NTNU

Norwegian University of
Science and Technology

An Application of Evolutionary Algorithms to Music:

Co-Evolving Melodies and Harmonization

Olav Andreas E Olseng

Master of Science in Computer Science
Submission date: June 2016
Supervisor: Bjørn Gamback, IDI
Co-supervisor: Lars Bungum, IDI

Norwegian University of Science and Technology
Department of Computer and Information Science

Olav Olseng

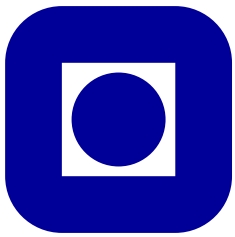
An Application of Evolutionary Algorithms to Music:

Co-Evolving Melodies and Harmonization

Master of Science in Computer Science, May 2016

Supervised by Björn Gambäck
Co-supervised by PhD Candidate Lars Bungum

Artificial Intelligence Group
Department of Computer and Information Science
Faculty of Information Technology, Mathematics and Electrical Engineering



Abstract

Algorithmic composition is a field that is close to 60 years old, and has seen much research. Systems today are able to do a wide range of compositional tasks, ranging from simple melody generation to fully automated orchestral composition. Systems for computer aided composition are becoming more and more common, either to evaluate music created by humans, or as generators of raw material to be used by composers.

This Master's Thesis describes a novel implementation of a multi-objective evolutionary algorithm, that is capable of generating short musical ideas consisting of a melody and abstract harmonization. The implementation is capable of creating these ideas based on provided material, or autonomously. Three automated fitness features were adapted to the model to evaluate the generated music during evolution, and a fourth was developed for ensuring harmonic progression. Four rhythmical pattern matching features were also developed.

The implementation produced 21 pieces of music, under various configurations, that were evaluated in a study. The results of this study indicates that the system is capable of composing ideas that are subjectively interesting and pleasant, but not consistently.

Sammendrag

Algoritmisk komposisjon er et felt som er nærmere 60 år gammelt, og har nytt mye forskning. Dagens systemer er i stand til å utføre forskjellige oppgaver innenfor komposisjon, fra å generere enkle melodier, til fullskala komposisjoner for orkester. Systemer for datastøttet komposisjon blir mer og mer vanlig, enten til å analyse og evaluering av komposisjoner laget av mennesker, eller som generatorer av råmateriale for komponister.

Denne masteroppgaven beskriver en ny implementasjon av en multi-objektiv evolusjonær algoritme, som er i stand til å generere korte musikalske idéer. Disse idéene består av en melodi og en abstrakt harmonisering. Implementasjonen kan basere disse idéene på musikalsk materiale, eller generere dem helt på egenhånd. Tre automatiserte fitness funksjoner er tilpasset modellen, for å evaluere den genererte musikken under den evolusjonære syklen. En fjerde funksjon er utviklet for å forsikre harmonisk progresjon. Fire målbare attributter som måler repetisjonen av rytmiske mønstre er også utviklet.

Implementasjonen genererte 21 stykker, under forskjellige konfigurasjoner, som ble evaluert i en studie. Resultatene av denne studien indikerer at implementasjonen er i stand til å komponere musikk som er subjektivt behagelig, og interessant, men ikke konsekvent.

Preface

This thesis is submitted to the Norwegian University of Science and Technology (NTNU) in partial fulfilment of the requirements for the degree of Master of Science in Computer Science. The Master's Thesis work has been performed at the Department of Computer and Information Science, NTNU, Trondheim, with supervisor Björn Gambäck and co-supervisor Lars Bungum.

Olav Olseng
Trondheim, 13.06.2016

Acknowledgements

I would like to thank my supervisor Björn Gambäck, and assisting supervisor Lars Bungum, for interesting discussions on creativity, computational approaches to creativity, feedback and guidance over the course of the semester. Special thanks to Björn Gambäck for allowing me to take on this project, allowing me a vast amount of creative freedom, as well as his ability to provide interesting papers and journals on a general basis.

I would also like to extend my gratitude towards all the people who partook in the study that was conducted.

Contents

List of Figures

List of Tables

1. Introduction

Music has been a part of human culture for thousands of years, and behind every piece of music there is a composer. In orchestral context, this has been reserved for fully educated, professional composers. On the other hand, you have music that is composed for a smaller ensembles, like jazz or rock, and music that can be performed by only one person.

With the emergence of cheap computing power, processing units are readily available, but the inspiration needed to compose is not always there. Even for a person with the skillset to create compositions, the ability to constantly re-invent themselves and produce new and interesting music can be elusive. Computers are, and have been, used as a source of inspiration for composers. An example of such a computer aided compositional tool is the Vox Populi system, [Moroni et al., 2000].

1.1. Algorithmic Composition

The notion of algorithmic composition is generally accepted to have first appeared in the 1950s with the Illiac suite [Hiller and Isaacson, 1958]. The system composed this suite through four experiments, which adopted both a rule-based approach as well as Markov chains, to generate a score for a string quartet. Since then, a wide range of techniques found in the field of artificial intelligence have been applied to composition. Rule-based systems, stochastic methods, grammar-based methods, neural networks, and population-based methods have all been utilized.

Systems have been designed to perform a wide range of compositional and musical tasks, such as generating melodies, harmonization, counterpoint, improvisation, arranging, and in some cases like Melomics and Iamus [Sánchez-Quintana et al., 2013], the ability to create fully fledged compositions aimed at orchestral performances in concert halls.

1.2. Project Goals

In this section the main goals for this master thesis are presented.

1.2.1. Develop a Novel and Useful Algorithm for Composition

In systems that are able to generate both melody and harmonziation, these to tasks are most commonly done sequentially, and by different modules in hybrid systems. Evolutionary algorithms have been applied to both tasks separately, but not in tandem. Implementing an evolutionary system that has the ability to co-evolve melody and harmonization has not been done before. Is it feasible to do this in an evolutionary algorithm, and what are the benefits and detriments of this approach? Developing an algorithm that is able to generate music that can be of use to humans is goal of this thesis.

1.2.2. Develop an Automated Fitness Functions

McCormack defined a set of five open questions for research in algorithmic composition, specifically in evolutionary music [McCormack, 2005]. His second open question is formalized as such: *"To devise formalized fitness functions that are capable of measuring human aesthetic properties of phenotypes. These functions must be machine representable and practically computable."*. This problem is still an important question [Galanter, 2010]. Critique has been made towards artefacts generated by evolutionary systems, in terms of having little structure. Defining fitness features that attempt to stimulate the emergence of patterns is worth delving into, and is a goal of this thesis.

1.3. Contributions

C1 A novel implementation of an algorithm that is capable of writing music autonomously.

C2 A configuration of automated fitness objectives for evaluating both melodies and harmonization in parallel.

C3 Four statistical features for rhythmical pattern matching in melodies.

C4 A fitness objective for harmonic progression.

1.4. Thesis Structure

Relevant background theory on evolutionary and multi-objective evolutionary algorithms are described in Chapter 2, while Chapter 3 provides an overview of basic music theory and music terms needed to understand and discuss much of the work in this thesis.

An overview of the State-of-art in algorithmic composition and related work is presented in Chapter 4. Chapter 5 describes the implemented algorithm in detail, including minor modifications to the non-dominated sorting genetic algorithm II, genotype representation, genetic operators and fitness functions.

In 6, the various configurations for how the music was generated is presented. Chapter 7 describes a survey used to evaluate the generated music, and also presents the results of the survey.

Finally, Chapter 8 contains a discussion over the results, accompanied by a conclusions and suggestion for future work.

2. Evolutionary Algorithms

Evolutionary algorithms (EAs) are inspired by the process of biological evolution, and try to optimize a solution in regards to some objective or environment. The algorithm is in essence a parallel heuristic search, and performs the search by maintaining and developing several potential solutions at once. The algorithm combines and mutates these solutions in an effort to converge towards a global minimum or maximum, an optimal solution, in regards to a fitness function.

2.1. The Standard Algorithm

Evolutionary algorithms use an iterated approach, where each iteration is called a generation. Each individual, or candidate solution, in a generation is represented by a genotype. This is the equivalent of the individuals' DNA. Through a development process, the phenotype for each individual is generated from their genotype. Together all the phenotypes in a generation is called the population. A flow chart of a typical evolutionary algorithm can be found in Figure 2.1.

2.1.1. Initialization

Before the the evolutionary algorithm can start its cyclic process, an initial population must be present. Genotypes for this population can either be created at random, or be provided. Once this is done, the phenotypes are developed from the genotypes, according to the development scheme. Then, each phenotype in the population is evaluated by the *fitness function* of the algorithm and assigned a fitness value. After this is done, the cyclic process is started.

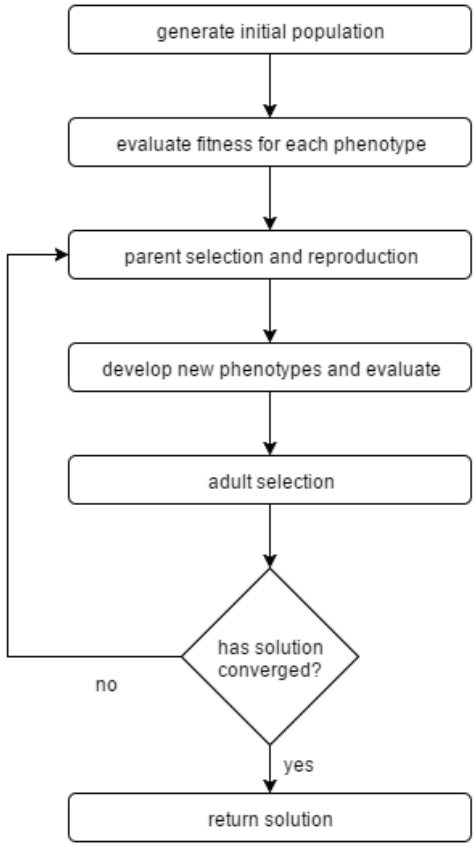


Figure 2.1.: Flow chart of an evolutionary algorithm.

2.1.2. Generational Cycle

To create a new population for the next generation, the algorithm has to select which phenotypes that are allowed to reproduce. This is done through a process called *parent* selection. There are several approaches to parent selection, but in common they will generally select individuals with a high fitness to ensure convergence towards a good solution. However, to keep the search from converging too quickly and terminate in a local minimum or maximum, the best solution is not always selected for reproduction.

To explore the search space, each new genotype is derived from individuals that are selected through parent selection. By copying the genotype of a selected individual and altering it, a new candidate solution that is similar to its parent is produced. This alteration of the genotype is called applying *genetic operators*, and there are two common ways of doing this. The first one is by *mutation*, where a single genotype is altered in a small way. For instance, a genotype consisting of a bit-string could be altered by flipping a random bit. These operators are called mutation operators.

The second class of operators are called *crossover*-operators. These operators take two genotypes, and recombine them into a new genotype. A typical way of doing this is to choose a point to split the genomes in two, and then take the first part from one parent and the second part from the other. This is called single-point crossover. After a new set of genotypes are generated and their phenotypes developed, the phenotypes are evaluated by the fitness function.

After the new phenotypes are developed and evaluated, the algorithm must select which phenotypes to populate the next generation with. This is called *adult selection*, and can be implemented in several ways. One implementation is to produce an entirely new population and discard the old one, but this approach has the weakness that it might discard all the good solutions and be left with only bad ones. To avoid this, some form of *elitism* is commonly used, which means that a fixed amount of the best solutions from the previous generation is transferred to the new one.

The final step in the cycle is to check if the solution has converged. If the problem has a well defined solution and the fitness has reached a known maximum value, the algorithm can terminate. This is often not the case, and other measures of convergence must be used. A typical measure of convergence would be if the highest fitness in the population does not change over a certain amount of generations. If this is not possible, the algorithm must be stopped manually or after a given amount of generations.

2.2. Multiple-Objective Evolutionary Algorithms

Multiple-Objective Evolutionary Algorithms (MOEAs) are a specific kind of evolutionary algorithms (EA), that differ from standard EAs in one conceptual way. In the standard EA the fitness function assigns a single numerical value to the phenotypes. In MOEAs the phenotypes are evaluated by more than one fitness function, called objectives, and thus have more than one fitness value. This enables the algorithm to work towards multiple, and sometimes conflicting, goals. The result is a trade-off between the objectives, unless some solution exists where all objectives can be maximized.

2.2.1. Non-Dominated Sorting Genetic Algorithm

The non-dominated sorting genetic algorithm (NSGA) is a MOEA implementation that uses pareto dominance and pareto optimality, from game theory, to rank the individuals in a population. An outcome of a game can be defined as pareto dominated, if there exists some other solution where at least one player would be better off, without it having any negative impact on any other player. A solution is said to be pareto optimal if it is not pareto dominated by any other solution in the solution space. When evaluating pareto dominance in MOEAs, the fitness values are considered to be the players of the game. This means that a solution is pareto dominated if there exists another solution where one fitness value is better, and the remaining features are not worse.

The algorithm described in Chapter 5 is an implementation of the non-dominated sorted genetic algorithm II (NSGA-II), proposed by [Deb et al., 2002]. It improved the run-time of the standard NSGA from $O(MN^3)$ to $O(MN^2)$, where M is the number of objectives and N is the population size. NSGA-II also supports elitism and a diversity preserving mechanism called crowding distance. These three improvements to the NSGA-II over the original NSGA were shown to have a positive impact on multi-objective optimization problems.

NSGA-II differs from the standard EA in how it compares fitness values between individuals. In the standard EA, where there is only one fitness value, the individual with the higher fitness value could be considered better. In NSGA-II, a rank is assigned each individual in a population

based on pareto optimality. The ranking is computed by finding the non-dominated individuals and removing them from the population. Some individuals that were previously dominated then become non-dominated. This is done iteratively until there are no more individuals left in the population. The rank assigned to an individual is the iteration it was removed from the population. The non-dominated solutions removed in the first iteration obtain rank 1, the solutions removed in the second iteration obtain rank 2, and so on.

When doing parent selection, tournament selection is used and the ranks are compared. The individual with the lower rank is considered the better individual. Note if two individuals with the same rank are to be compared, the comparison will be resolved by comparing another value, called the crowding distance. This value is in essence a measure of how close an individual is to its closest neighbours in the fitness landscape. The bigger the crowding distance, the more unique the solution is within a rank. By selecting individuals with a high crowding distance over a low one, genetic diversity is maintained in the population. When performing adult selection, the newly generated individuals are merged with the current population before ranking is performed. The population can then be trimmed with regards to a maximum population size. This is how elitism is supported, as individuals from previous generations will survive if they have a high enough rank to remain in the population during adult selection.

3. Basic Music Theory

Music theory has been well defined and formalized in western culture over several hundred years. For readers who are not initiated in basic music theory, this section is meant to give a minimal frame of reference for discussion around musical concepts used in algorithmic composition.

