A variable neighbourhood search algorithm to generate first species counterpoint musical scores

Dorien Herremans

23.11.11 - Doctoral Day

Overview

Computer aided composing (CAC)

Variable Neigborhood Search

Experiments & Results

Implementation

Conclusion

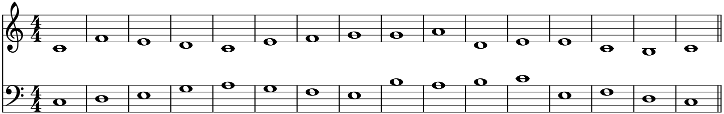
Computer aided composing (CAC)

Composing music = combinatorial optimization problem

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

1st species counterpoint

- Counterpoint & Cantus firmus



- Represented as 2 vectors with midi values

  $\left[\, 60\ 65\ 64\ 62\ 60\ 64\ 65\ 67\ 67\ 69\ 62\ 64\ 64\ 60\ 59\ 60 \,\right]$

$\rightarrow$ Formal rules written by Fux in 1725

Quantifying musical quality

Examples of rules:

- Each large leap should be followed by stepwise motion in the opposite direction
- Only consonant intervals are allowed
- The climax should be melodically consonant with the tonic
- All perfect intervals should be approached by contrary or oblique motion

$\rightarrow$ 15 vertical and 18 horizontal subscores between 0 and 1

Quantifying musical quality

$$f_{\mathsf{CF}}(s) = \underbrace{\sum_i a_i.\mathsf{subscore}_i^H(s)}_{\text{horizontal aspect CF}} \tag{1}$$

$$f_{\mathsf{CP}}(s) = \underbrace{\sum_i a_i.\mathsf{subscore}_i^H(s)}_{\text{horizontal aspect CP}} + \underbrace{\sum_j b_j.\mathsf{subscore}_j^V(s)}_{\text{vertical aspect CP}} \tag{2}$$

$$f(s) = f_{\mathsf{CF}}(s) + f_{\mathsf{CP}}(s) \tag{3}$$

Optimization methods

- Exact methods
  - The *best* solution
  - E.g. exhaustive enumeration
  - 16 notes with 14 different notes $\rightarrow 14^{16}$ possibilities $\rightarrow$ exponential
- Heuristic methods
  - A *good* solution
  - 'Rules of thumb'
  - Fast

$\rightarrow$ Metaheuristics

Metaheuristics

Framework that provides guidelines for the development of problem specific solution methods

→ Variable Neighborhood Search

Variable Neigborhood Search (VNS)

1. Cantus firmus
2. Counterpoint

$\rightarrow$ Same algorithm, different objective function

Variable Neigborhood Search (VNS)

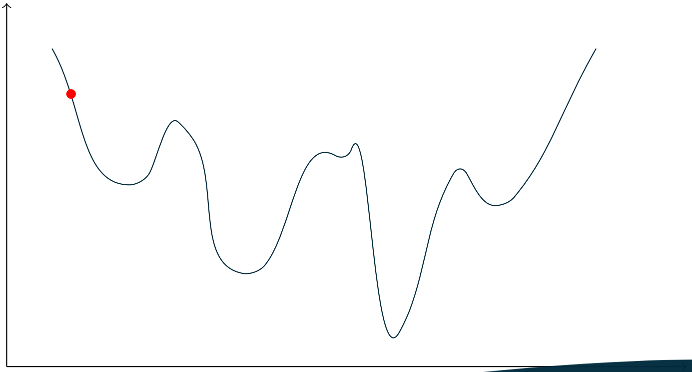
- Local search: make small changes (moves) to a solution to go from one solution to the next.
- Neighborhood $N(x)$: set of all solutions that can be reached from a given solution by move x

N_i	Name	Description
N_1	Swap	Swap two notes
N_2	Change1	Change one note
N_3	Change2	Change two notes

