Evolving Palestrinian counterpoint with an EA

Søren Tjagvad Madsen

Abstract—**This paper presents a novel approach to make computer-generated Palestrina-style counterpoint. It uses an EA and a knowledge base as fitness function.**

The idea is tested with an implementation of a subset of the species counterpoint rules as defined by Knud Jeppesen. That involves melodic rules and harmonic rules. Experiments have been done with 1st, 2nd and 4th species, which results in acceptable "well-working" counterpoints.

The paper also suggests a way use the knowledge base to evolve music beyond the species counterpoint.

1 Introduction

Some approaches have been made on automatic generation of counterpoint. Farbood [1] describes a probabilistic way of doing first species counterpoint. The article also describes prior works on the subject. None of them uses an EA as the search algorithm.

Knud Jeppesen, a danish professor in music, has done an excellent book on Palestrina-style counterpoint, based on older studies of the subject. The book is describing the subject melody in great detail. Jeppesen has studied the works by Palestrina, and found a way to describe the music - which has resulted in the so called species counterpoint.

From the music he has derived a quite large set of "rules" or tendencies which the music follows more or less. Some of the rules are more strict than others, and some are just describing pleasant elements, which the style tries to incorporate.

2 Two part species counterpoint

The two part species counterpoint are all concerned about adding a voice to a given cantus firmus (fixed melody). The species counterpoint is an exercise in making larger musical pieces. The rules are fundamental for the larger scale music as well. There are 5 species for 2 part music. The cantus firmus (the given melody) is in all species in whole notes.

The tasks of the species are (in short) defined like this:

1. Add a counterpoint in whole notes. Consonance everywhere.
2. Add a counterpoint in half notes. Dissonances can occur on weak beats.
3. Add a counterpoint in quarter notes.
4. Add a counterpoint in half notes. Dissonances can occur on strong beats.
5. Add a counterpoint in free rhythm, according to the rules in the previous species.

Besides the species rules, which are mainly concerning harmony, there are some overall melodic rules which are also to be followed.

As indicated in the 5th species, the rules are building blocks for making larger music. They say something about the correlation between a pair voices. The rules apply in different settings, and together they deal with all cases.

3 The evolutionary algorithm

The evolutionary algorithm is a search algorithm. I will use it to search for the best counterpoint among many possibly which are correct, according to the musical style.

The idea is to maintain a population of counterpoints for the given cantus firmus. Each counterpoint is evaluated according to the implemented rules given by Jeppesen [2]. So I start with some random music and then evaluates it with the set af rules which apply. Only the best individuals are kept for the next epoch. They are recombined with each other and also mutated a bit before the next evaluation. The idea is to evolve a counterpoint which adheres to all the rules. This counterpoint should then hopefully be worth listening to!

3.1 Implementation details

I bring an overview of the algorithm:

- Population of counterpoints. The counterpoints are initialised with notes in the mode corresponding to the cantus firmus and with rhythmic values according to the wanted species.
- Evaluation:
  - Melodic contour and quality.
  - Species-rules according to the given cantus firmus.
- Tournamentselection. The best is kept in a safe place!
- Crossover: single- and doublepoint.
- Mutation: move a note up/down an interval (second, third or fourth) (in the correct mode)

To be more explicit: two counterpoints can be crossed in one or two points like the the crossover in traditional binary encoded genes (see [3]). For example a new (crossed) melody can consist of the start of the first parent and the end of the other parent.

3.2 The fitness function

The hard part of this algorithm (where to put the most effort) is to code the fitness evaluation.

I try to split the evaluation rules in two categories: Those concerning melodic contour and those concerning harmony. The melodic rules for the counterpoint are not correlated with the cantus firmus whereas the species rules are dealing with the two melodies sounding together.

I haven't implemented all rules in [2], but I began with the counterpoint for 2 part music. To give an impression of the rules, I will now introduce some.

Some fundamental rules

The counterpoint must not be too far away from the cantus firmus (optimally not more than an octave plus a third (decim)). Not all jumps are legal for the counterpoint for example the tritone and the seventh. In some modes, where the seventh in the scale is small (as in dorian and mixolydian), it should be altered in the cadence (leading tone). (For example c in d-dorian becomes c# in the cadence). The leading tone should be introduced stepwise or in a descending third – never by jumping up to it.

First species rules

Only consonant intervals are allowed (third, fifth, sixth, octave, decim). Beginning and ending: only true consonance (fifth, octave). Unison only allowed in first and last measure. Parallel and hidden parallel fifths or octaves (where both voices move in the same direction into the interval) is not allowed. The cantus firmus and the counterpoint are not allowed to move in parallel thirds or sixths for too long time (not more than 4 whole notes), since the counterpoint then loses it's individuality. If both voices jump in the same direction, none of them must jump more than a fourth (except an octave).

