University of Antwerp
Operations Research Group
ANT/OR

Composing Counterpoint Music With Variable Neighborhood Search

D. Herremans & K. Sörensen

ORBEL26, February 2-3, 2012

Overview

Computer aided composing (CAC)

Variable Neigborhood Search

Experiments & Results

Implementation

Conclusion

Computer aided composing (CAC)

Composing music = combinatorial optimization problem

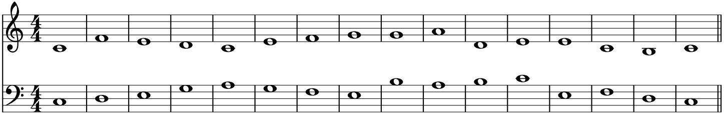
- Music $\rightarrow$ combination of notes
- "Good" music $\rightarrow$ fits a style as well as possible
- Formalized and quantified "rules" of a style $\rightarrow$ objective function

Counterpoint

- Polyphonic classical music
- Inspired Bach, Haydn,...
- One of the most formally defined musical styles
  $\rightarrow$ 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

$$f(s) = \underbrace{\sum_i a_i.\mathsf{subscore}_i^H(s)}_{\text{horizontal aspect}} + \underbrace{\sum_j b_j.\mathsf{subscore}_j^V(s)}_{\text{vertical aspect}} \qquad (1)$$

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

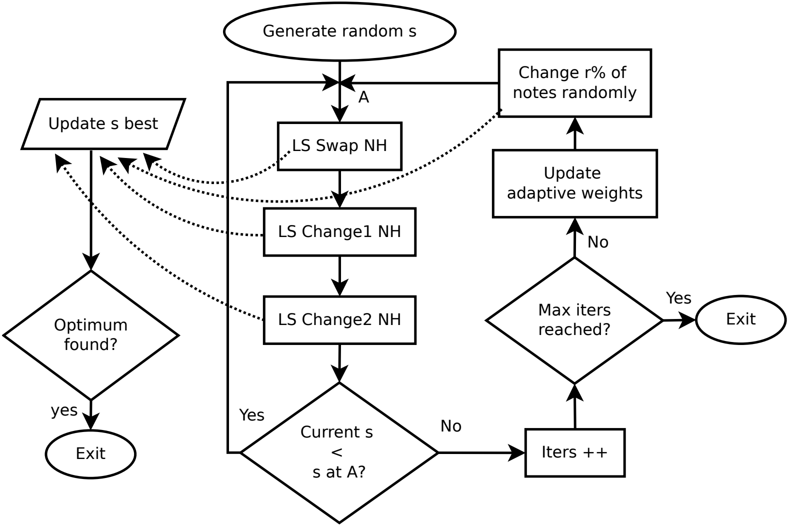
Variable Neigborhood Search

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

N_i	Name	Description
N_{sw}	Swap	Swap two notes
N_{c1}	Change1	Change one note
N_{c2}	Change2	Change two notes

Variable Neigborhood Search

- Excluded framents
  - Tabu list
  - Infeasible
- Perturbation
  - Change r% of the notes randomly
- Adaptive weights mechanism
- Update best solution s_{best}, based on original score $f(s_{\text{best}})$



Generate random s
A
Update s best
LS Swap NH
LS Change1 NH
LS Change2 NH
Change r% of notes randomly
Update adaptive weights
No
Max iters reached?
Yes
Exit
Optimum found?
yes
Exit
Yes
Current s < s at A?
No
Iters ++

Experiments & Results

- Full factorial experiment, n=2304

Parameter	Values	Nr. of levels
N_{sw} - Swap	on with tt_{sw}=0, $tt_{sw}=\frac{1}{16}$, $tt_{sw}=\frac{1}{8}$, off	4
N_{c1} - Change1	on with tt_{c1}=0, $tt_{c1}=\frac{1}{16}$, $tt_{c1}=\frac{1}{8}$, off	4
N_{c2} - Change2	on with tt_{c2}=0, $tt_{c2}=\frac{1}{16}$, $tt_{c2}=\frac{1}{8}$, off	4
Random move	$\frac{1}{4}$ changed, $\frac{1}{8}$ changed, off	3
Adaptive weights	on, off	2
Max. iterations	5, 20, 50	3
Length of music	16, 32 measures	2