- Choose the best solution from the neighborhood

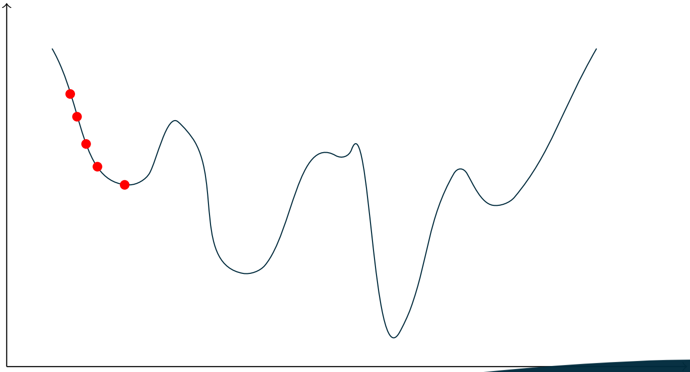
Variable neighbourhood search

- Start from an initial feasible musical fragment



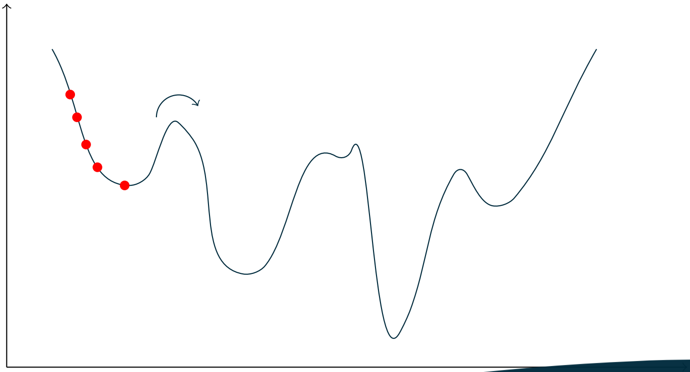
Variable neighbourhood search

- Iterate over the neighborhoods



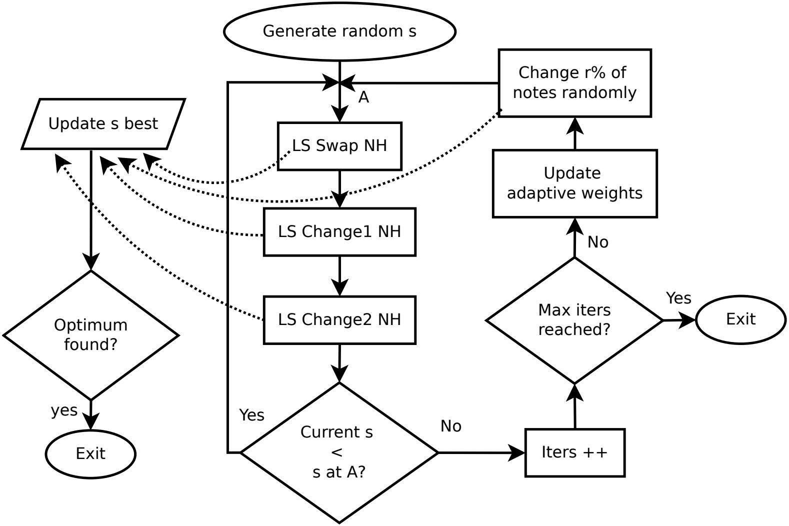
Variable neighbourhood search

- perturbation: change $x\%$ of the notes randomly



Components of the algorithm

- Local search with 3 neighborhoods
- Perturbation: escape from local optima
- Tabu list: avoid circles
- Adaptive weights mechanism
  $\rightarrow$ Increase weight of subscore with highest value
  $\rightarrow$ Keeps the search in the right direction



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

Experiments & Results

- Full factorial experiment, n= 4068

Parameter	Values	Nr. of levels
N_1 - Swap	on with tt_1=0, $tt_1=\frac{1}{4}$, $tt_1=\frac{1}{2}$, off	4
N_2 - Change1	on with tt_2=0, $tt_2=\frac{1}{4}$, $tt_2=\frac{1}{2}$, off	4
N_3 - Change2	on with tt_3=0, $tt_3=\frac{1}{4}$, $tt_3=\frac{1}{2}$, off	4
Random move	$\frac{1}{4}$ changed, $\frac{1}{8}$ changed, off	3
Adaptive weights	on, off	2
Max. iterations	10, 50, 100	3
Length of music	16, 32, 48, 64 notes	4