3.1. Basic Musical Elements

Note Pitch

Every note in a piece of music has a pitch, also known as tone. In western music, there are twelve different named pitches that are commonly used. The white keys on the keyboard of a piano are named in a wrapping pattern of seven notes (A, B, C, D, E, F, G) known as an octave, The black keys to the right of a white note are represented by adding a # after it, like A#. Each octave is enumerated, which means that A1 and A2 are distinct. The "reference tone" is 440Hz, which is the $A4$ pitch. All other used pitches are derived from this pitch. A pause is an absence of pitch.

Musical Key

A musical key denotes a collection of pitches that "belong" in a musical context. In western musical culture a key contains seven pitches, and are denoted by a root pitch and a mode. There are two modes commonly used in western music, major and minor. Together with a chosen root pitch, the mode defines the musical key. In C major, the pitches are: C, D, E, F, G, A, B, C. This particular ordering of pitches is known as the C major scale. Most musical pieces only make use of one key.

Interval

An interval is the distance relation between two pitches, commonly denoted in semi-tones. The most common intervals and names can be found in Table 3.1. An octave up from C1 is C2, the fifth up from C1 is G1 and the fifth down from C1 would be F0.

Table 3.1.: Interval names and corresponding distances in semi-tones

Interval Name	**Distance**
Unison	0
Minor Second	1
Major Second	2
Minor Third	3
Major Third	4
Fourth	5
Augmented Fourth/Diminished Fifth	6
Fifth	7
Minor Sixth	8
Major Sixth	9
Minor Seventh	10
Major Seventh	11
Octave	12

Step Interval

And interval between two consecutive notes in a melody that is a general second. That means the two pitches in a step interval are either one or two semitones apart.

Chord

A chord is a combination of at least three pitches (a triad) played together at the same time, where one of them is denoted as the chord root. The two others are the ones that are two and four steps above the chord root in the scale, also known as the third and the fifth of the chord. The two most common chords are major and minor chords.

Dissonance
A relative graded measure of how "tense" or unpleasant an interval or a chord sounds to the listener. The opposite of dissonance is consonance.

Note Duration
A note needs both pitch and a duration to have purpose in music. The duration of a note is represented as a fraction, the most common being: $\frac{1}{1}$, $\frac{1}{2}$, $\frac{1}{4}$, $\frac{1}{8}$ and $\frac{1}{16}$. Pauses also have duration.

Time Signature
A compartmentalization of notes, again represented as a fraction. The most common one is $\frac{4}{4}$, but others can be observed as well. For instance, all waltzes use a $\frac{3}{4}$ time signature.

Measure
A measure, or bar, of music is one pass of the chosen time signature. A piece of music usually consists of many sequential measure. Each fraction of a measure is known as a beat.

Tempo
Refers to the time between beats in a measure while the music is being performed, in beats-per-minute (bpm).

Figure
A tiny musical idea that can be repeated. It may contain any combination of melodic pitch, harmony and rhythm.

3.2. Melodies and Harmonization

One of the more familiar terms from music is a melody, or a lead. Usually this is performed by a singer or a monophonic instrument. The melody is often what makes a song recognizable to the listener, and is often paired with a text if it is performed by a singer. Melodies can be split into musical phrases, which consist of an arbitrary amount of notes over an arbitrary amount of bars.

Harmonization is the act of putting together two or more pitches at the same time. A harmonization can be built around a melody, or a melody

around a harmonization. In modern music this is commonly represented as a chord progression. Through different paradigms in music history, there have been several rules and norms for what chords should follow what chords, and how a beautiful melody should sound. These rules were based on the aesthetic of the time, and have changed throughout both history and cultures.

4. Related Work

This chapter is intended to give an overview of the State-of-the-Art in algorithmic composition, with extra attention to applications of evolutionary algorithms.

4.1. Literary Search Method

There has already been conducted literary surveys in the field of algorithmic composition by Papadopoulus and Wiggins [1999] and Fernández and Vico [2013]. The latter survey references over 280 works. These two papers are also the top results returned from Google Scholar with the following input:

```
("computer composition" || "algorithmic composition" )
& ("artificial intelligence" || AI)
```

At the time of writing the query returned a total of 1650 papers. The two surveys were used as a foundation for literary research, as they give a comprehensive overview of the field's history at their respective time of writing. Filtering the query by release year, looking up specific authors with referenced works in the surveys and citations, enabled the author to find relevant papers published after the submission of the Fernández and Vico paper.

To obtain other papers concerning the use of evolutionary and multi-objective evolutionary algorithms in algorithmic composition not mentioned in the surveys, one of the following terms was concatenated to previously mentioned query:

```
& (evolutionary)
or
& (multi objective)
```

4.2. State-of-the-Art in Algorithmic Composition

After the publication of the Illiac Suite[Hiller and Isaacson, 1958], much research has been made in the field of algorithmic composition. The predominant technologies being used are constraint logic programming, Markov chains, artificial neural networks and evolutionary algorithms. The material here does by no means cover the field in its entirety. It is intended to give the reader an overview of the more popular applications of various methods used in the field.

4.2.1. Grammars

The first systems produced, including that of the Illiac Suite, utilized symbolic methods. Grammars have been a popular method in algorithmic composition. Stochastic grammars, as they can be translated into Markov chains to perform generative tasks, have been frequently used. The first examples of grammars to appear were defined by human human hands, such as Rader [1974]. Systems have also been developed that extract grammars automatically from a corpus, and then use these grammars to generate similar compositions to the ones in the corpus. An example of this is the work by Cruz-Alcázar and Vidal-Ruiz [1998]. Application of L-systems is also a popular approach, and an example of this is [McCormack, 1996], where he generates melodies.

4.2.2. Constraint Satisfaction Solvers

Knowledge and rule-based systems have been applied frequently in compositional tasks, and frequently with success to classical styles of music and developmental tasks. CHORAL [Ebcioğlu, 1988] is an example of such a system. It uses constraint language programming to solve four-part choral harmonization, in the style of Bach. The system employs around 350 constraints to guide the musical composition, as well as heuristic guidance to generate notes before testing, and backtracking in the case that the constraints fail to be met. The system uses various "views" to evaluate constraints, most notably vertical ones to evaluate harmony, and horizontal ones to evaluate progression of the voices. Ebcioğlu found problems with his system, as the rules employed were incomplete, or inconsistent with

certain passages in Bach's work. This is a typical shortcoming of expert systems. They fail to capture the full creative capabilities of acclaimed composers, and similar issues were already found by Rothgeb [1969] two decades earlier.

4.2.3. Markov Chains

Markov chains are a very popular method in algorithmic composition, and can be traced back to the Illiac suite. It was very popular in the early days of algorithmic composition, despite the fact that some of their limitations were discovered quite quickly. Moorer [1972] wrote how higher order Markov chains would just reproduce excerpts of the corpus it was trained on, and the lower order chains failed to capture global structures, such as figuration.

Markov chains are still frequently used, but in more elaborate ways. A recent example is the SMUG system [Scirea et al., 2015], which generates melodies using two Markov chains, one for pitch and one for rhythm. The chains in this system are trained through the use of an evolutionary algorithm. Markov chains are most often used in conjunction with some of the other techniques. Another example of this is the system proposed by Gillick et al. [2009], which constructs Markov chains from extracted grammars.

4.2.4. Artificial Neural Networks

Artificial Neural Networks (ANNs), within algorithmic composition, started to appear around the beginning of the 1990s. Recurrent ANNs recieved a lot of attention, as they have temporal capabilities that are useful when it comes to music. One example of this is Todd [1989]. The system encoded melodies by absolute pitch, and presented training data to a three-layer network sequentially to train it. When provided with other melodic material as input after training, the network would generate a new sequence. Interpolation was achieved if several pieces were supplied, and extrapolation when a single piece was provided.

Efforts have been made to enable the ANNs to capture global structures in addition to local ones. Eck and Schmidhuber [2002] attacked this problem in melodic generation by using long-short term memory networks, to

some success. Boulanger-Lewandowski et al. [2012] attempted the same using a 67 hour corpus of polyphonic music pieces, but failed to capture the global structures. The representation used allows for all 88 notes on the piano to be played, and a maximum of 15 notes played at the same time. The consequence is in an enormous state-space given the temporal dimension.

ANNs are also applied frequently in hybrid systems, and some of which will be mentioned in section 4.2.6.

4.2.5. Evolutionary Methods

The use of evolutionary algorithms (EAs) within algorithmic composition, and computer aided composition, first appeared in the early 90s. Some of them were fully automated, such as Horner and Goldberg [1991] and Ricanek et al. [1993]. Both these were solving simple thematic bridging problems, in essence formulated as a puzzle, or toy problem, where the fitness function was a simple distance measure to a pre-defined solution.

In the following years, fully automated systems were developed for various compositional tasks with a variety of automated fitness functions. Marques et al. [2000] composed polyphonic pieces, Papadopoulus and Wiggins [1998] did jazz improvisation, and Johnson et al. [2004] composed melodies. All these implementations used a fitness function that was a weighted sum of features.

Towsey et al. [2001] published a paper that exclusively provided a description of a a set of musical features used to analyse musical pieces of various styles, but no generative implementation was presented. Fully automated fitness functions have been a point of much research and discussion [de Freitas et al., 2012][McCormack, 2005][Galanter, 2010]. It has even been suggested using no fitness function at all [Biles, 2001]. Freitas and Guimarães [2011b] suggested this as a means to circumvent the artistic constraints that fitness functions impose. As a consequence, much of the domain knowledge previously found in the fitness function is rather moved into the genetic operators, the representation, and the initial population to ensure that the produced material is acceptable.

Another proposed way of introducing originality and diversity to the systems produced artefacts is using multi-objective fitness functions. This has been a point of research in the recent years, and has been applied to

harmonization [Freitas and Guimarães, 2011a]De Prisco et al. [2010], and melodic composition [Jeong and Ahn, 2015]. All of the three aforementioned papers use two contradicting functions, one for consonance and one for dissonance. This results in a musical trade-off between the two functions. Compositions made by these systems are said to have a broader language of expression, producing subjectively interesting compositions. These three papers are also the most relevant to the described design in Chapter 5.

Another group of EAs are commonly referred to as "Musical Interactive Genetic Algorithms" (MIGAs). In these cases, the fitness function is replaced by a human evaluator. Generally, a handful of solutions are presented to the reviewer, and are ranked in some fashion. The next generation of individuals are then produced, based on the human evaluation. The first occurrences of MIGAs typically also worked on toy problems. Nelson [1994] described a very simple MIGA for generating rhythmic patterns over short melodies, where the generated music simply contained the presence or absence of sound.

The most notable MIGA at the time of writing is the GenJam system [Biles, 1994]. It improvises jazz solos. During the interactive fitness evaluation of the composed music, the user simply responds with a good or bad evaluation, and the system learns from this. A common problem for interactive genetic algorithms is user fatigue. Evaluating the candidate solutions for each generation can be extremely time consuming. Biles tried to eliminate the human evaluator in his system on two occasions. In 1996 he attempts to replace the human element with a trained neural network, to no success [Biles et al., 1996]. In 2001, however, he successfully removes the fitness function in its entirety [Biles, 2001]. The musical acceptability was ensured by programming the knowledge previously found in the fitness function into the genetic operators, as well as presenting a good initial population. This version of the system is said to outperform the original.

On the cutting edge of evolutionary music, focus has been directed at the encoding between genotype and phenotype. Given a complex search space, having similar musical figures appear in the artefacts is unlikely to happen by random mutation. The main contribution by the Melomics system [Sánchez-Quintana et al., 2013] is something coined an "evo-devo"

approach. In essence the systems genotype has an indirect encoding, that during development phase expands the genome into a more advanced piece of music than it could do with direct encoding. Thus it is able to deliberately develop themes, global structure and alter motifs.

4.2.6. Hybrid Systems

Due to the inherent strengths and weaknesses of different approaches, all of the techniques previously mentioned in this section (4.2) have at some point been hybridized. Two such systems are the HARMONET [Hild et al., 1992] and its further development, MELONET [Feulner and Hörnel, 1994]. HARMONET has three layers to its architecture, which enables it to compose four-part chorales. The first layer being a neural network that extracts harmonic information from a melody. This information is then fed to a rule-based layer, that creates a chord progression for the melody. The third layer arranges a four-part harmonization for the melody. MELONET adds a final layer to this system, another neural network, that creates melodic variations for the voices, in an attempt to make the produced artefact more interesting to the listener.

Evolutionary algorithms lend themselves very well to hybridization. Both the phenotype and the fitness function can be changed from the traditional approaches mentioned in Section 4.2.5. The system introduced by Bell [2011] uses an interactive evolutionary algorithm to evolve Markov chains that write melodies with accompanying text. Each genome is developed into three first-order chains, one for melody, one for rhythm and one for harmonization. Thywissen [1999] designed an algorithm where generative grammars are evolved, and Khalifa et al. [2007] used grammars as a part of the fitness function. Phon-Amnuaisuk et al. [1999] programmed a genetic algorithm for harmonization, where some of the genetic operators follow explicit rules, instead of random bitwise mutations. NEUROGEN [Gibson and Byrne, 1991] used ANNs trained on a corpus as a fitness function, in an attempt to emulate the musical style found in the training data, while Chen and Miikkulainen [2001] evolved recurrent neural networks that did the actual generation of the melodies.

A system called MetaCompose was recently developed for generating melodies and harmonization [Scirea et al., 2016]. The system composes autonomously and does not base the compositions on any musical material.

It generates chord progressions by doing a random walk through a directed graph, then creates a melody for the chords through a multi-objective evolutionary algorithm, supported by some a probability driven module for rhythms and and accompaniment.

5. NSGA-II Algorithm Implementation

The problem domain specific components of the algorithm are described in this chapter, including the genetic model, operators and fitness functions. Minor modifications to the NSGA-II algorithm are also described.

5.1. Non-problem Specific Configuration

During preliminary tests of the algorithm, the performance of the ranking scheme collapsed when four objectives were used. 100% of the population would be ranked to the first front, meaning that they were all pareto optimal, but still far from the true pareto front. When this happens, there is an extreme selection pressure, and exploration of the search is hampered. Deb [2014] describes this as a phenomenon that occurrs when there are many fitness functions. This problem also occurred even if genetically identical phenotypes were removed from the population before ranking, as many of them evaluated to the same fitness value, despite being genetically very different. This implementation therefore employs a scheme that eliminates fitness duplicates to lessen the genetic drift, as described by Aguirre and Tanaka [2005], to improve the performance when there are epistatic interactions in genomes. Since notes in the melody are evaluated in relation to consecutive and preceding notes, as well as the harmonization, employing this scheme should improve the convergence of the algorithm by limiting the genetic drift.

For parent selection, a binary tournament selection with an 80% chance to keep the highest ranked individual was adopted.

5.2. Genotypes and Phenotype

The genotype will consist of two separate parts. One part will be for the melody, the other part for the harmonization. Representing the music this way will allow the algorithm to optimize one of the two separately during run-time, or both simultaneously.