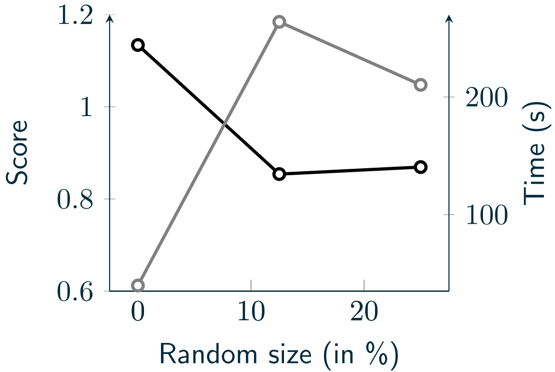
Experiments & Results

- Multi-Way ANOVA model with interaction effects, using R
- $R^2 = 0.9642$

Parameter	Df	Sum Sq	Mean Sq	F value	Prob $(> F)$
N_{c1}	1	155.73	155.73	4857.6450	$< 2.2e^{-16}$
N_{c2}	1	238.40	238.40	7436.5417	$< 2.2e^{-16}$
N_{sw}	1	69.13	69.13	2156.2797	$< 2.2e^{-16}$
randsize	2	38.09	19.05	594.1391	$< 2.2e^{-16}$
maxiters	2	9.30	4.65	145.0207	$< 2.2e^{-16}$
tt_{c1}	2	0.05	0.02	0.7588	0.468333
tt_{c2}	2	0.15	0.08	2.3595	0.094707
tt_{sw}	2	0.08	0.04	1.3150	0.268681
adj. weights	1	0.30	0.30	9.3497	0.002257

Experiments & Results

- Mean plot for the size of the random jump



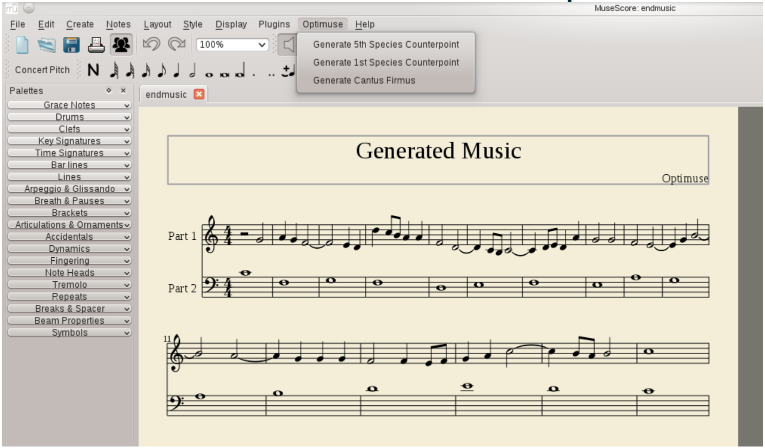
Optimal parameter settings

Parameter	Value
N_{sw} - Swap	on with tt_{sw}=0
N_{c1} - Change1	on with $tt_{c1}=\frac{1}{4}$
N_{c2} - Change2	on with $tt_{c2}=\frac{1}{2}$
Random move	$\frac{1}{8}$ changed
Adaptive weights	on
Max. number of iterations	50

Implementation

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

Implementation



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
- Analyse DB of existing music and extract composer characteristics
- Compare the VNS to other algorithms, e.g. genetic algorithm

University of Antwerp
Operations Research Group
ANT/OR

Composing Counterpoint Music With Variable Neighborhood Search

D. Herremans & K. Sörensen

ORBEL26, February 2-3, 2012