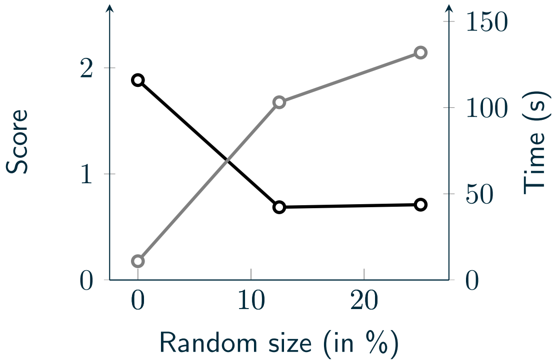
Experiments & Results

- Multi-Way ANOVA model with interaction effects, using R
- $R^2 = 0.9122$

Parameter	Df	Sum Sq	Mean Sq	F value	Prob ($> F$)
N_1	1	323.99	323.99	1173.4292	$< 2.2e^{-16}$ *
N_2	1	723.12	723.12	2618.9755	$< 2.2e^{-16}$ *
N_3	1	1794.21	1794.21	6498.1957	$< 2.2e^{-16}$ *
randsize	2	1441.36	720.68	2610.1349	$< 2.2e^{-16}$ *
iters	2	61.69	30.84	111.7095	$< 2.2e^{-16}$ *
tt_1	2	0.76	0.38	1.3815	0.2513093
tt_2	2	4.17	2.09	7.5519	0.0005321 *
tt_3	2	104.13	52.07	188.5756	$< 2.2e^{-16}$ *
adj. weights	1	5.13	5.13	18.5697	$1.675e^{-05}$ *

Experiments & Results

- Mean plot for size of random jump



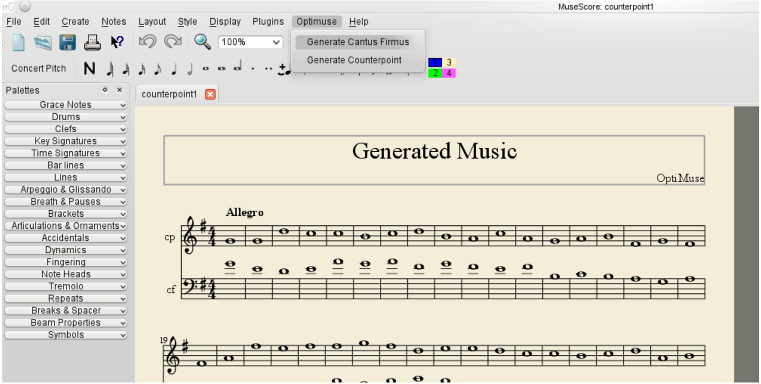
Optimal parameter settings

Parameter	Value
N_1 - Swap	on with tt_1=0
N_2 - Change1	on with $tt_2=\frac{1}{4}$
N_3 - Change2	on with $tt_3=\frac{1}{2}$
Random move	$\frac{1}{8}$ changed
Adaptive weights	on
Max. number of iterations	100
Length of music	64 notes

Implementation

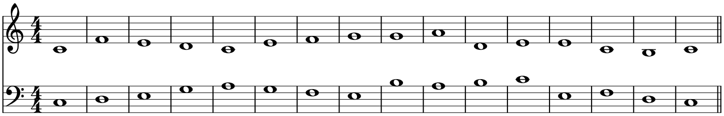
- C++ $\rightarrow$ VNS
- JavaScript using the QtScript engine $\rightarrow$ MuseScore plugin
- Input:
  - Key (i.e., G# minor)
  - Weights for each subscores
  - VNS parameters
- Result: MusicXML

Implementation



Results

- Example of counterpoint with score of 0.371394



Conclusion

The first 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
  - Rythmic component
  - More parts
- Analyse DB of existing music and extract composer characteristics
- Compare the algorithm to others, e.g. genetic algorithm

A variable neighbourhood search algorithm to generate first species counterpoint musical scores

Dorien Herremans

23.11.11 - Doctoral Day