5.2.1. Genotype for Melody

A flexible representation is implemented to allow the exploration of a large melodic space. This allows the algorithm to represent a variety of musical expressions, and to develop many different musical ideas as possible. Some implementations, such as the GenJam system [Biles, 1994], use a database of melodic figures to map the resulting genotype into a melody. Such a database is not part of this architecture. Efforts were made to put as little musical knowledge as possible into the representation, by allowing occurrences of what can be considered as "errors" in certain musical paradigms.

The model used in Jeong and Ahn [2015] is a very flexible one, and is used as a baseline for the implemented model. A measure of music is divided into fractions, and each fraction represents a time-step, or a beat. Each step is occupied by an integer value that represents one of three things: either a pitch, a rest, or a hold. A rest means the absence of anything played, and a pitch value means that a new note of the corresponding pitch starts, analogous to striking a key on the piano. The hold value means that the previous note, either a pause or a pitch, is held from the previous step. The minimal time-step used in this model is a $\frac{1}{16}$ note, which results in twice as many time-steps per measure compared to the representation used by Jeong and Ahn [2015]. The representation allows the algorithm to generate notes of $\frac{1}{16}$ length and longer. It also supports repeating pitches, and leaps of any size, within the defined range of allowed pitches. The values are stored in bytes, in consideration to performance.

Table 5.1 describes the mapping between valid byte values and note values in the melody genome. The range of pitches employed correspond to the integer-pitch mapping defined in MIDI standard [MIDI Manufacturers Association et al., 1996]. As a result, the representable pitches are in the

Table 5.1.: Table describing the relation between the values of a byte and corresponding musical interpretation in the melody genome.

Integer value	**Musical interpretation**
0	pause
1	hold
65-85	pitch values

range from F4 to C#6.

The total number of different representable melody genotypes n, of x measures of length, is therefore defined as:

$$n = 22^{16x}$$

5.2.2. Genotype for Harmony

The harmonization is defined by chords, where a new chord appears at the first beat of every measure. Each chord is represented as a vector of four bytes, where the bytes hold a value in the range of [-1-11]. The integers from and including 0 represents a pitch, and -1 represents an absence of a pitch. Each chord contains at least three notes, where the fourth optional note is a flavor note that allows chords to be built with a more musically advanced expression, compared to what is possible with only three pitches. An example of the genotype can be found in Figure 5.1.

Figure 5.1.: Example harmony gene, in this case a C major triad chord.

0	4	7	-1

In this representation, a 0 translates into a musical pitch of C, and following a chromatic scale, a 2 will represent an D, and so on.

The total number of different representable harmony genotypes n, of x measures of length, is therefore defined as:

$$n = (12 \cdot 11^3)^x$$

5.2.3. Phenotype and Developmental Method

Both genotypes employ a direct representation, and hence no explicit developmental method or phenotype is required. However, for optimization reasons a phenotype representation is employed to aid in the evaluation of the generated music. The developmental method iterates through the genotype and creates a variety of structs. These structs contain information about pitch and rest positions, durations and intervals, and mostly serves as indices utilized by the fitness functions.

5.3. Genetic Operators

As the system will have two different data structures in the genotype (one each for melody and harmony), two different sets of mutation operators are needed.

5.3.1. Crossover Operators

There are two different crossover operators that are employed by the algorithm. In both operators both the harmonization and melody are crossed over together, since they are heavily interdependent. Crossover is applied during offspring creation by a 50% chance.

Single Point Crossover A single point crossover that splices two genomes together. The splicing points are selected at random, and are restricted to being at the start of a measure. The same splicing point is used for both genotypes.

Single Measure Crossover This is in essence a dual point crossover operator, with restrictions applied to the splicing points, which are always placed at the start and end of the same measure. It copies a measure from one genome to the other.

5.3.2. Melodic Mutation Operators

An overview of the melodic operators can be found in Table 5.2. All the operators take the following restriction into account: A pause cannot

Table 5.2.: Table describing the melodic operators and their functions.

Nr.	Operator name	Operator function
M1	Note Mode	Changes the mode of a pitch or rest.
M2	Random Pitch	Selects a random pitch value for a time-step.
M3	Pitch Modulation	Modulates the pitch by [-4, 4] semitones.
M4	Note Position	Swaps a note by [-16, 16] time-steps.
M5	Duplication	Duplicates half a measure in the melody.

follow a pause or a held pause. This ensures the effectiveness of the operators, as these particular mutations will not be reflected in the phenotype, as this is an inaudible mutation. There are also cases where a melodic operator is not applicable to the genotype. The algorithm ensures that any operator that is chosen is applicable. One, and only one, melodic operator is always applied during offspring creation.

Note Mode This operator generates a new value at a randomly selected time-step in the melody genome. A chosen gene can not retain its current mode. For instance, a pitch can never remain a pitch, or a rest remain a rest. If a new pitch is to be created, the pitch value will be copied from the previous note or, if not applicable, chosen at random.

Random Pitch This operator generates a randomly generated pitch value at a random position in the genotype. If any value is found at the chosen position, it is overwritten.

Pitch Modulation Requires the presence of at least one pitch in the genotype to be applicable. Changes a randomly selected pitch by x, where $x \in$ [-4, 4] and $x \neq 0$, semitones. If a modulation will put a pitch outside of the valid defined range, it will perform the modulation in the other direction, by changing the sign of the modulation.

Note Position Selects a random pitch or rest in the genotype, and moves it by a random amount of time-steps x, where $x \in$ [-16, 16] and $x \neq 0$. Any value at the destination gene is swapped into the position of the initial gene.

Duplication This operator that causes biggest change in the genome. It selects a random half measure, with a starting bound restricted to the beginning or middle of a measure, and overwrites another half measure in the genome with the same values. This operator is designed for repetition of patterns to occur by intention, and not by chance.

5.3.3. Harmonic Mutation Operators

An overview of the harmonic operators can be found in Table 5.3.

Table 5.3.: Table of the harmonic operators.

Nr.	Operator Name	Operator Function
H1	Chord Change	Changes the entire chord.
H2	Chord Pitch	Mutates a single pitch in the chord.
H3	Chord Swap	Swaps two chords in the genotype.

Chord Change Changes the entire chord with regards to the musical key provided to the algorithm. It will select a random root pitch for the chord, that is within the scale of the provided key, and derive a triad that is native to the scale of the key. The operator might also add a flavor note, with a 50% chance, which will always be the appropriate seventh in regards to the root pitch and the scale.

Chord Pitch Normally modulates a random pitch in a random chord by a semitone up or down. In the case that the fourth pitch of the selected chord is chosen, different rules apply. If the fourth pitch is absent, a random pitch is inserted. If a pitch is present, it has a 50% chance to be set to the silent value, otherwise it will be modulated like a normal pitch.

Chord Swap Swaps the position of two randomly selected chords in the genotype.

5.4. Fitness Objectives

The implementation uses four different fitness objectives, described in the following subsections, to evaluate the phenotypes. Each objective has a different purpose and evaluates a different aspect of the generated music.

5.4.1. O1: Melodic Local Objective

This objective is based on one of the fitness functions described in Wu et al. [2014]. It is concerned with the tonality of the melody, the relation pitches to the given key and the interrelation of pitches within a measure. The objective is designed with the intention that melodies generated by the algorithm sound harmonic and pleasant.

In this objective, a pitch can defined as one of the three following classes:

1. A pitch that occurs within the chord of the given measure is regarded as a **chord pitch**. This is regardless of whether the pitch occurs in the provided key or not. This is a non-harmonic pitch.

2. A pitch that occurs within the given scale, or key, is regarded as a **scale pitch**. This is also a non-harmonic pitch.

3. Any other pitch is a **non-scale pitch**.

Further, a pitch that is not classified as a chord pitch is categorized if certain conditions are met. A passing tone is a pitch that is initiated by a step from a chord tone, and resolved by a step in the same direction to another chord tone. A neighbour tone is a pitch that is initiated by a step from a chord tone, and resolved by a step to the same chord tone. The sum of passing tones and neighbour tones are labeled ornament tones.

The objective scores each phenotype based of the eight rules found in Table 5.4. The first six rules give a positive score of 1, if the condition is met, while the last three deduct points. The maximum score obtainable by this objective per measure of music is then 6. In rule seven, n denotes the amount of unresolved non-scale pitches within the measure. The 8th rule is defined as:

$$-\sum_i f(x_i) \quad \text{where} \quad f(x_i) = \begin{cases} x-7 & x > 7 \\ 0 & \text{otherwise} \end{cases} \tag{5.1}$$

Table 5.4.: Table describing the rules per, measure of music, in the Melody Voice Objective.

#	**Condition**	**Fitness value**
1	Non-harmonic Pitches < Chord Pitches	+1
2	Passing Tones ≤ Scale Pitches	+1
3	Neighbour Tones ≤ Scale Pitches	+1
4	Non-scale Pitches ≤ Ornament Pitches	+1
5	Ornament Pitches ≤ Scale Pitches	+1
6	First Pitch is a Chord Pitch	+1
7	Unresolved Non-scale Pitches	$-n$
8	Interval greater than a fifth	eq. (5.1)
9	Augmented Ninth Interval	eq. (5.2)

Where i is an interval in the measure, and x_i is the semitone distance of interval i. The 9th rule is defined as:

$$-\sum_i g(x_i) \quad \text{where} \quad g(x_i) = \begin{cases} x & x = 13 \\ 0 & \text{otherwise} \end{cases} \tag{5.2}$$

Where i is an interval in the measure, and x_i is is the semitone distance of interval i.

In the special case that no pitches are present in a measure, a score of 0 is assigned. The total score of a phenotype is the sum of the scores assigned to each measure.

5.4.2. O2: Melodic Global Objective

The melodic style objective is based on the high-level melodic features found in Towsey et al. [2001], that were developed for compositional and analytical purposes of monophonic melodies. It analyses an entire melody, and gives a more global evaluation of features in the melody, compared to the objective described in Section 5.4.1. The function uses 18 different statistical features, as normalized values, described at the end of this section. Features 1 through 7 and 9 through 13 are taken directly from Towsey et al. [2001]. All of the features from the paper were initially implemented. The pattern features, except pitch and timing repetitions,

were discarded through qualitative evaluation. The same features were also excluded by Towsey et al. in their paper, as these particular features had very large standard deviations. The features that were developed to replace them are features 15-18 through, and are all based on pattern matching. Other features such as dissonant intervals, are covered in the melodic local objective described in section 5.4.1.

Each feature is scored by equation (5.3), where x is the feature value of the phenotype, and y is a target value provided to the algorithm. The final score assigned to the phenotype by this fitness objective is the sum of all the feature scores, resulting in a maximum score of 18.

$$f(x, y) = 1 - |x - y| \quad \text{where} \quad x,\ y\ \in [0,\ 1] \tag{5.3}$$

Feature Descriptions

1. **Pitch Variety** The ratio of distinct pitches to notes. A higher value means a greater variety of notes.

$$\frac{\#\text{distinct pitches}}{\#\text{notes}}$$

2. **Pitch Range** The semitone difference between the highest and the lowest pitch in the melody, divided by the maximum range possible in the model. In this implementation the maximum range is 20.

$$\frac{\text{maximum pitch - minimum pitch}}{20}$$

3. **Step Movement** The number of intervals that are of step distance, divided the total number of intervals.

$$\frac{\#\text{step intervals}}{\#\text{intervals}}$$

4. **Non-Scale Pitch Quanta** The number of notes outside the scale, divided by the total number of notes.

$$\frac{\#\text{non-scale notes}}{\#\text{notes}}$$

5. **Contour Stability** The proportion of consecutive intervals that follow in the same direction. Two consecutive intervals incorporate three pitches and if all the three notes are of the same pitch, it is counted as moving in the same direction. Rests are ignored in this feature.

$$\frac{\text{consecutive intervals in the same direction}}{\text{\#intervals - 1}}$$

6. **Contour Direction** The sum, measured in semitones, of all rising intervals divided by the (absolute) sum of all intervals. A melody starting and ending on the same note will get a score of 0.5. Rests are ignored in this feature.

$$\frac{\text{sum of rising intervals}}{\text{sum of all intervals}}$$

7. **Pitch Frequency** The ratio of notes with pitches, to total time-steps. This feature indicates how busy the melody is.

$$\frac{\text{\#notes that are pitches}}{\text{\#time-steps}}$$

8. **Rest Frequency** The ratio of notes that are rests, to total time-steps.

$$\frac{\text{\#notes that are rest}}{\text{\#time-steps}}$$

9. **Rest Density** The proportion of silent time-steps. This measures the degree of silence in the melody. Together with feature 8, this feature indicates how silence is distributed in the melody, i.e., few and long pauses or many short pauses.

$$\frac{\text{\#silent time-steps}}{\text{\#time-steps}}$$

10. **Rhythmic Variety** This feature measures the the degree of 16 different note durations (16_{th} to whole note) used.

$$\frac{\text{\#distinct note durations used}}{16}$$

11. **Syncopation** The proportion of syncopated notes. A syncopated note is a note with duration $\geq$ one beat, and that starts off the beat defined by the time-signature.

$$\frac{\#\text{syncopated notes}}{\#\text{notes}}$$

12. **Repeated Pitches** The ratio of intervals of 0 semitones, to the total number of intervals. It only considers consecutive pitches.

$$\frac{\#\text{repeated pitches}}{\#\text{intervals}}$$

13. **Repeated Timings** The proportion of consecutive pitches with the same duration. It only considers consecutive pitches, meaning rests are ignored.

$$\frac{\#\text{repeated timings}}{\#\text{intervals}}$$

14. **On-Beat Pitch Coverage** The ratio of beats that contain pitch notes. This feature is a measure of how rhythmically the melody is centered around the time signature.

$$\frac{\#\text{beats covered by a pitch}}{\#\text{beats}}$$

15. **Distinct Whole Measure Patterns** This feature measures the ratio of distinct whole measure rhythmical patterns. A value of 1 means that every measure has a unique rhythmical pattern. Rests are ignored for this feature.

$$\frac{\#\text{distinct rhythmic patterns}}{\#\text{measures}}$$

16. **Distinct Half-Measure Patterns** This feature measure the ratio of distinct half-measure rhythmical patterns. A value of 1 means that every half-measure has a unique rhythmical pattern. Rests are ignored for this feature.

$$\frac{\#\text{distinct rhythmic patterns}}{\#\text{measures} \cdot 2}$$

17. **Positional Rhythmic Measure Repetitions** This feature gives an indication of how the repeating rhythmical pitch patterns of whole measures are positionally related to each other. A measure is counted as a positional repetition if it is repeated four measures later, or either directly or two measures after. Rests are ignored. The last four measures are only checked against the previous four, and not between themselves.

$$\frac{\#\text{positional repetitions}}{\#\text{measures - 4}}$$

18. **Positional Rest Measure Repetitions** This feature gives how the repeating rhythmical rest patterns of whole measures are positionally related to each other. A measure is counted as a positional repetition if it is repeated four measure later, or either directly after or with two measures after. Pitches are ignored. The last four measures are only checked against the previous four, and not between themselves.

$$\frac{\#\text{positional repetitions}}{\#\text{measures - 4}}$$

5.4.3. O3: Harmonization Objective

The representation for the harmonization, often referred to as chords, supports every possible chord that can be created with four pitches. This objective attempts to stabilize the chords vertically, based on the melody and the musical key within each measure, by building triads that belong to the key. It also allows the presence of a fourth pitch in the chord, but does not encourage it. Compared to the objective described in Section 5.4.4, this is the more local of the two for harmonization. It is largely based on the simplicity fitness objective presented in Freitas and Guimarães [2011a]. The difference in feature values and features in this system, is due to the absence of a corresponding dissonance function. The feature scores can be found in Table 5.5. The final score for a phenotype is the summed score for each chord, where the maximum achievable score is 0.