The most beautiful thing in this species is countermovement. It is not a requirement, but one should try to do so whenever possible.

Melodic rules

The melodic rules are not so important when the music moves slow as in first species (whole notes). But if possible, we try to keep them. I have implemented a few melodic rules. The melody should not shift direction all the time. If it shifts direction 4 times in a row it is an error. Larger steps should come before smaller when the melodic curve is going up, and smaller before the larger when the curve is going down. (There are some allowed special cases.) The top note (climax) should be unique, and should not be in the beginning of the phrase or at the very end. It is a good idea to have the climax about 1/3 before the end of the phrase.

Second species rules

Dissonance is allowed on weak beats, but only if they are introduced stepwise. Consonance is allowed everywhere. Unison only in first

and last measure. Parallels between the strong beats should be avoided.

Fourth species rules

Dissonance (the intervals fourth (4) and seventh (7)) is allowed on strong beat, if it is prepared from the previous measure as a consonance (3, 5, 6, 8 or 10) and if it is resolved on the next weak beat one step down to a non-true consonance (3, 6 or 10).

If there is no dissonance on the strong beat, the rules from 2nd species are used. Since 2nd and 4th species both are about putting two half notes to each whole note of the cantus firmus, they are naturally connected.

4 Experimental results

The set of rules I implemented concerning 1st species is quite complete. First species counterpoint is not a hard problem, so the computer did not take long to find a solution. And for each new rule added, there was no problem in fulfilling it.

All errors discovered was punished with the value 1.0, so it was easy to count the errors. The soft rules like the countermovement rule was on the other hand given 0.1 in punishment, everytime there was a non-countermovement. Figure 1 shows an example of an evolved counterpoint with fitness 0.6 in this setting. All harmonic and melodic rules have been kept, but there are 6 non-countermovements. I have put the sounding intervals between the notes.

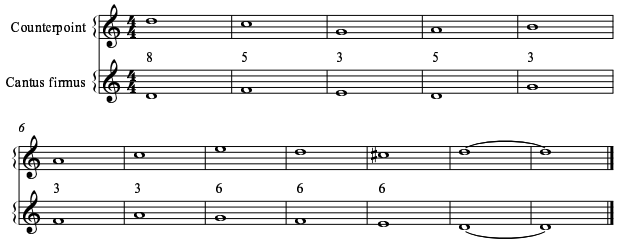

Figure 1: First species counterpoint

With less restriction on the cadence (see below), the algorithm found a counterpoint with only 2 non-countermovements.

The next experiment was about making music in half notes: the 2nd and 4th species. Now there is the possibility of making dissonances, if they are treated properly. So I run the algorithm, but only consonant intervals had been found which is just as correct, but a little boring, and certainly not the point of doing the species. It was of course easier for the algorithm to fulfil the simplest rules (make consonances everywhere) in stead of making dissonances which depends on the notes before and after it. So I had to trick the algorithm into making the fun stuff. I introduced a small reward in the evaluation, everytime the algorithm found a correct treated dissonance. I spent some time adjusting the size of the reward. It seemed that it should be smaller than the standard error punishment, so a correct dissonance can not compensate for some other broken fundamental rules (then it is better to keep it correct).

A similar problem is the making of the cadence. We like to have the leading tone introduced in the next to last note (the # altered note in both figures). It also requires special treatment in introducing, so again a reward was necessairy.

I ended with the reward value -0.1 for every correct treated dissonance and leading tone. With this setup the program was able to produce the counterpoint shown in figure 2:

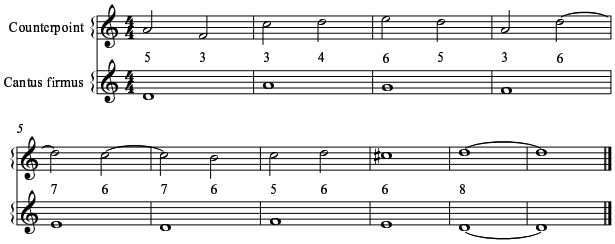

Figure 2: Second and fourth species counterpoint

There are features from both 2nd and 4th species. Remember that the intervals: 2, 4 and 7 are dissonant. 5 and 8 are true consonances and 3, 6 and 10 are just consonances. The fourth in measure 2 is introduced stepwise on the week beat and between 2 consonances (2nd species). The sevenths in measure 5 and

6 are prepared from a consomance in the preceding measure and are resolved to a non-true consonance (4th species).

