Composing Counterpoint Music With Variable Neighborhood Search

D. Herremans & K. Sørensen
EURO, July 8-11, 2012
Overview

Computer aided composing (CAC)

Variable Neigborhood Search

Experiments & Results

Implementation

Conclusion
Computer aided composing (CAC)

- Music $\rightarrow$ combination of notes
- “Good” music $\rightarrow$ fits a style as well as possible
- Formalized and quantified “rules” of a style $\rightarrow$ objective function
Counterpoint

- Polyphonic classical music
- Inspired Bach, Haydn, ...
- One of the most formally defined musical styles
  → Rules written by Fux in 1725
1st species counterpoint

- Counterpoint & Cantus firmus

- Represented as 2 vectors with midi values

\[
\begin{bmatrix}
60 & 65 & 64 & 62 & 60 & 64 & 65 & 67 & 67 & 69 & 62 & 64 & 64 & 60 & 59 & 60
\end{bmatrix}
\]
5th species counterpoint

- Counterpoint & Cantus firmus

- Represented as a vector of note objects, each with:
  - Pitch: midi value
  - Duration
  - Beat number
  - Measure number
  - Tied?
Quantifying musical quality

Examples of rules:

- Each large leap should be followed by stepwise motion in the opposite direction
- Half notes should always be consonant on the first beat, unless they are suspended and continued stepwise and downward
- All perfect intervals should be approached by contrary or oblique motion

→ 19 vertical and 19 horizontal subscores between 0 and 1
Quantifying musical quality

\[ f(s) = \sum_{i=0}^{19} a_i \cdot \text{subscore}_i^H(s) + \sum_{j=0}^{19} b_j \cdot \text{subscore}_j^V(s) \]  

(1)

horizontal aspect  
vertical aspect
Quantifying musical quality

- Weights $a_i$ and $b_j$
- Specified at input
  - Emphasize subscore from start
- Adaptive weights mechanism
  - Increase weight of subscore with highest value
  - Keeps the search in the right direction
Variable Neighborhood Search

- Local search with 3 neighborhoods
- Selection
  - Steepest descent
  - Based on adaptive score $f^a(s)$

<table>
<thead>
<tr>
<th>$N_i$</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N_{sw}$</td>
<td>Swap</td>
<td>Swap two notes</td>
</tr>
<tr>
<td>$N_{c1}$</td>
<td>Change1</td>
<td>Change one note</td>
</tr>
<tr>
<td>$N_{c2}$</td>
<td>Change2</td>
<td>Change two notes</td>
</tr>
</tbody>
</table>
Variable Neighborhood Search

- Excluded fragments
  - Tabu list
  - Infeasible
- Perturbation
  - Change r% of the notes randomly
- Adaptive weights mechanism
- Update best solution $s_{best}$, based on original score $f(s_{best})$
### Experiments & Results

- Full factorial experiment, \( n=2304 \)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Values</th>
<th>Nr. of levels</th>
</tr>
</thead>
<tbody>
<tr>
<td>( N_{sw} ) - Swap</td>
<td>on with ( tt_{sw}=0, \frac{1}{16}, \frac{1}{8} ), off</td>
<td>4</td>
</tr>
<tr>
<td>( N_{c1} ) - Change1</td>
<td>on with ( tt_{c1}=0, \frac{1}{16}, \frac{1}{8} ), off</td>
<td>4</td>
</tr>
<tr>
<td>( N_{c2} ) - Change2</td>
<td>on with ( tt_{c2}=0, \frac{1}{16}, \frac{1}{8} ), off</td>
<td>4</td>
</tr>
<tr>
<td>Random move</td>
<td>( \frac{1}{4} ) changed, ( \frac{1}{8} ) changed, off</td>
<td>3</td>
</tr>
<tr>
<td>Adaptive weights</td>
<td>on, off</td>
<td>2</td>
</tr>
<tr>
<td>Max. iterations</td>
<td>5, 20, 50</td>
<td>3</td>
</tr>
<tr>
<td>Length of music</td>
<td>16, 32 measures</td>
<td>2</td>
</tr>
</tbody>
</table>
Experiments & Results

- Multi-Way ANOVA model with interaction effects, using R
- $R^2 = 0.98$

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Df</th>
<th>F value</th>
<th>Prob ($&gt; F$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N_{c1}$</td>
<td>1</td>
<td>9886.2323</td>
<td>$&lt; 2.2e^{-16}$</td>
</tr>
<tr>
<td>$N_{c2}$</td>
<td>1</td>
<td>15690.7234</td>
<td>$&lt; 2.2e^{-16}$</td>
</tr>
<tr>
<td>$N_{sw}$</td>
<td>1</td>
<td>3909.2959</td>
<td>$&lt; 2.2e^{-16}$</td>
</tr>
<tr>
<td>randsize</td>
<td>2</td>
<td>1110.1724</td>
<td>$&lt; 2.2e^{-16}$</td>
</tr>
<tr>
<td>maxiters</td>
<td>2</td>
<td>322.6488</td>
<td>$&lt; 2.2e^{-16}$</td>
</tr>
<tr>
<td>length</td>
<td>1</td>
<td>165.6053</td>
<td>$&lt; 2.2e^{-16}$</td>
</tr>
<tr>
<td>adj. weights</td>
<td>1</td>
<td>4.0298</td>
<td>0.0448367</td>
</tr>
<tr>
<td>$tt_{c1}$</td>
<td>2</td>
<td>2.2575</td>
<td>0.1048791</td>
</tr>
<tr>
<td>$tt_{c2}$</td>
<td>2</td>
<td>8.271</td>
<td>0.0002646</td>
</tr>
<tr>
<td>$tt_{sw}$</td>
<td>2</td>
<td>3.2447</td>
<td>0.0391833</td>
</tr>
</tbody>
</table>
Experiments & Results

- Mean plot for the size of the random jump
Optimal parameter settings

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N_{sw}$</td>
<td>on with $tt_{sw} = \frac{1}{16}$</td>
</tr>
<tr>
<td>$N_{c1}$</td>
<td>on with $tt_{c1} = \frac{1}{16}$</td>
</tr>
<tr>
<td>$N_{c2}$</td>
<td>on with $tt_{c2} = \frac{1}{16}$</td>
</tr>
<tr>
<td>Random move</td>
<td>$\frac{1}{8}$ changed</td>
</tr>
<tr>
<td>Adaptive weights</td>
<td>on</td>
</tr>
<tr>
<td>Max. number of iterations</td>
<td>50</td>
</tr>
</tbody>
</table>
Implementation

- C++ → VNS
- JavaScript using the QtScript engine → MuseScore plugin
- Input:
  - Key (i.e., G# minor)
  - Weights for each subscores
  - VNS parameters
- Result: MusicXML
Results

- Example of a generated fragment with score 0.556776.
Conclusion

The fifth species counterpoint rules have been quantified and an efficient algorithm has been implemented to compose this style of music.

Future research:
- More complex music:
  - Different styles
  - More parts
  - Theme
- Analyse DB of existing music and extract composer characteristics
- Android App for continuously generating music
Composing Counterpoint Music With Variable Neighborhood Search

D. Herremans & K. Sørensen
EURO, July 8-11, 2012

dorien.herremans@ua.ac.be