Condition Descriptions

Chord Root Not in Key Punishes the phenotype harshly if the chord root is not a pitch within the key.

Table 5.5.: Overview of feature scores in the Harmonization Objective.

No.	Condition	Score
1	Chord Root Not in Key	-50
2	Third Absence	-40
3	Fifth Absence	-10
4	Non-Root Unison	-5
4.1	Triad Unison	-10
5	Semitone Dissonance	-20
6	Dissonant Pitch	-10
7	Invalid Pitch	-30
8	Meaningful Seventh	+10

Third Absence This condition is met if there is no major or minor third in the chord, in relation to the chord root.

Fifth Absence Gives a negative score if a perfect fifth is not present in the chord, in relation to the chord root. A diminished fifth is not punished in the case it falls naturally within the scale, which is the case with a 2nd step root chord in a minor key or a 7th step root chord in a major key.

Non-Root Unison The presence of a unison that is not of the root pitch are punished. A triad unison is punished harder than a standard unison, as it is considered the least desirable unison.

Semitone Dissonance Checks whether any of the pitches is in a semitone distance to any other note in the chord. In the case a pitch is considered a meaningful seventh, the presence of this condition is ignored for that particular pitch.

Dissonant and Invalid Pitches A pitch is considered invalid if it is not found in the chord, or in the melody within the measure the chord is played. A pitch is considered dissonant if it does not belong in the triad chord specified by the chord root. A major third is not considered dissonant, nor invalid, if it appears within a dominant chord in a minor key.

Meaningful Seventh A seventh is considered meaningful if it is resolved by a step in the following chord. A seventh will always be considered as a dissonant pitch in the implementation, and this feature is present to offset the punishment in the case the seventh is considered meaningful. The end result is that meaningful sevenths are neither encouraged nor discouraged.

5.4.4. O4: Harmonic Progression Objective

Table 5.6.: Overview of conditions in the harmonic progression objective.

No.	Condition	Score
1	First Chord is Not Tonic	-30
2	Dominant Absent	-20
3	Unresolved Dominant	-20
4	Unresolved Diminished Chord	-10
5	Unresolved X7 Chord	-20
6	Excessive Chord Repetition	-20
7	Positional Chord Repetition	eq. (5.4)
8	Harmonic Variety	eq. (5.5)

The harmonic progression objective can be considered as a horizontal, or global, objective for the harmonies generated by the algorithm. While the harmonization objective described in Section 5.4.3 attempts to establish triads in regards to the melody within a measure, this objective rewards, or punishes chords based on their relation to chords in the other measures of the phenotype. It establishes harmonic variety by punishing excessive repetitions of any chord, as well as enforcing a global variety of distinct chord roots. Some simple rules for how certain chords should be resolved by the following chord are also implemented. An overview of the conditions in this objective can be found in Table 5.6, and a more thorough descriptions for each condition at the end of this subsection. The maximum achievable score obtainable in this objective is bound by the length of the generated phenotypes, due to the nature of the 7th feature, as described in eq. (5.4).

Condition Descriptions

First Chord is Not Tonic If the first chord in is not the tonic chord, the phenotype is punished.

Dominant Absent If a phenotype contains no chord with a 5th step root, a dominant, the phenotype is punished.

Unresolved Dominant If the chord following a dominant chord is not a tonic chord, the phenotype is punished. This function wraps, which means that a dominant chord in the last position of the phenotype will be checked for a resolution against the first chord.

Unresolved Diminished Chord This condition ensures that any naturally occurring diminished chord is resolved in one of the following ways:

1. The following chord is a D7, a dominant major chord with a minor 7th chord.
2. The following chord is one step above the diminished chord in the scale.

In a minor key the second criterion will result in a 3rd step chord. In a major key the second criterion will be the tonic. This function wraps.

Unresolved X7 Chord If a major chord with a minor 7th is not resolved by circular motion, a punishment is given. Circular motion means that the distance between two chord roots is either up a fourth, or down a fifth, as defined by the scale. This function wraps.

Excessive Chord Repetition If the same chord root appears in three consecutive positions, a punishment is dealt to the phenotype. This feature wraps.

Positional Chord Repetition If a chord is repeated in the position eight measures after, a reward is given to the phenotype. This encourages the repetition of chord progression patterns that are of eight measures in size. The condition is similar, but more relaxed, compared to feature 17 and 18 in Section 5.4.2. It rewards the emergence of

patterns through repetition. The maximum score, $f(x)$, obtainable by this condition is:

$$f(x) = \begin{cases} x - 8 & x > 8 \\ 0 & \text{otherwise} \end{cases} \tag{5.4}$$

Where x is the amount of measures in the phenotype.

Harmonic Variety This feature punishes any phenotype that does not have a predefined number of distinct chord roots. The equation is described by eq. (5.5), where x is the number of distinct chord roots in the evaluated phenotype, and y is the target value.

$$g(x, y) = |y - x| \cdot -1 \quad \text{where} \quad x,\ y\ \in [1,\ 11] \tag{5.5}$$

This entails that the maximum achievable score for this feature is 0.

5.5. Initialization and Global Constraints

In the case that one or more pieces of music are supplied to the algorithm for initialization, the initial population is derived from this material by applying the standard reproduction mechanic. In the case that no data is supplied, an arbitrary amount of random genotypes will be generated for initialization.

Since both melody and harmony are evolved simultaneously, some constraints are put on the domain. First, a diatonic key must be set. This will ensure that the harmonic domain is consistent between individuals of the population. If this is not done, crossover operators are likely to change the harmonic and melodic context to the point of reverting progress in the search, as measured by the fitness function.

6. Experiment Design

The algorithm implementation described int Chapter 5 was implemented in Java, and was run with 17 different configurations to generate a total of 21 artefacts. The artefacts would later be evaluated by a survey described in Section 7.1. The various configurations are determined by the set of target values for the Global Fitness Objective described in Section 5.4.2, the supplied genetic material for deriving the initial population, whether randomly generated individuals were used for initialization or not, and variations of genotype lengths.

Sheet music of all the generated music can be found in Appendix A.

6.1. Shared Configuration

The following configuration was common for all the 21 runs of the algorithm.

- Maximum population size $N = 1000$.
- Maximum generation count of $G = 3000$.

These parameter values were found to be fitting during preliminary test runs of the algorithm, in terms of convergence to a set of near optimal solutions. In several cases a single optimal solution was evolved that obtained the maximum possible score in all fitness objectives, a perfect score.

The transformation of the phenotypes into something that would be audibly presentable to the human evaluators was done by translating the generated melodies and chord progressions into MIDI. Using Cubase 5, a digital audio workstation, each melody and chord progression was played by a sampled piano at 120 beats per minute. No performance data was added.

The probabilities of selecting a specific genetic operator is described in Table 6.1. To reiterate what was stated in Section 5.3, there is a defined

probability, in this case 50%, that exactly one crossover operator will be applied, and there will always be exactly on mutation operator applied.

Table 6.1.: Overview of operator selection probabilities.

Crossover Probability		**50%**
	Single Measure Crossover	34%
	Single Point Crossover	66%
Mutation Probability		**100%**
Melodic Operators		
	Note Mode	15%
	Random Pitch	15%
	Pitch Modulation	15%
	Note Position	7.5%
	Duplication	7.5%
Harmonic Operators		
	Chord Change	15%
	Chord Pitch	15%
	Chord Swap	10%

6.1.1. Phenotype Selection Methodology

With each run of the algorithm, a single produced artefact was chosen for evaluation in the survey. In the case that a single individual was non-dominated, it was selected for the survey. If the algorithm did not converge on a single solution after 3000 generations, but rather a Pareto front with several rank 1 phenotypes in it, the phenotype was selected purely on achieved fitness scores. No form of curating was made by the author. The phenotype in the first front with maximized scores in objective O1, O3 and O4 was selected.

A consequence of this methodology, is that one or more features in O2 were not identical to the the target value. This is an outcome of two possible situations. Either the solution converged on a local maxima in the fitness landscape, or a target value for at least one feature in the O2 objective contradicts a rule or condition in one of the other fitness

functions. As an arbitrary example, O2 feature 7 (pitch frequency) could have a target value of 0.0, meaning the melody would be only silence. This feature can never reach its target value without receiving a score of 0 in objective O1, since no pitches would be present in the phenotype for evaluation.

6.2. Configuration Taxonomy

The 17 different configurations can be classified into groups, and establishing a taxonomy can benefit the reader by easily providing context. The taxonomy is described in the following sections.

There are three areas that differ in the configurations. The target values in the melodic global objective, the data provided to the algorithm for generating the initial population, and the size of the phenotypes. Further, the provided data differ in two ways. A single piece of music was, or was not provided to the algorithm. Either a melody or a chord progression, and a varying amount of randomly generated phenotypes could be supplied. The varying sets of target values for objective O2 can be classified by their "Fitness Configuration", and are described in Section 6.3.

A string of four variables can be created to describe a configuration, and it has the following format: **FC-S-R-M**.

FC This is the fitness configuration for the algorithm run. There are a total of four different configurations, labeled F0 through F3 that are described more in depth in section 6.3.

S Describes a single piece of music that is provided to the algorithm for initialization. There was a total of six different pieces of material that was provided, which can be found in Table 6.2, along with their respective label. Three pieces of simple music was used, and split into melody and harmonization. "Lisa Gikk Til Skolen", an eight measure Norwegian nursery rhyme, that is frequently used as a first lesson for piano students. "She Loves You", a work of The Beatles from their album "Twist and Shout" album, released in 1964. Eight bars of the verse was used. Finally, the twelve measure verse of Britney Spears' breakthrough pop hit "...Baby One More Time", released in 1998, was used.

Table 6.2.: Description of labels for musical material provided to the algorithm.

Label	Type	Material Source
S0	-	No material
SM1	Melody	"Lisa Gikk Til Skolen"
SM2	Melody	"She Loves You", verse
SM3	Melody	"...Baby One More Time", verse
SC1	Harmonization	"Lisa Gikk Til Skolen"
SC2	Harmonization	"She Loves You", verse
SC3	Harmonization	"...Baby One More Time", verse

R This label simply describes how many randomly generated individuals were used in generating the initial population population. There are two variants: **R0**, which means zero individuals were provided, and **R10** that corresponds to ten randomly generated individuals being provided.

M The count of total measures in the developed phenotypes, represented by the corresponding number. Three different values were used: 8, 12, and 16.

An example description of a run-configuration would then be: F2-SM2-R10-8. The interpreted meaning is that fitness configuration F2 was used, the melody of "She Loves You" as well as 10 randomly generated individuals were provided for initialization, and the length of the phenotypes was eight measures.

6.3. Fitness Configuration Specifications

The Melodic Global Objective, O2, is guided by a set of target values. In this section, the four different configurations that were used during artefact generation are described.

6.3.1. F0 Configuration

This set of target values were defined by the author, as it qualitatively produced satisfying phenotypes during development of the fitness functions. Target values can be found in table 6.3.

Table 6.3.: Target values of the F0 configuration.

Feature	**Target Value**
Pitch Variety	0.30
Pitch Range	0.60
Step Movement	0.60
Non-Scale Pitch Quanta	0.00
Contour Stability	0.60
Contour Direction	0.70
Pitch Frequency	0.30
Rest Frequency	0.10
Rest Density	0.10
Rhythmic Variety	0.40
Syncopation	0.00
Repeated Pitches	0.30
Repeated Timings	0.65
On-Beat Pitch Coverage	0.70
Distinct Whole Measure Patterns	0.50
Distinct Half Measure Patterns	0.20
Positional Rhythmic Measure Repetitions	1.00
Positional Rest Measure Repetitions	1.00

6.3.2. F1 Configuration

The set of values obtained by analysing the melody of "Lisa Gikk Til Skolen" as a phenotype, were set to as the corresponding target value. Rounded target values are found in Table 6.4. At run-time the exact float values were used.

Table 6.4.: Rounded target values of the F1 configuration.

Feature	**Target Value**
Pitch Variety	0.27
Pitch Range	0.45
Step Movement	0.45
Non-Scale Pitch Quanta	0.00
Contour Stability	0.38
Contour Direction	0.39
Pitch Frequency	0.17
Rest Frequency	0.00
Rest Density	0.00
Rhythmic Variety	0.19
Syncopation	0.00
Repeated Pitches	0.50
Repeated Timings	0.67
On-Beat Pitch Coverage	0.69
Distinct Whole Measure Patterns	0.63
Distinct Half Measure Patterns	0.19
Positional Rhythmic Measure Repetitions	1.00
Positional Rest Measure Repetitions	0.00

6.3.3. F2 Configuaration

This configuration was found by analysing the "She Loves You" melody with the features from the O2 objective. The set of rounded values are found in table 6.5. During execution, the exact float values were used.

Table 6.5.: Target values of the F2 configuration.

Feature	**Target Value**
Pitch Variety	0.27
Pitch Range	0.60
Step Movement	0.45
Non-Scale Pitch Quanta	0.00
Contour Stability	0.43
Contour Direction	0.55
Pitch Frequency	0.23
Rest Frequency	0.02
Rest Density	0.08
Rhythmic Variety	0.31
Syncopation	0.07
Repeated Pitches	0.17
Repeated Timings	0.03
On-Beat Pitch Coverage	0.63
Distinct Whole Measure Patterns	0.5
Distinct Half Measure Patterns	0.44
Positional Rhythmic Measure Repetitions	1.00
Positional Rest Measure Repetitions	0.25

6.3.4. F3 Configuration

The following configuration was generated similarly to F1 and F2, with the target values extracted by analysis of the "...Oh Baby Baby" melody. Table 6.6 shows the rounded float values. The exact values were used during run-time.

Table 6.6.: Target values for F3 configuration.

