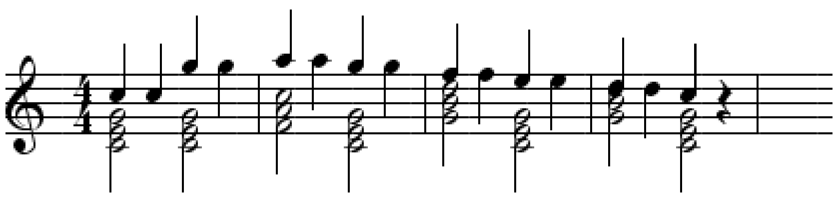
- Domain of converted node is $M \times C$ where M is the domain of melody node and C the domain of chord node.
- In this way, we are able to use variable elimination with dynamic programming to do Viterbi on the converted MRF.
- Time complexity: $O(|M|^2|C|^2L)$, where L is the number of units

Experiments

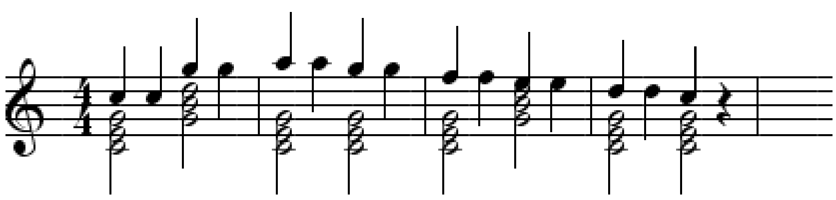
- We perform some experiments to compare our model to a naive one, which is done by sampling the chord and choosing the top-N frequent melody to fill the empty nodes.
- Experiment 1
  - Input



  - Output of our model



  - Output of naive model



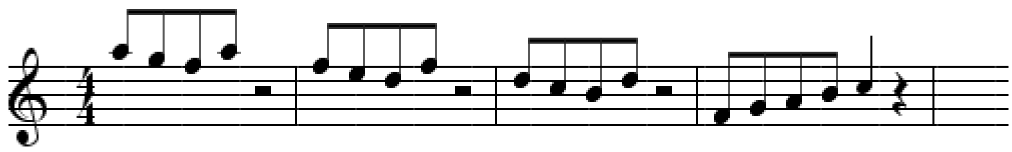
- Experiment 2
  - Input

  - Output of our model

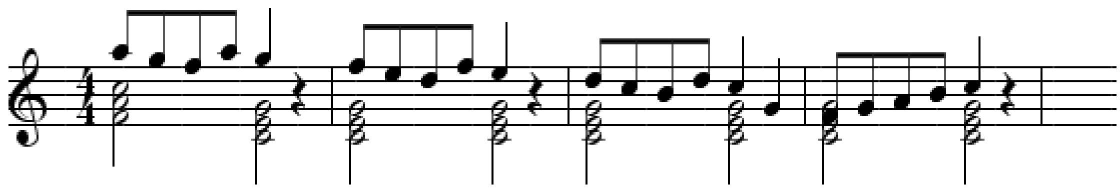
  - Output of naive model

- Experiment 3

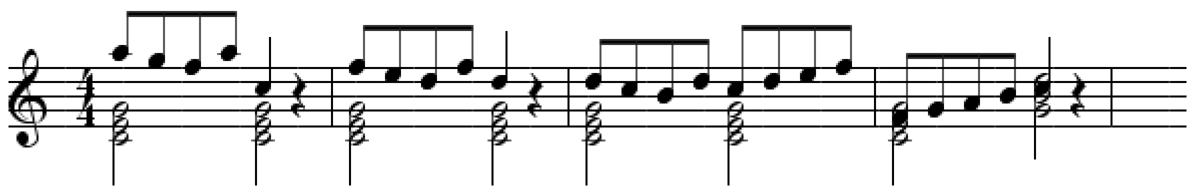
- Input



- Output of our model



- Output of naive model



- These experiments show that our model does create smoother music period than the naive one does.

Remarks

- Out model is good at connecting peaks and valleys of a melody line, but it rarely creates them. This is an effect of maximum likelihood.
- However, peaks and valleys of a song are often the parts by which listeners are impressed.

Third-part Libraries

- Python-mido (https://mido.readthedocs.io/en/latest/)
- Java-abc4j (https://code.google.com/archive/p/abc4j/)