It is starting to get a little more difficult to fulfil all the restrictions. Jeppesen points out that the melodic rule of the order of large and small steps loses it's importance when dealing with 4th species, since you locallly hear the syncopated dissonance a lot more than the melodic line. The given example however fulfills all the introduced rules. And still the running time is within a minute. For the experiments I used a population of 1500 for 60 epochs.

I must admit that I have only tried the algorithm on three different cantus firmus and in one mode (d-dorian), but I very much expect the algorithm to behave similarly on other melodies.

In my implementation I still need to take care about some special events, which are allowed in the music. But these are mainly minor changes or updates to the program. One of them is concerning the treatment of upbeat. More modes (than dorian) can easily be implemented. But more fundamental: I still miss something to take care of the tritone (augmented fourth / diminished fifth – most dissonant and therefore forbidden) interval which for the moment do not recieve special treatment (which it should). I must read the chapter on flat alterations to avoid the tritone interval.

4.1 The musical quality of the counterpoints

The music evolved is clearly acceptable – it adheres to the rules. I still need some melodic rules to implement. I can get an advantage from implementing rules dealing with sequences (repeats of musical fragments) and consequtive jumps in triads. Both are considered to deteriorate the counterpoint.

The melodic lines are sometimes a bit without direction, which they should indeed not be. A couple of preference rules could be added, to deal with that kind of problems.

5 Perspectives and future ideas

The species rules are, as we have seen, building blocks for evaluation of music with mixed rhythm values (like real melodies). So with some rules for quarter notes and for combinations of notes with different rhythm values it is possible to make 5th species counterpoint.

In 5th species it is necessary to introduce rhythmic operators for making mutations in the counterpoint. So far I have only used pitch mutation since the rhythm was fixed. They should not be totally random, but rather suggest som standard rhythmic licks from the style! For example mutations for making one whole note into one dotted half note and a quarter note. Pitch mutation should continue as before.

The next step is then to make free two part music. By then, it is possible to make different rhythm values in bots parts. We do not have the cantus firmus here, but still our species rules can be used – one of the voices is simply locally considered as the cantus firmus.[1]

My idea is to make a unit in the program, which looks at one measure at a time, and finds out which set of rules should be used for the evaluation, and then evaluates it.

The search might be somewhat harder when none of the voices are fixed, and both of them can be altered or mutated. The search space is then growing consederably, but then again there are a lot more correct solutions.

The next step in Jeppesen is 2 part imitation-style. We need to introduce a notion of imitation. In the simple style the imitations should be the excact reproduction of the theme, but later one can experiment with variation / mutation of the theme.

And then three part music and four etc. There are also rules for how to put lyrics on the music.

Another experiment which could be nice to do, is to evolve a cantus firmus. So far I use the ones in Jeppesens book, but with the melodic rules and a little more, it should be possible to evolve a wellformed melody which could be

[1] So far I have not talked about what happens when the counterpoint is below the cantus firmus, but the rules are exactly the same: same dissonance treatment, melodic lines etc.

used as cantus firmus in a species-piece.

All through the experiment I have been very discrete in the calculation of the evaluation. EA's usually perform very well on a more fuzzy evaluation function, but I haven't found any reasonably (musical) way of grading music which is almost correct (more than already described by the punishment and reward approach). Since some of the rules are "either-or" I don't think it is approproate to make those fuzzy, but I could experiment more with grading the more soft rules, when the algorithm meets larger challenges and not all wishes can be met.

6 Conclusions

This project shows how to use an EA to search for counterpoints. Farbood [1] describes a very slow, but quite complete system for making 5th species counterpoint for up to 6 voices made by Schottstaedt [4]. It seems that an EA would be able to perform a bit faster than the recursive method of that system.

I haven't had a second persons opinion of the quality of the counterpoints. The quality of the correct counterpoints do vary, but I think that the two examples brougt here are quite good. Some rules yet need to be implemented, so I have hopes. The posibility of extending the system are quite clear to me. The real challenge is to adjust the musical parameters with the grading of the fitness function.

References

[1] Mary Farbood Bernd Schoner. Analysis and synthesis of palestrina-style counterpoint using markov chains. *Proceedings of International Computer Music Conference. Havana, Cuba*, 2001.

[2] Knud Jeppesen. *Kontrapunkt, Vokalpolyfoni*. Wilhelm Hansen, Copenhagen, 4th ed edition, 1968.

[3] Thiemo Krink and Rasmus K. Ursem. *Introduction to Evolutionary Computation*, chapter 1. (in prep.), 2003?

[4] William Schottstaedt. Automatic counterpoint. *In: Current directions in Computer Music Research*, 1989.