Feature	**Target Value**
Pitch Variety	0.15
Pitch Range	0.95
Step Movement	0.4
Non-Scale Pitch Quanta	0.03
Contour Stability	0.43
Contour Direction	0.52
Pitch Frequency	0.31
Rest Frequency	0.00
Rest Density	0.00
Rhythmic Variety	0.31
Syncopation	0.10
Repeated Pitches	0.42
Repeated Timings	0.59
On-Beat Pitch Coverage	0.69
Distinct Whole Measure Patterns	0.5
Distinct Half Measure Patterns	0.29
Positional Rhythmic Measure Repetitions	0.50
Positional Rest Measure Repetitions	0.00

7. Results

A survey for evaluating artefacts produced by the algorithm is described int this chapter, along with the results of said survey.

7.1. Survey Design

A quantitative online study was performed to evaluate the generated artefacts. The main goal is to evaluate how different configurations, and achieved fitness values, impact how the musical ideas are perceived by a human audience.

Participants of the survey were presented one artefact generated by the system at a time, and asked to evaluate and score it in three criteria: *pleasantness, interestingness and randomness*. The assigneable values in each criteria were integers on a scale from 1 to 5, where 1 was defined as the lowest possible value, and 5 the highest. The terms were not explicitly described to the participant, but antonyms were provided. Respectively *unpleasant, boring and structured*. The artefacts themselves were also referred to as musical ideas, as they are not fully produced or arranged pieces of music.

Pleasantness is intended as a measure of how pleasing the musical piece is to the ear. This criterion does however not give a good measure of musical quality. There have been several paradigms, or genres, where unpleasant music has been regarded as "good" music by individual listeners. Some examples are contemporary classical music and death metal, which are notorious for using dissonances.

Interestingness is meant to help offset personal preferences when it comes to pleasantness, as it is possible to find music unpleasant but interesting. This criteria is however very subjective, and is possibly prone to a fatigue bias.

Finally, *randomness* is introduced to to give an indication of logic or

structure. This criteria is specifically introduced to help evaluate the pattern features that were designed for fitness objectives O2.

A text box accompanying each piece of music was also provided, where the participant could enter optional comments or observations. The participants were initially asked to describe their relationship with music, and examples such as consumer, musician, hobby musician, composer and producer were stated.

Pieces in the survey were all presented in a random order, both to eliminate possible biases on the artefacts presented last in the survey due to fatigue. Randomizing the order of presentation also helps getting a uniform distribution of data, if many participants do not complete the survey. An artefact would be presented a maximum of one time to each participant.

7.2. Survey Results

A total of 93 participants contributed to the survey, where 35 of them evaluated all 21 artefacts. A total of 884 evaluations were made. The average number of evaluations per artefact was 42 rounded down.

Tables in the following sections describe the scores received in the survey by average (AVG), standard deviation (SD) and the median (MED). Table 7.1 shows the aggregate scores from the the study for all the artefacts. Fitness features referred to in this chapter are all from the O2 objective, described in Section 5.4.2.

Table 7.1.: Aggregate values for all artefacts.

	Pleasant	**Interesting**	**Random**
AVG	3.256	3.048	2.550
SD	1.079	1.140	1.152
MED	3	3	2

7.2.1. Fitness Configuration Class Grouping

Grouping the results from the survey based on their fitness configuration class gives the results found in Table 7.2. Most of the average scores for

pleasantness do not big variations when grouped in this manner. Every fitness configuration scores slightly above 3 in terms of pleasantness, which is the absolute neutral in terms of the scale used for scoring.

Table 7.2.: Aggregate values, grouped by fitness configuration.

		Pleasing	**Interesting**	**Random**
F0	AVG	3.141	2.843	2.603
	SD	1.067	1.126	1.132
	MED	3	3	2.5
F1	AVG	3.138	2.693	2.334
	SD	1.161	1.095	1.148
	MED	3	3	2
F2	AVG	3.273	3.222	2.847
	SD	1.122	1.115	1.161
	MED	3	3	3
F3	AVG	3.088	3.300	2.773
	SD	1.053	1.137	1.138
	MED	3	3	3

In terms of interestingness, the F2 and F3 configurations outperform the other two. This is possibly due to the material the configurations were extracted from. F2 and F3 are extracted from what could be considered major "hits", while F0 is made up of arbitrary numbers. F1 is from a very simple nursery rhyme, that has very few notes, a very small pitch range, and very simple rhythms. F1 also scores the lowest of all configurations in terms of randomness.

The highest average score obtained in randomness by any of the configurations, is also the one considered the most pleasant, this is however not consistent with other classes discussed in this section.

7.2.2. Seed Material Class Grouping

If the results are grouped by their seed material class, some interesting numbers appear. In terms of previous research Freitas and Guimarães [2011b], the genetic material provided for initialization has a profound impact on the output of the algorithm. In this grouping, the class that is

provided melodies for generating the initial population does indeed score highest in both pleasantness and interestingness, and the lowest in randomness. This class consists of 6 artefacts, where all of them converged to a single solution. It is also noteworthy that 5 out these 6 melodies obtained a perfect fitness score. No other experiments achieved perfect fitness scores. This is likely caused by the fact that the fitness configurations use, were extracted to match the respective melodies 100%. This means that immediately after initialization of the algorithm, the phenotypes of the seeded melodies already have the maximum score in the O2 objective.

Table 7.3.: Aggregate values, grouped by seed material type provided for intialization.

		Pleasing	**Interesting**	**Random**
S0	AVG	3.190	3.000	2.740
	SD	1.059	1.110	1.090
	MED	3	3	3
SM	AVG	3.253	3.274	2.359
	SD	1.250	1.189	1.179
	MED	3	3	2
SC	AVG	3.142	2.988	2.700
	SD	1.024	1.110	1.181
	MED	3	3	3

It might seem that perfect fitness scores result in the best evaluations, but computing the correlation between the fitness scores and the evaluation results reveal that there likely is none. Table 7.4 holds the correlation coefficients, and Figure 7.1(a), 7.1(b) and 7.1(c) show the relating scatterplots. It is worthy to note that all but one of the generated artefacts scored over 17.0 in the O2 objective, when the maximum is 18.0.

Table 7.4.: Correlation between O2 fitness score and survey score.

	O2 Fitness Score
Pleasant	-0,0175571708
Interesting	0,0558052797
Random	-0,1665035061

7.2.3. Random Seeds Class Grouping

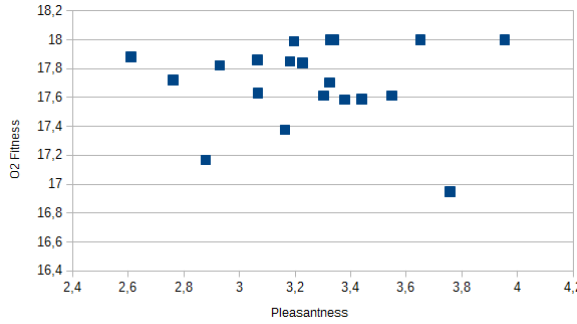
This classification shows the lowest amount of differentiation in scores between the configurations. This is interesting as it implies that the algorithm is able to reach similar solutions regardless of noise in the starting material. Also note that all the phenotypes that were develop without any musical material started with ten randomly generated individuals. The scores can be found in Table 7.5

Table 7.5.: Aggregate values, grouped by the amount random individuals provided for initialization.

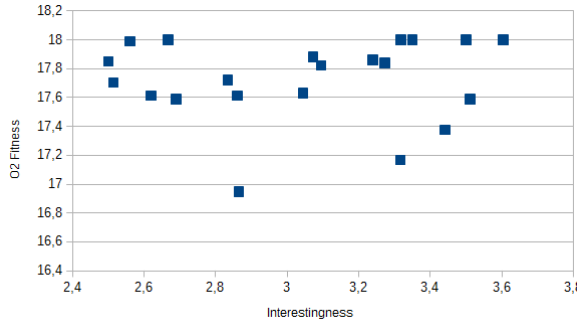
		Pleasant	**Interesting**	**Random**
R0	AVG	3.307	3.096	2.535
	SD	1.074	1.140	1.178
	MED	3	3	2
R10	AVG	3.229	3.022	2.558
	SD	1.082	1.141	1.140
	MED	3	3	2

7.2.4. Phenotype Size Class Grouping

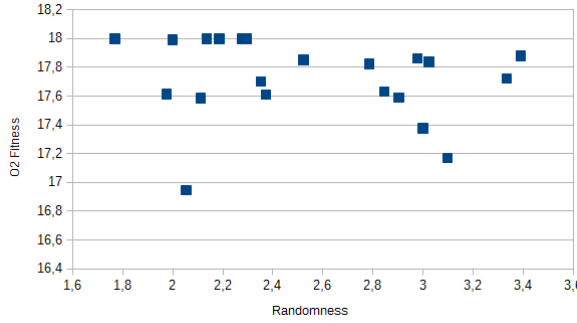
If you look at the values by grouping the results of the phenotype size class, described in Table 7.6 it might appear at first glance as if pleasantness falls as size increases, and the opposite with randomness. This might be caused by varying sample sizes in terms of artefacts. The "8" class group consists of 14 artefacts, the "12" group of 6, and the final "16" group of a single individual.



(a) Pleasantness



(b) Interestingness



(c) Randomness

Figure 7.1.: Scatterplots of O2 fitness to survey average scores.

Table 7.6.: Aggregate values, grouped by the phenotype sizes.

		Pleasant	Interesting	Random
8	AVG	3.357	2.916	2.412
	SD	1.090	1.121	1.142
	MED	3	3	2
12	AVG	3.089	3.300	2.773
	SD	1.053	1.137	1.138
	MED	3	3	3
16	AVG	2.878	3.317	3.098
	SD	0.872	1.150	1.020
	MED	3	3	3

7.2.5. Rank 1 Size Grouping

Out of the 21 runs of the algorithm, 11 of them converged on a single non-dominated phenotype. Five of these converged to a single fitness perfect solution. All of these artefacts were created by seeding a melody, the SMx configurations, and already had a perfect score in the O2 objective. In the remaining cases the first non-dominated front held between 4 and 22 phenotypes.

Table 7.7.: Aggregate scores, grouped by convergence to a single phenotype.

		Pleasant	Interesting	Random
Single	AVG	3.367	3.085	2.409
	SD	1.113	1.156	1.141
	MEDIAN	3	3	2
Multiple	AVG	3.134	3.007	2.703
	SD	1.028	1.122	1.147
	MEDIAN	3	3	3

7.2.6. Pattern Feature Correlations

Features 15 through 18 in objective O2 are the pattern matching features developed for this thesis. The correlation coefficients between their achieved fitness score and survey score are located in Table 7.8.

Table 7.8.: Fitness pattern features correlation to survey average scores.

Feature no.	**15**	**16**	**17**	**18**
Pleasant	-0.0091	0.2087	0.3126	0.1269
Interesting	0.6956	0.7528	-0.6158	0.0916
Random	0.4218	0.3405	-0.4068	0.0603

While feature 18, the positional rest measure repetitions feature, shows no sign of correlation, the other three do. Feature 15, 16 and 17, have a statistically significant correlation to the interestingness score at a 99.5% confidence interval. Feature 15 and 17 have a statistically significant correlation to randomness at a 95% confidence interval. Feature 15 and 16, Whole Measure Distinctness and Half Measure Distinctness respectively, correlate positively with interestingness, which means the more distinct patterns appear, the more interesting the pieces are perceived. It is very doubtful that this correlation is of a linear nature, but within the range of values for the fitness features obtained by the artefacts it might be. All the feature scores were below 0.5. Feature 17, labeled positional rhythmic measure repetitions, has a negative correlation to both interestingness and randomness. These correlations are quite confusing. When feature 17 gets close to its maximum value, it means that a very rigid structure is observed, where the exact rhythmical patterns are repeated at given positions later in the piece. A low value merely the means the absence of the exact patterns it detects.

7.2.7. Participant Sentiments and Qualitative Observations

The sentiments collected from the optional comment boxes in the survey can provide some interesting opinions. The sentiments can be found in Appendix A, accompanied by the sheet music and scores obtained in the survey. Enjoyment and appreciation of music is a very subjective experience. A self-labeled "Consumer" left a comment on the last question:

"Not that much good music", and had consistently given scores below 2 for pleasantness and interestingness. Another participant commented on piece S3-S0-R10-12.1: *"Some weird melody beats... Some times very standard, but other times suddenly weird. "*, and scored it 4 in randomness, but rated it at 5 in interestingness and 4 in pleasing. By looking at the sheet music, which is located in Figure 7.2, you can see that it has some unusual rhythmical phrasings in measure two and three, which are also not repeated later in the piece. This sentiment is somewhat re-iterated by another participant who wrote: *"First 10 seconds sound like an intro, but after that is a bit more expected"*.

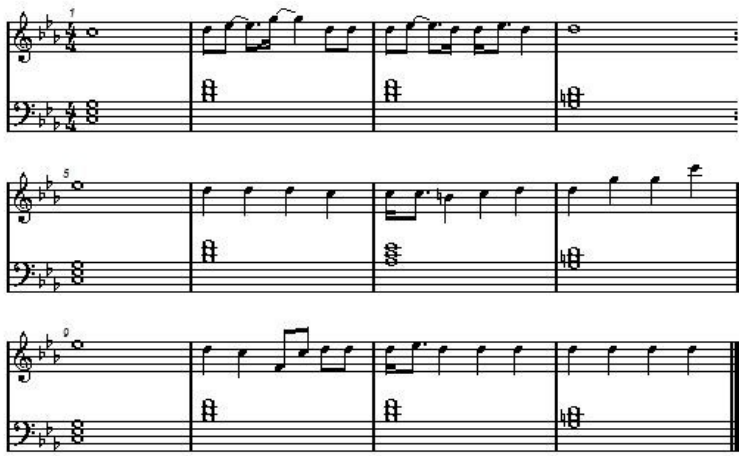


Figure 7.2.: Generated music by the first F3-S0-R10-12.1 run.

Pieces that were generated without melodic seeds, sported rhythmical phrases that were critiqued at times. This was however not always the case. A comment on piece F2-S0-R10-8.1 states: *"Very good melody line. Nice rhythm on that. "* A possible cause of this is that the features that measure rhythms do not reward good rhythmical phrases, nor punish bad ones. Something interesting to note, is that all the pieces generated with a melodic seed were recognized by at least one participant, often more.

8. Discussion and Conclusion

This chapter discusses the obtained results, and makes a conclusion in Section 8.2, and finally, suggestions for future works is proposed.

8.1. Discussion

In the introduction, two goals were introduced for this Master's Thesis:

G1: Develop a Novel and Useful Algorithm for Composition

G2: Develop an Automated Fitness Function

Whether these these goals were reached or not, will be discussed in the following sections.

8.1.1. G1: Develop a Novel and Useful Algorithm for Composition

In terms of novelty, systems within algorithmic composition that make both harmonization and melody is not unprecedented. Evolutionary algorithms that generate either melodies or harmonization are fairly common, but developing both in tandem is not common. They are usually developed sequentially.

