Automatic Accompaniment System for Western Music Using Finite State Machines

Jorge Humberto Acosta de la Pena

N 19796138

Submitted in partial fulfillment of the requirements for the

Master of Music in Music Technology

in the Department of Music and Performing Arts Professions

Steinhardt School

New York University

Advisors: [Juan Bello]

[DATE:2018/11/26]

TABLE OF CONTENTS

1. Abstract

Harmonizing a melody in the style of popular western music is a task that requires a working knowledge of music theory. Computer aided composition has been used in the past to apply these rules to imitate styles of composing by using set algorithms. Thanks to the development of speech processing techniques, it is now possible to train a probabilistic system with a corpus of data to achieve a plausible accompaniment for any input melody. Instead of mimicking harmonic rules, the accompaniment can be created using a probabilistic framework. This thesis seeks to create such a system by using Finite State Machines trained on a set of harmonic and melodic transcriptions taken from popular music. A Hidden Markov Model is created by using an ergodic single state transducer and a finite state automaton to model the emission and transition probabilities. Using algorithms available to finite state machines, the machines are composed into a single transducer that is populated with the probabilities taken from the training set. A melodic input is composed with this transducer and using the shortest path algorithm an output sequence is created. This sequence is then evaluated using three quantitative metrics. First each note-chord pairing is compared to the training set in order to establish an accuracy score. Then the output sequence itself is evaluated by calculating the distance of the sequence from the original training data by calculating the Kullback-Leibler Divergence between the reference training set and the output sequence. Finally, a harmonic dissonance metric is used to evaluate the overall dissonance between each note and the corresponding chord. Three experiments are performed using combinations of two different data sets. The results show a promising starting point for further development of the system by including smoothing techniques as well as a means of achieving some melodic context in the encoding of the finite state transducers.

Note: The code used in this thesis is available at https://github.com/JorgeAcostaNYU/thesis

2. Introduction

Historically, the development of music has been closely tied to technology, from the creation of the first instruments to the use of MIDI to represent harmonic events. In the western music tradition, religion has crafted strict rules that a piece should adhere to in order to be thought of as music (Grout, 1996). These sets of rules are not really intuitive, however accustomed our ears have become to them. Music is universal, but the creation of music is not as accessible a pursuit as it could be.

Most people can craft a melody out of thin air just by singing it. It is often the most memorable part of a musical piece because it is the most readily available to us, but it is only part of the equation. A melody by itself can be quite entrancing, but it is the harmony which provides context for an emotional impact (Laitz, 2009). Creating a worthy chord sequence for even the simplest melody requires an in depth understanding of music theory and understanding of concepts such as harmonic cadence and dissonance. Mastering even the most basic concepts of harmonization is a time-consuming endeavor not everyone is willing to take on, which stops many would be song-crafters from participating in the creation of music.

Since the 1950s, technology has been used to find a way in which an algorithm can 'fill the gap' between a melody and a musical composition complete with harmony and rhythm or even create entire compositions without human intervention: computers have been used since the mid 1950s to aid in the composition of music, from stochastic to rules-based systems (Chadabe, 1997).

While this is a field that is relatively new in music technology, vast advances have been made in a very short time. Many systems for automatic chord generation have been created with relative success. Computers allow us to explore many different approaches to the solution of the automatic accompaniment problem.

Furthermore, Music Information Retrieval is always evolving and new answers to this problem, are always being developed. Among the latest techniques pursued is the use of speech to text models such as finite state automata, which take in an input sequence of audio phonemes and output a visual representation in the form of words. By changing the input signal from phonemes to notes and the output sequence from words into chords, we can implement these kinds of models to the problem of harmonization. *It is the goal of this thesis to create an automatic accompaniment system using speech processing techniques to compute a viable chord sequence for a melodic input in the style of western music.*

The methods used in this thesis are taken from the field of Finite State Machines. A Finite State Transducer will be trained on melody/chord pairings found in a data set created from a variety of western pop and rock artists. This set contains a list of melodic and harmonic progressions, which will be used to calculate the probability of each note-chord pairing. This transducer will be composed with a Finite State Acceptor, not unlike the Grammar models used in speech to text processing technology. This FSA will encode the likelihood of a chord following another chord. Using composition and a shortest path algorithm a chord sequence will be assigned to a melodic input M.