Developing a multiple-objective evolutionary algorithm (MOEA) for co-evolving melody and harmonization provided quite a few obstacles. Simply making the algorithm converge on a set of solutions that were acceptable to humans proved challenging. The state space of possible solutions for genotypes of eight measures is of size 10^{205} in magnitude, and 10^{410} for

16 measures. This is caused by developing both melody and harmonization at the same time, as the state space grows multiplicatively. For eight measures of melody exclusively, the state space is 34 orders of magnitude smaller than when combined with a harmonization. Also the use of $\frac{1}{16}$ notes increase the state space tremendously. The size of the state space could be dramatically reduced by only allowing notes of $\frac{1}{8}$ or bigger. However, supporting such a large state space is necessary to allow a wide range of musical expressions.

Allowing the algorithm to create such a huge variety of phenotypes can aid in evaluation of fitness functions on a general basis, in relation to human perception of music. If the algorithm is capable of representing what could be considered as "bad music", it is possible to validate fitness functions and features if they guide the search towards what is considered "good" solutions, or "good music".

Having to apply the step of the algorithm that removes fitness duplicates is something that could be considered a weakness of the implementation, as it confines the algorithm to explore a relatively small part of the state space on any given run. If several equal best solutions in a maxima do exist, only one will be kept. Removing genetically identical phenotypes, instead of fitness identical, might be more desirable. It would also allow the algorithm to escape local maxima more frequently. However, this was not feasible with while working with four fitness objectives computationally. Not eliminating fitness duplicates also has the benefit of providing options to anyone who is using the algorithm. Having a set of slightly different solutions, that is considered equally good, could provide any user of the implementation with more ideas, more inspiration.

No correlation was found between the O2 fitness objective scores and the human evaluations of the music. While little difference, in the perceived quality of the pieces, was found between the artefacts that maxed out the O2 score, and the ones that didn't. I does not mean that this is not desirable. Getting closer to the maximum score of the O2 objective could mean getting closer to a desired result, if a user was to tweak the parameters to their own preference.

While the algorithm did generate pieces of music that were appreciated by participants of the study, it was not able to do so consistently. The results proved to be slightly more pleasant and interesting when melodic

material was provided for initialization, but no evidence was found as to conclude why.

8.1.2. G2: Develop an Automated Fitness Function

Integrating four fitness functions on a new model, spurred quite a few unforeseen interactions between themselves and the model. The most apparent problem with co-evolving melody and harmonization was to the lack of harmonic context needed by the individual fitness objectives. Pitches that are outside of the musical key are not punished in the melodic harmonic objective, as long as the same pitch occurs in the chord of the measure. The same is also true for the harmonization objective, where a non-scale pitch in a chord is not punished if it occurs in the melody. The result of this interaction was that both the melody and harmonization disregarded the musical key it was supposed to compose in, and generated very atonal pieces.

While this is not inherently wrong, a goal of this project was to implement an algorithm that can generate useful music for humans. If the algorithm was not able to work within a given key, it loses its usefulness in many situations. The harmonization objective was made very rigid in terms of punishing chords that were not triads in the key, to mitigate the atonal tendencies. If a slightl atonal piece is desired, then non-scale pitch quanta feature can be set to a higher value, to provide an atonal context to the harmonization functions. The harmonic progression objective had two rules developed to encourage variation of chord roots in the generated pieces, namely the "excessive chord repetition" rule and the "harmonic variety feature". Without any of these features, the generated phenotypes would often consist of one single chord, being repeated for every measure. Limiting repetitions by punishing three consecutive repetitions of the same chord root, caused the music pieces to vary between two chords. This incited the addition of the harmonic variety rule.

The F1 objective for melodic harmony does not take rhythm or note frequency into consideration, and would award melodies flooded with notes, as they would satisfy its constraints. This objective is based on a fitness function that was intended for altering melodies with fixed note positions, and did not work entirely as intended. The second melodic objective, F2, mitigates this problem with the note frequency feature. It does however

disregard local distribution of notes. As a consequence, some measures in the generated music may contain melodic phrases with a very high note density, that might appear "out of place" to the listener. This is supported by the evidence that artefacts generated with a melody for initialization generally performed better, where an even and purposeful distribution of the notes appear naturally. Overall, the fitness objectives lack features or rules that reward "good" rhythmical phrases.

Three of the four developed fitness features for pattern matching on melodies showed strong correlations to the interestingness score, where whole measure distinctness and half measure distinctness both showed a positive correlation. Whether or not more distinct patterns cause the music to be more interesting, is inconclusive, but one could be inclined to believe that more variation in the melodies cause them to be more interesting.

The positional rhythmic measure repetitions feature has a negative correlation to interestingness. This feature is very strict when matching patterns to positions. Any variation in the rhythm will be interpreted as a mismatch. It only accepts perfect matches of the rhythmical phrases. It is important to note that a fitness value of 0 does not mean that no pattern structure occurs, but rather a distinct absence of the particular structures it detects. The correlation might be caused by the fact that this feature creates a boring pattern as it gets closer to 1. Altering this feature to accept less than perfect matches might cause it evaluate to 1 for a large amount of melodies, and having a target value for the feature would be meaningless.

A few of the participant sentiments stated that an artefact was "not good", or even bad. There was however more positive sentiments than bad ones. If the average scores for pleasantness, interestingness and randomness are considered together with the sentiments, it is reasonable to conclude that the fitness functions are able to measure human aesthetic to some extent.

8.2. Conclusion

A novel approach to generating musical material with a multi-objective evolutionary algorithm was implemented, along with a set of 43 fitness measures split into four objectives. The implementation was then used to produce 21 pieces of music with different configurations, which were evaluated in a study.

The implementation is not able to write music that is perceived by everyone to be aesthetically good consistently. The best results were achieved when some melodic material was supplied during the initialization phase. Features that evaluate rhythmical phrasing is lacking in the fitness functions, and the performance of the system could possibly improve by introducing new features that guide the search of the algorithm towards good rhythmical phrasing.

While the implementation shows that it is feasible to develop both harmonization and melodies in parallel with an evolutionary algorithm, the benefits of doing so are not prominent besides the ability for both harmonization and melody to adapt to each other during generation.

Due to the large search space and relatively high amount of fitness objectives, computational viability was hard to achieve. To do this, only short musical ideas were developed, and a scheme for removing duplicates in the population was implemented, which might not be desirable.

Overall the algorithm will probably never write any musical masterpiece, but evaluations made by participants of the study indicate that some of the generated pieces were subjectively enjoyable. This suggests that the implementation could be useful in terms computer assisted composition.

8.3. Possible Future Work

Efforts could be made in developing fitness features that concern melodic phrases, especially rhythms. Toussaint [2010] suggests ways of generating "good" musical rhytms, and Toussaint et al. [2012] contains a variety features and concepts that could be adopted.

The most interesting prospect, in regards to future work, is to integrate a feasible–infeasible two-population approach to the implementation [Kimbrough et al., 2008]. This was recently done in the MetaCompose system

Scirea et al. [2016]. By adopting this approach, all the fitness objectives but the F2 objective could be used as the constraints for separating the populations, allowing other objectives to be introduced to the algorithm. One such objective could work on rhythms.

Bibliography

H. E. Aguirre and K. Tanaka. Selection, drift, recombination, and mutation in multiobjective evolutionary algorithms on scalable mnk-landscapes. In *Evolutionary Multi-Criterion Optimization*, pages 355–369. Springer, 2005.

C. Bell. Algorithmic music composition using dynamic markov chains and genetic algorithms. *Journal of Computing Sciences in Colleges*, 27(2): 99–107, 2011.

J. A. Biles. Genjam: A genetic algorithm for generating jazz solos. In *Proceedings of the International Computer Music Conference*, 1994.

J. A. Biles. Autonomous genjam: eliminating the fitness bottleneck by eliminating fitness. In *Proceedings of the 2001 Genetic and Evolutionary Computation Conference Workshop Program, San Francisco*, 2001.

J. A. Biles, P. Anderson, and L. Loggi. Neural network fitness function for a musical IGA. In *Proceedings of the International Symposium on Intelligent Industrial Automation and Soft Computing.*, 1996.

N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent. Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. *arXiv preprint arXiv:1206.6392*, 2012.

Chun-Chi J. Chen and R. Miikkulainen. Creating melodies with evolving recurrent neural networks. In *Neural Networks, 2001. Proceedings. IJCNN'01. International Joint Conference on*, volume 3, pages 2241–2246. IEEE, 2001.

P. P. Cruz-Alcázar and E. Vidal-Ruiz. Learning regular grammars to model musical style: Comparing different coding schemes. In *Grammatical Inference*, pages 211–222. Springer, 1998.

A. R.R. de Freitas, F. G. Guimaraes, and R. V. Barbosa. Ideas in automatic evaluation methods for melodies in algorithmic composition. 2012.

R. De Prisco, G. Zaccagnino, and Rocco Zaccagnino. Evobasscomposer: a multi-objective genetic algorithm for 4-voice compositions. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, pages 817–818. ACM, 2010.

K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *Evolutionary Computation, IEEE Transactions on*, 6(2):182–197, 2002.

Kalyanmoy Deb. Multi-objective optimization. In *Search methodologies*, pages 403–449. Springer, 2014.

K. Ebcioğlu. An expert system for harmonizing four-part chorales. *Computer Music Journal*, pages 43–51, 1988.

D. Eck and J. Schmidhuber. A first look at music composition using lstm recurrent neural networks. *Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale*, 2002.

J. D. Fernández and F. Vico. AI methods in algorithmic composition: A comprehensive survey. *Journal of Artificial Intelligence Research*, 48: 513–582, 2013.

J. Feulner and D. Hörnel. Melonet: Neural networks that learn harmony-based melodic variations. In *Proceedings of the International Computer Music Conference*, pages 121–121. INTERNATIONAL COMPUTER MUSIC ACCOCIATION, 1994.

A. Freitas and F. Guimarães. Melody harmonization in evolutionary music using multiobjective genetic algorithms. In *Proceedings of the Sound and Music Computing Conference*, 2011a.

A. Freitas and F. Guimarães. Originality and diversity in the artificial evolution of melodies. In *Proceedings of the 13th annual conference on Genetic and Evolutionary Computation*, pages 419–416, 2011b.

P. Galanter. The problem with evolutionary art is... In *Applications of Evolutionary Computation*, pages 321–330. Springer, 2010.

P. M. Gibson and J. A. Byrne. Neurogen, musical composition using genetic algorithms and cooperating neural networks. In *Artificial Neural Networks, 1991., Second International Conference on*, pages 309–313. IET, 1991.

J. Gillick, K. Tang, and R. M. Keller. Learning jazz grammars. In *Proceedings of the Sound and Music Computing Conference*, pages 125–130, 2009.

H. Hild, J. Feulner, and W. Menzel. Harmonet: A neural net for harmonizing chorales in the style of js bach. In *Advances in Neural Information Processing Systems*, pages 267–274, 1992.

L. A. Hiller and L. M. Isaacson. Musical composition with a high-speed digital computer. *Journal of the Audio Engineering Society*, 6(3):154–160, 1958.

A. Horner and D. E. Goldberg. Genetic algorithms and computer assisted music composition. In *Proceedings of the International Conference on Genetic Algorithms*, pages 337–441, 1991.

J. H. Jeong and C. W. Ahn. Automatic evolutionary music composition based on multi-objective genetic algorithm. In *Proceedings of the 18th Asia Pacific Symposium on Intelligent and Evolutionary Systems*, volume 2, pages 105–115, 2015.

M. Johnson, D. R. Tauritz, and R. Wilkerson. Evolutionary computation applied to melody generation. In *Proceedings of the Artificial Neural Networks in Engineering (ANNIE) Conference*, 2004.

Y. Khalifa, B. K Khan, J. Begovic, A. Wisdom, and A. M. Wheeler. Evolutionary music composer integrating formal grammar. In *Proceedings of the 9th annual conference companion on Genetic and evolutionary computation*, pages 2519–2526. ACM, 2007.

S. O. Kimbrough, G. J. Koehler, Ming Lu, and D. H. Wood. On a feasible–infeasible two-population (fi-2pop) genetic algorithm for constrained optimization: Distance tracing and no free lunch. *European Journal of Operational Research*, 190(2):310–327, 2008.

V. M. Marques, V. Oliveira, S. Vieira, and A. C. Rosa. Music composition using genetic evolutionary algorithms. In *Proceedings of the IEEE International Conference on Evolutionary Computation*, pages 714–719, 2000.

J. McCormack. Grammar based music composition. *Complex systems*, 96: 321–336, 1996.

J. McCormack. Open problems in evolutionary music and art. In *Applications of Evolutionary Computing*, pages 428–436. Springer, 2005.

MIDI Manufacturers Association et al. *The complete MIDI 1.0 detailed specification: incorporating all recommended practices*. MIDI Manufacturers Association, 1996.

J. A. Moorer. Music and computer composition. *Communications of the ACM*, 15(2):104–113, 1972.

A. Moroni, J. Manzolli, F. V. Zuben, and R. Gudwin. Vox populi: An interactive evolutionary system for algorithmic composition. *Leonardo Music Journal*, 10:49–54, 2000.

G. L. Nelson. Sonomorphs: An Application of genetic algorithms to the growth and development of musical organisms. In *Proceedings of the Biennial Symposium on Arts and Technology*, pages 155–169, 1994.

G. Papadopoulus and G. Wiggins. A genetic algorithm for the generation of jazz melodies. In *Proceedings of the Finnish Conference on Artificial Intelligence (STeP)*, 1998.

G. Papadopoulus and G. Wiggins. AI methods for algorithmic composition: A survey, a critical view and future prospects. In *Proceedings of the Symposium on Musical Creativity*, pages 110–117, 1999.

S. Phon-Amnuaisuk, A. Tuson, and G. Wiggins. Evolving musical harmonisation. In *Artificial Neural Nets and Genetic Algorithms*, pages 229–234. Springer, 1999.

G. M. Rader. A method for composing simple traditional music by computer. *Communications of the ACM*, 17(11):631–638, 1974.

K. Ricanek, A. Homaifar, and G. Lebby. Genetic algorithm composes music. In *Proceedings of the Southeastern Symposium on System Theory*, pages 223–227, 1993.

J. E. Rothgeb. *Harmonizing the unfigured bass: A computational study.* PhD thesis, Yale Univ., Dept.of Music, New Haven. Conn., 1969.

M. Scirea, G. Barros, N. Shaker, and J. Togelius. Smug: Scientific music generator. In *Proceedings of the Sixth International Conference on Computational Creativity June*, page 204, 2015.

M. Scirea, J. Togelius, P. Eklund, and S. Risi. Metacompose: A compositional evolutionary music composer. In *International Conference on Evolutionary and Biologically Inspired Music and Art*, pages 202–217. Springer, 2016.

C. Sánchez-Quintana, F. Moreno-Arcas, D. Albarracín-Molina, J. D Fernández, and F. Vico. Melomics: A case-study of AI in Spain. *AI magazine*, 34:99–103, 2013.

K. Thywissen. Genotator: an environment for exploring the application of evolutionary techniques in computer-assisted composition. *Organised Sound*, 4(02):127–133, 1999.

P. M. Todd. A connectionist approach to algorithmic composition. *Computer Music Journal*, pages 27–43, 1989.

G. T. Toussaint. Generating "good" musical rhythms algorithmically. In *Proceedings of the 8th International Conference on Arts and Humanities, Honolulu, Hawaii*, pages 774–791, 2010.

G. T. Toussaint, L. Matthews, M. Campbell, and N. Brown. Measuring musical rhythm similarity: Transformation versus feature-based methods. *J. Interdiscip. Music Stud*, 6:23–53, 2012.

M. W. Towsey, A. R. Brown, S. K. Wright, and J. Diedereich. Towards melodic extension using genetic algorithms. *Educational Technology & Society*, 4(2):54–65, 2001.

Chia-Lin Wu, Chien-Hung Liu, and Chuan-Kang Ting. A novel genetic algorithm considering measures and phrases for generating melody. In *Evolutionary Computation (CEC), 2014 IEEE Congress on*, pages 2101–2107. IEEE, 2014.

A. Sheet Music and Sentiments

In this appendix, the sheet music for the generated music can be found, along with the sentiments and achieved average evaluations from the survey as [pleasant, interesting, random].

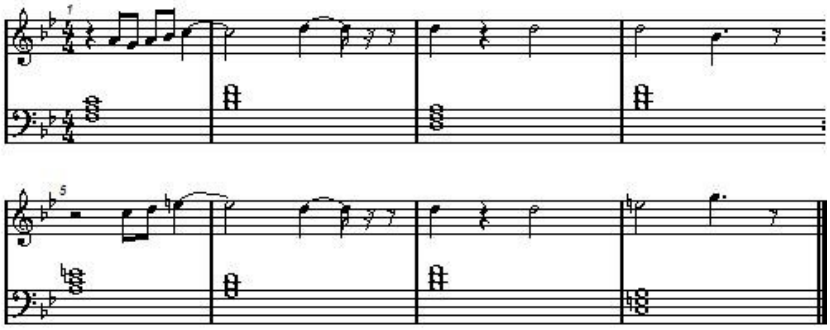


Figure A.1.: Sheet music for F0-S0-R10-8.1.

Survey scores: [3.757, 2.865, 2.054].

F0-S0-R10-8.1 sentiments:

- "Quite OK, It is just very hard to judge music from just 16 seconds. As a listener, I mentally add a temporal context."

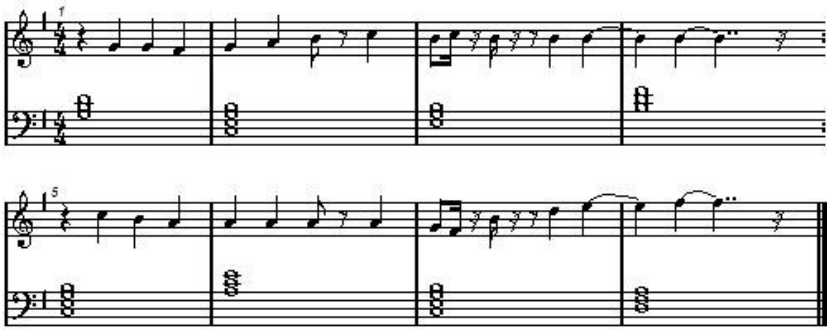


Figure A.2.: Sheet music for F0-S0-R10-8.2.

Survey scores: [3.302, 2.860, 2.372].

F0-S0-R10-8.2 had no sentiments.

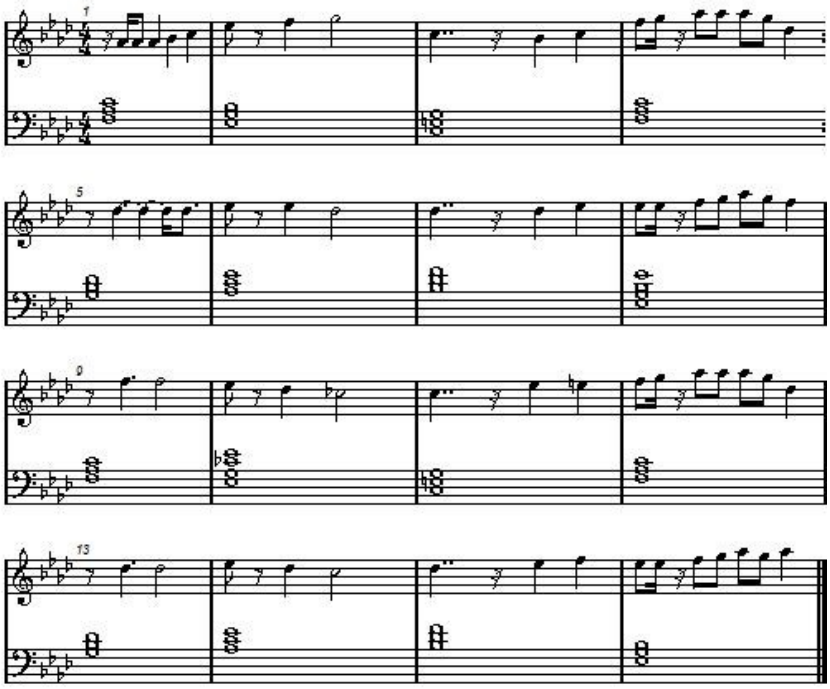


Figure A.3.: Sheet music for F0-S0-R10-16.

Survey scores: [2.879, 3.317, 3.098].

F0-S0-R10-16 sentiments:

- "Very interesting!"

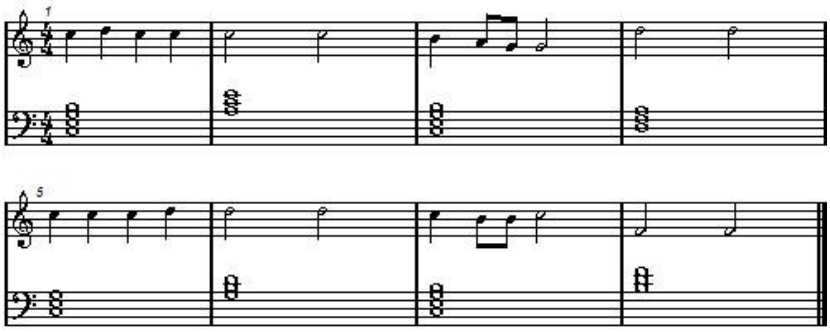


Figure A.4.: Sheet music for F1-S0-R10-8.1.

Survey scores: [3.182, 2.500, 2.523].

F1-S0-R10-8.1 sentiments:

- "well the end chord was unexpected..."

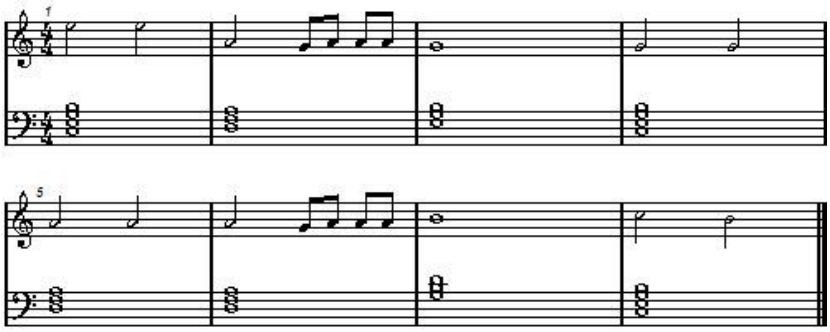


Figure A.5.: Sheet music for F1-S0-R10-8.2.

Survey scores: [3.324, 2.514, 2.351].

F1-S0-R10-8.2 sentiments:

- "Fantastic last chord."

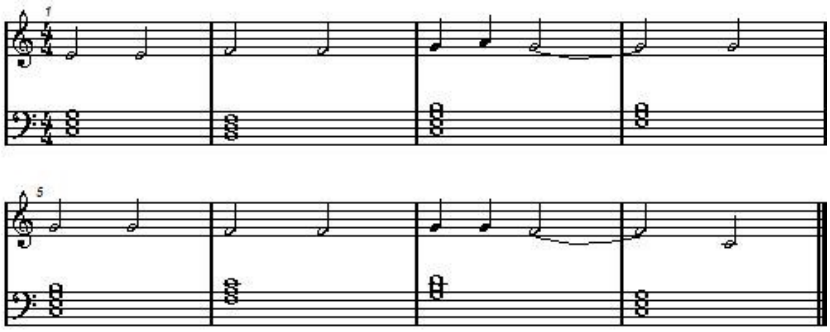


Figure A.6.: Sheet music for F1-SC1-R0-8.

Survey scores: [3.378, 2.689, 2.111].

F1-SC1-R0-8 sentiments:

- "Standard rhythm on the melody line."
- "Seems like a 4 year old plays the piano."

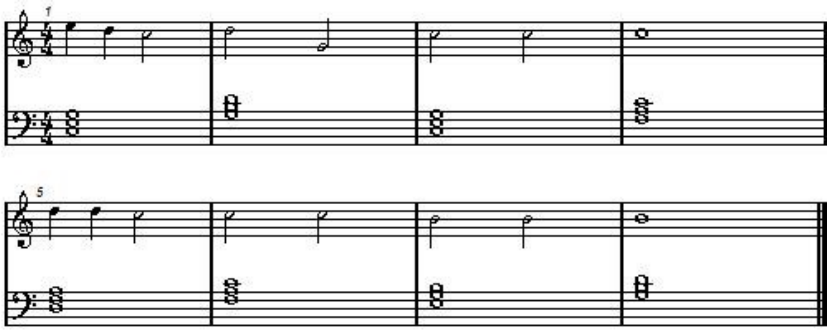


Figure A.7.: Sheet music for F1-SC1-R10-8.

Survey scores: [3.548, 2.619, 1.976].

F1-SC1-R10-8 sentiments:

- "A bit standard."
- "This was quite boring. The chord progression felt unsurprising all the way and the melody didn't bring much to it either."

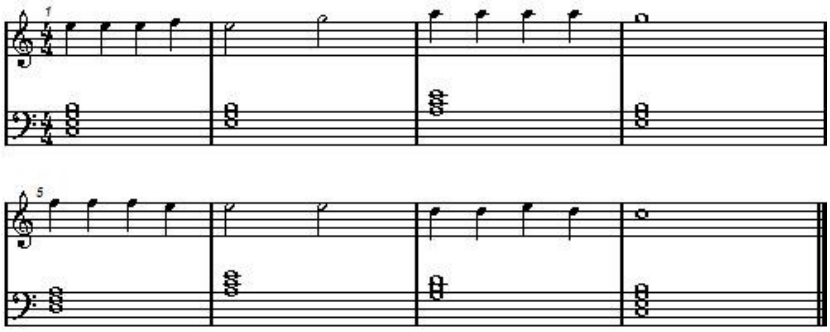


Figure A.8.: Sheet music for F1-SM1-R0-8.

Survey scores: [3.195, 2.561, 2.000].

F1-SM1-R0-8 sentiments:

- "Seems like a pretty classic progression."
- "Lisa gikk til skolen", 5 times.

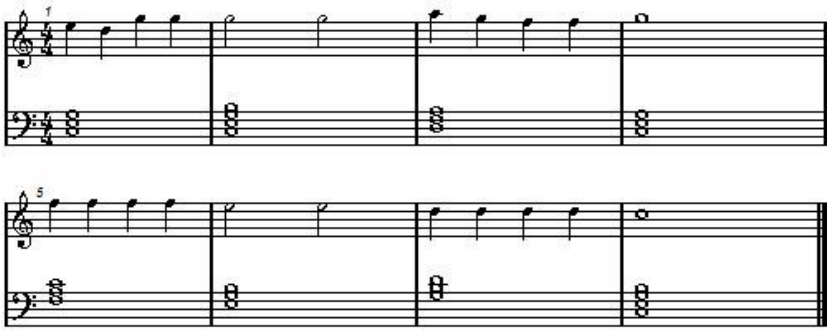


Figure A.9.: Sheet music for F1-SM1-R10-8.

Survey scores: [3.333, 2.667, 1.769].

F1-SM1-R0-8 sentiments:

- "Is there maj7 chords making it a lot more interesting??"
- "Lisa gikk til skolen", 7 times.

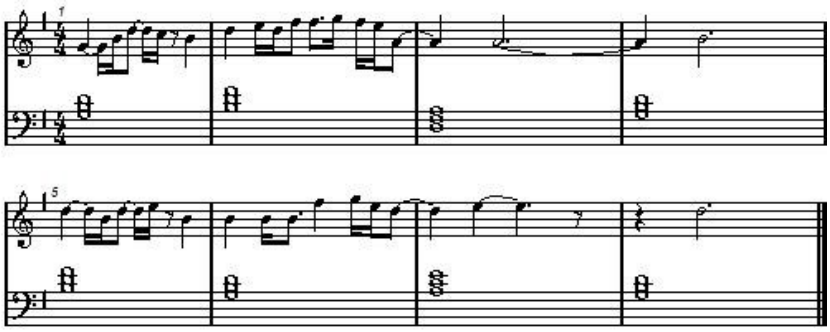


Figure A.10.: Sheet music for F2-S0-R10-12.1.

Survey scores: [3.439, 3.512, 2.902].

F2-S0-R10-12.1 sentiments:

- "Very good melody line. Nice rhythm on that."
- "Awesome!"
- "Melody is very nice! It is more alive in both ascending/descending manner and tone length. Chords are more boring and didn't create much tension or enthusiasm."
- "Despite the scores, it sounds it could have been a human pianist (under severe influence of substances)".

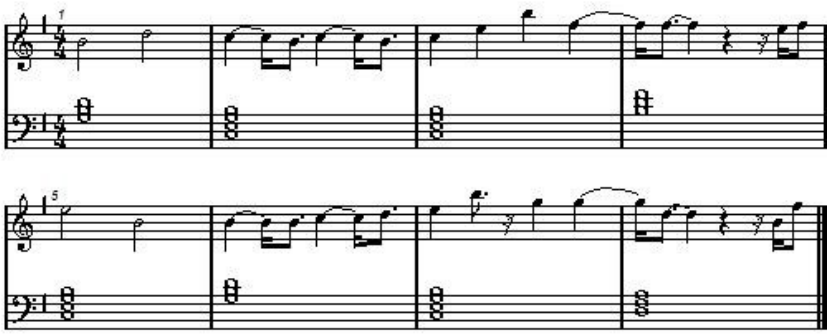


Figure A.11.: Sheet music for F2-S0-R10-12.2.

Survey scores: [3.067, 3.044, 2.844].

F2-S0-R10-12.2 sentiments:

- "Such a sweet melody!"
- "Very nice melody line! The high note did everything! (along with a good chord progression)."