In order to evaluate this system, we make the proposition based on the assumption that while there is no one single best chord progression for any given melody, it is possible to asses whether or not a chord sequence is suitable for a particular melody. A set of metrics will be proposed for assessing quantitatively how appropriate a chord sequence is for a melodic input. While there is no 'best chord sequence' for a given input, accuracy ratings, harmonic dissonance and divergence of the chord sequence might give some insight into how well the system is performing.

3. Literature Review

a. Music Theory

Music has always been a part of the human experience. From the beginning, animal bone flutes and Paleolithic paintings in caves, which might have served both as a canvas and as an auditorium for performance, are among the first pieces of evidence that sound has been honed by human beings into music since the beginning of civilization. The craft of music has evolved alongside the development of human society, both with the creation of instruments—such as the lyres of Mesopotamia—and in theory with the creation of the tonal scale closely related to mathematics, developed by Pythagoras (Grout & Palisca, 1996). Indeed, the coupling of math with the musical development of sacred music have come up with a fairly rigid set of rules that govern the creation of musical pieces. Concepts such as tonality, dissonance and cadence are still the basis for popular music in the western culture.

In order to understand the problem of creating an accompaniment for a melodic sequence there are some concepts we need to be familiar with. According to Laitz and Bartlette, music might be classified both in time and space. The time domain deals with accents in music, rhythms and metrical disturbances. The space realm deals in frequencies or pitches (Laitz & Bartlette, 2009).

Accompaniment is defined as two or more simultaneous pitches that serve to establish the space in which a melody line, a collection of pitches in time, might develop. Tonal theory deals with the relationships between harmony, melody and rhythm in the classical tradition that has been developed since the early 1400s in western society.

This field of tonal theory still impacts the creation of most popular music in our society and has permeated our lives so much that certain emotions can be evoked in most people by a few simple sequences of chords. Two concepts in particular govern the way we judge harmonic sequences: Laitz and Bartlett describe consonance and dissonance as the feeling evoked from hearing the distance between two simultaneous notes. These range from pleasant stability of perfect consonances (octave, perfect fifth and unison) to imperfect consonances (major and minor thirds and sixths) to dissonant intervals (second, augmented and perfect fourths and sevenths). The second concept is cadence. There are a number of chord sequences that we have come to expect to signal the end of a musical phrase. When we hear these kinds of progression, we experience a feeling of closure (Laitz & Bartlett, 2009).

b. Music Aided Composition

Technological advances have always impacted music composition. As computer programming defined a new era for manufacturing processes increasing speed, precision and analytical capabilities, they also enabled new methods for the creation of music. As soon as computers started being developed in the mid 1950s, they were used for musical purposes. According to Chadabe the first computer music piece was created at Bell labs in New Jersey in 1957 using a new kind of composition. Algorithmic composition is a system that can provide a complete musical idea or composition with minimum human interaction (Chadabe, 1997).

If music theory can provide a series of rules, then an algorithm can be created as a series of steps bound by those rules to provoke a certain outcome. Chadabe describes the earliest instances of computer-generated compositions by Lejaren Hiller and Leonard Isaacson (Illiac Suite), which employed a rules-based system, and Iannis Xennakis, who used stochastic compositions. These were probability-based systems in which millions of calculations of densities and random variables were calculated from inputs provided by the user.

Stochastic methods rely on an interplay between randomness and probability. This approach can be as simple as creating a random sequence of notes depending on the outcome of rolling dice. Starting in the 1950s and onward, the composer John Cage grew in prominence partly through the creation a number of important stochastic pieces. The rule-based approach is that in which the system goes through a series of steps, bound by the rules of tonal theory and weighted by the rigidness of each rule can be used to imitate compositional styles. Every time a rule is broken a penalty value is assigned. This system of steps takes into account the steps already taken and can 'backtrack' to find a new 'path' if the penalties accumulated surpass a certain threshold. Kemal Ebcioglu's used this in his work on an automatic 4-part choral generator based on over 350 rules to emulate J.S. Bach's style (Burns, 2014).