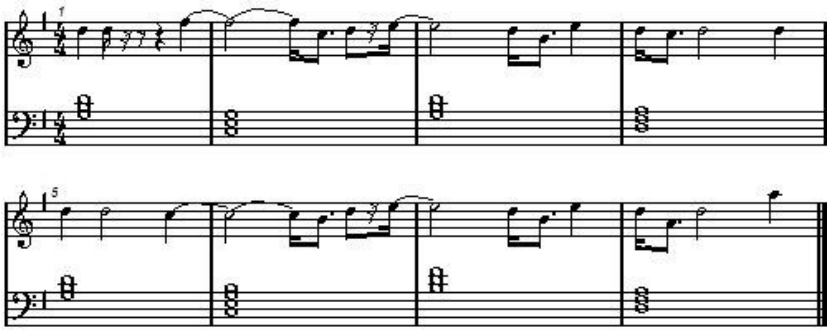


Figure A.12.: Sheet music for F2-SC2-R0-12.

Survey scores: [2.762, 2.833, 3.333].

F2-SC2-R0-12 sentiments:

- "A bit off beat?"

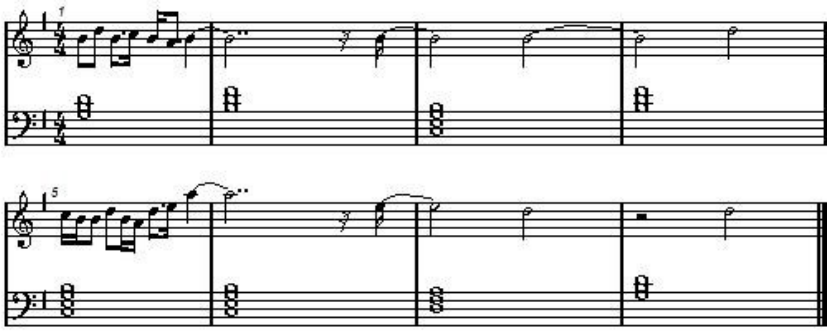


Figure A.13.: Sheet music for F2-SC2-R10-12.

Survey scores: [3.163, 3.442, 3.000].

F2-SC2-R10-12 had no sentiments.

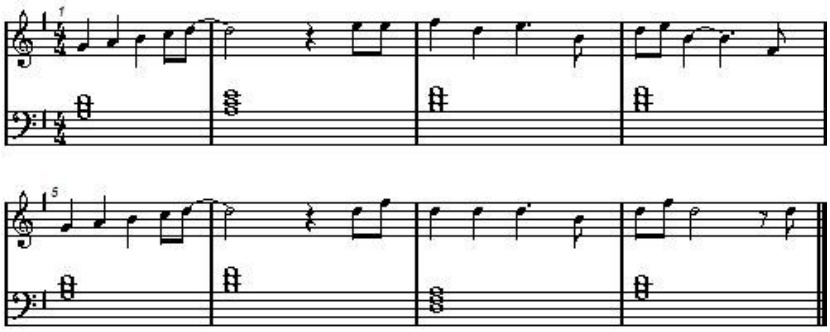


Figure A.14.: Sheet music for F2-SM2-R0-12.

Survey scores: [3.955, 3.318, 2.136].

F2-SM2-R0-12 sentiments:

- "The chord progression was a little weird."
- "Happy and nice."
- "Beatles", two times.

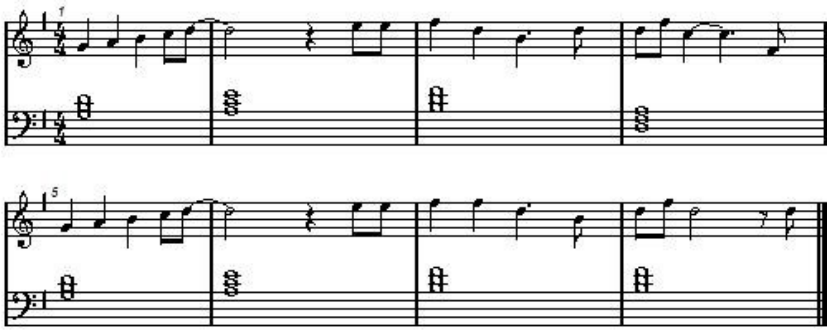


Figure A.15.: Sheet music for F2-SM2-R10-12.

Survey scores: [3.650, 3.350, 2.275].

F2-SM2-R10-12 sentiments:

- "Nice!"
- "Min første kjærlighet?"
- "Sounded like Beatles."

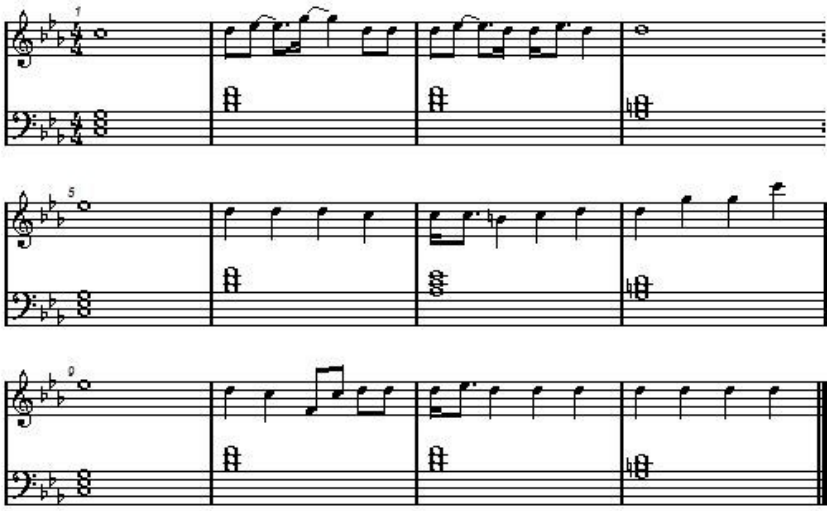


Figure A.16.: Sheet music for F3-S0-R10-12.1.

Survey scores: [3.227, 3.273, 3.023].

F3-S0-R10-8.1 sentiments:

- "Pretty random phrases."
- "Some weird melody beats... Some times very standard, but other times suddenly weird."
- "First 10s sounds like an intro, but after that is a bit more expected."
- "Far from adequate, however still quite impressive for a machine."

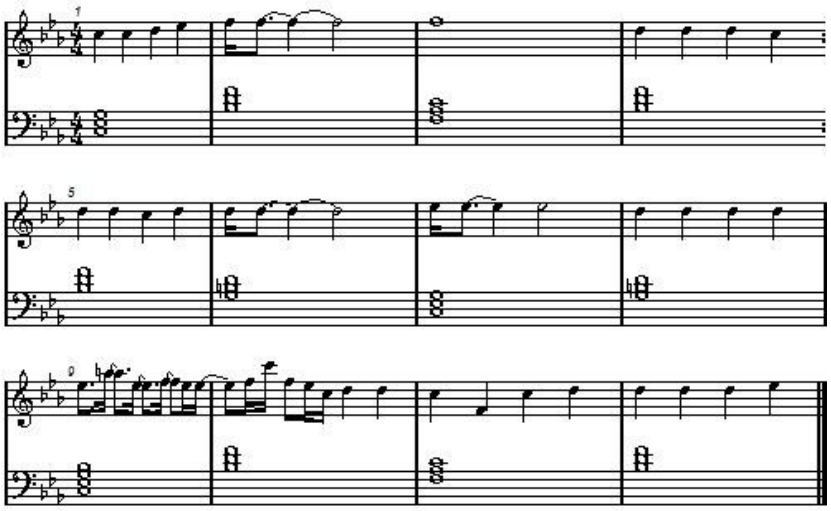


Figure A.17.: Sheet music for F3-S0-R10-12.2.

Survey scores: [2.610, 3.073, 3.390].

F3-S0-R10-12.2 sentiments:

- "A little messy with the melody and chords."
- "Some stuff i really didn't expect here..."
- "First 7 sec is sweet classical sounding with a nice minor chord where I suspected major. Then the beat felt like it stumbled, and it all got a bit random. And from there on it never settled on one idea."
- "It sounds like some keys are hit without intention (like a beginner playing)."
- "Poorly written piano play."

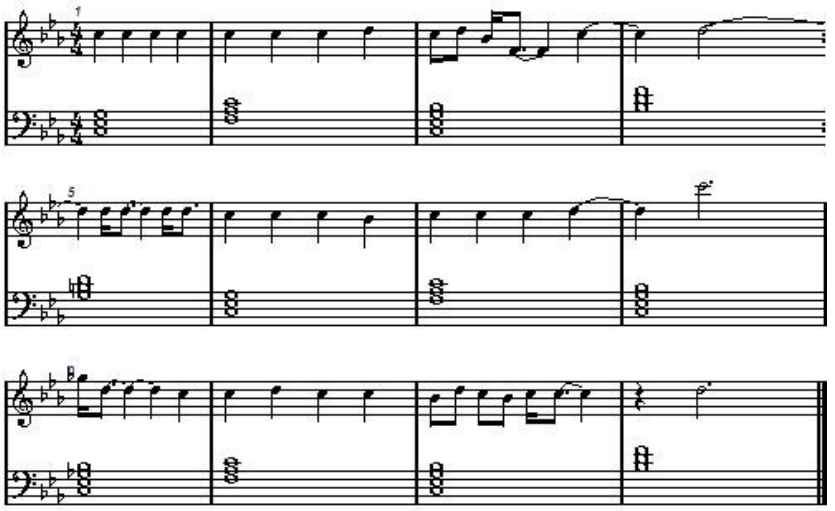


Figure A.18.: Sheet music for F3-SC3-R0-12.

Survey scores: [3.065, 3.239, 2.978].

F3-SC3-R0-12 sentiments:

- "I like this one! The chords are far more interesting and the length and pace of the notes are more dynamic."
- "A bit boring rhythm on the melody line."
- "Sounds quite human-like, but a few details gives it away; i.e. a number of very short notes. These details have no link back to (Western) musical tradition (not Bach, not Palestrina, not even popular music tradition)."

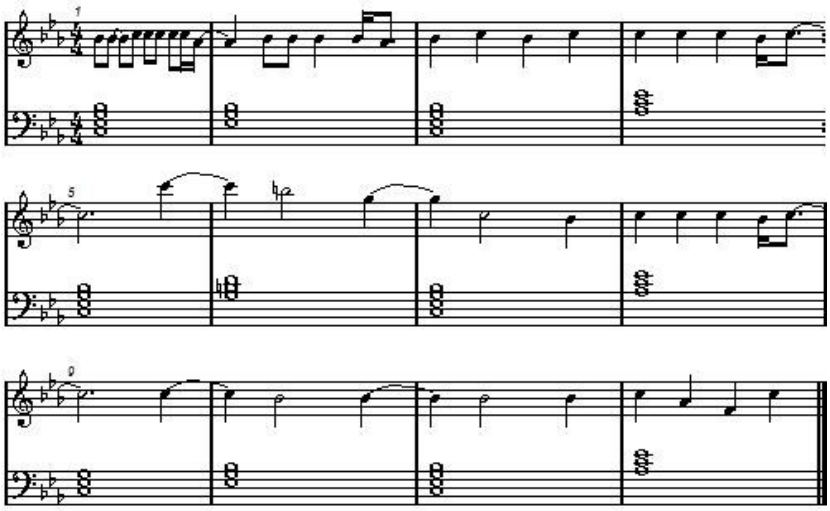


Figure A.19.: Sheet music for F3-SC3-R10-12.

Survey scores: [2.929, 3.095, 2.786].

F3-SC3-R10-12 had no sentiments.

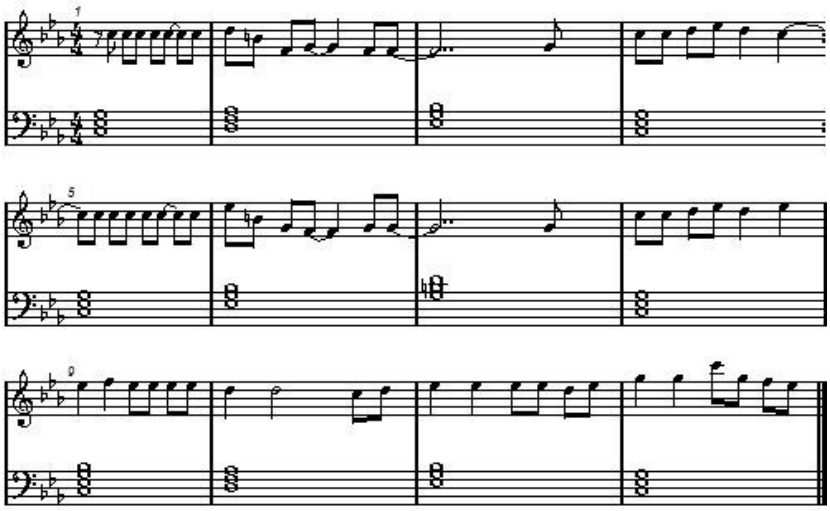


Figure A.20.: Sheet music for F3-SM3-R0-12.

Survey scores: [3.341, 3.500, 2.295].

F3-SM3-R0-12 sentiments:

- "Pokemon and Britney spears?"
- "The melody has a pretty big range of tones. And it has some nice dissonances!"
- "This is "Baby One More Time" by Britney Spears, with some notes off here and there."
- "Oooops! But of course a very good progression."
- "Sounds like it is a little bit off at 9 seconds."
- "Britney Spears", four times.

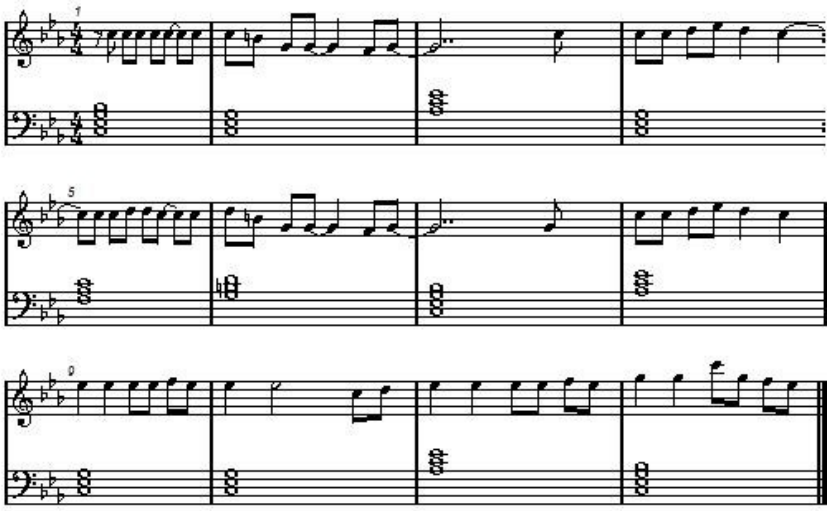


Figure A.21.: Sheet music for F3-SM3-R10-12.

Survey scores: [3.326, 3.605, 2.186].

F3-SM3-R10-12 sentiments:

- "Oh baby baby, in minor this time."
- "Britney with Baby One More Time, just a little better."
- "Britney Spears", ten times.