There is a third system for algorithmic composition that is based on artificial intelligence. It is an extension of the rule-based system, but with the added functionality of the system itself being able to make up its own rules; that is, to learn. This kind of system depends on genetic programming. Genetic programming is a kind of natural selection in an artificial setting, where certain functions and constants are used to describe ways in which the program can act on, or evolve, a piece of music. Some of those functions might be the manipulation of individual aspects of music, such as transposition, note generation, rhythm values, among many others. Once these functions are defined, the system creates a ‘population’ of randomly generated programs from the function set and are in turn measured in terms of ‘fitness’. Most of these programs will have a very low level of fitness but some might be slightly better, and these are selected and used as the basis for a new population until the overall fitness is good enough to pass a certain ‘critic’ or threshold. Depending on the level of fitness a program can be reproduced unchanged, by crossover (code swapping between parent programs), mutations, where the code is slightly changed, etc. (Alpern, 1995). It is important to note that a genetic system is dependent on the quantity and quality of the data set used to train it. Therefore, a high-quality dataset with enough information is necessary for the creation of systems of this sort.

The most successful approach so far is the use of dynamic probabilistic models which evolve over time. Artificial neural networks have enjoyed a lot of success over a wide variety of fields and deep generative models have been enjoying more success than even rules-based systems in generating harmony in the style of certain composers, such as J.S. Bach (Chen, 2018). Recurrent Neural Networks (RNNs) incorporate a hidden state, or internal memory that is quite applicable for temporal series such as those found in harmonic sequences. RNNs can describe long, complex temporal dependencies and systems that employ them have a higher success rate than previous systems based in more basic probabilistic frameworks (Boulanger-Lewandowski, 2013).

Dataset construction is one of the most important pieces of the puzzle in music information retrieval, specially dealing with stochastic systems that rely on a high quantity of quality data. The process of creating a database can be time consuming because it usually requires individual annotations by a qualified transcriber. Isophonics (source 16) features a number of datasets useful for systems that require annotated chords that match audio files and features annotation for The Beatles, Carole King, Queen among others. The Lakh MIDI Dataset v0.1 features a collection of 176,581 MIDI files. Over 45,000 of them have been matched to the Million Song Dataset for both systems that require symbolic data (MIDI) as well as audio matched content (Raffel, 2017)

In order to understand the work already realized in automatic accompaniment using genetic algorithms, an important concept to understand is the Markov Chain. It is the basis for the models that have gained most traction in the last 20 years.

c. Hidden Markov Models

Markov models operate on the premise that the past is independent of the future given the present. It is used for modeling some sort of temporal or sequential data, which makes it a good model for music systems. As described by to Forsyth and Bello, the Markov model assigns the value of a given variable (chord) depending on the value of the previous variable in the sequence. Thus, the probability of a given chord value is dependent on the previous value's probability in a first order Markov chain at a given time. Second or higher order Markov chains may be employed where the probability of event X_0 is dependent on past events at any given time, depending on the order of the algorithm.

$$P(x_t \mid x_{0:t-1})= P(x_t \mid x_{t-1-N:t-1})$$ (Forsyth, 2016).

Furthermore, a 'variable order' Markov chain uses a 'tree data structure' to handle the probability of events with no simple cycles. This is the process used in speech processing algorithms, where a 'Probabilistic Suffix Tree' models the distribution of words in a language by encoding each suffix in the PST (Ron, Singer & Tischby, 1994).

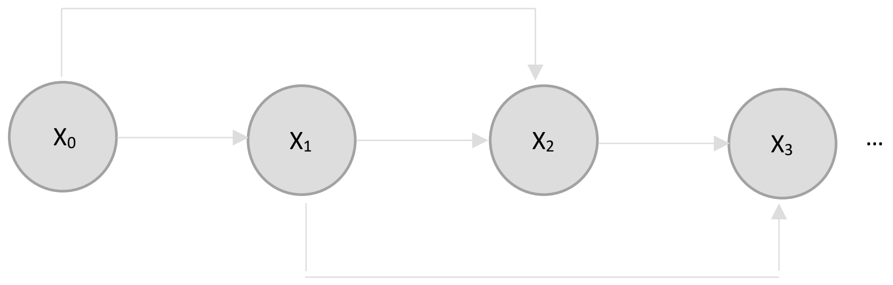


Figure 1 Unrolled Markov Chain. x_o influences x_1 and $x_{2.}$

d. Systems

There exists significant work on automatic accompaniment generation. Pardo and Birmingham present an automated accompaniment via score-following which uses Hidden Markov Models to break down a performance recorded on MIDI. By using structural information from the score, such as Codas or DC Al Capo information the system breaks down the score into sections via metric reduction. Six chordal qualities are used as the basis vocabulary (72 total chord templates) for the system each section is matched to the best probable chords depending on the matching notes of the melody and probable chords (Pardo & Birmingham, 2001).

MySong is another system that uses vocal melodies as inputs for an automated accompaniment system. It has been developed in conjunction with Microsoft and uses an HMM to match individual notes with certain chord types from a data set that usually match each individual note as well as the chords that precede and follow each chord in popular music. This model is trained on a popular music data set and has had very successful testing periods in which musicians validate each source of song (Simon, Boris, Sumit, 2008).

Chen's 'Emotional Accompaniment Generation System Based on Harmonic Progression' presents another take on this issue. This system incorporates valence and arousal curves as melodic input based on the premise that harmonic progressions determine the emotional impact of a piece of music. A library of accompaniment types is linked to the user's valence and arousal curves to determine the rhythm and number of notes played as part of the chordal structure. If the arousal is set to high there are more chordal changes and the quality of chord is selected via the valence curve (Chen, Lin & Chen, 2013).

It is important to note that these systems are all based on solving the problem of the 'best possible accompaniment' for a given set of notes. However emotional content can add the necessary randomness to an automatic accompaniment that can further the interaction with the user. Flowsson proposes an algorithmic compositional system that incorporates both rhythm and harmonic structure to a melody based on prosodic patterns that are recognized by listeners (Flowsson & Anders, 2012). This can add to the 'pleasing' aspect of an aided composition piece and in combination with the other techniques described in this review might be a step forward in the field.

f. Speech Processing Techniques

Jurafsky states that probabilistic or weighted data driven models have become the standard for speech processing techniques. In particular, the ability to combine several Finite State Machines allows us to tackle large problems by solving smaller ones and composing together a solution. An example of this is the problem of speech to text technology. This large problem is subdivided into a problem of translating phonemes to words, the probability of a phoneme given the previous and next phoneme (or triphone as is known in the field) and the grammar model that takes into account the context of the sentence uttered (Jurafsky, 2000). Functions available to Finite State Machines, such as composition, determinization and minimization have allowed us to create the systems that are in use today.

Forsyth's doctoral dissertation 'Automatic Musical Accompaniment Using Finite State Machines' (2016) is innovative in that he uses techniques used in speech recognition algorithm to construct a new model for chord generation. This model has been the basis for this thesis. By treating notes in a melodic sequence as symbols in one language and chords as symbols in another it is possible to use finite state machines to model a set of plausible 'conversions' (Forsyth & Bello, 2013).

According to Mohri, the Hidden Markov models in speech recognition techniques are a specific case of weighted Finite State Machines (Mohri, Pereira, Riley, 2002). Finite State

Machines (or Finite State Automata) model mathematically an abstract machine that has a predetermined number or states and can be in only one of these states at any time. They are often represented by a diagram composed of nodes and edges, and contain the following elements: one initial state, one final state and lines representing possible sequences between states.

When weights (or probabilities) are assigned to each transition a Weighted or Deterministic FSA is built (Forsyth, 2016). A Finite State Transducer can be quite similar to a DFSA with the added element of an output for each input. It can used to map out symbols from one alphabet to another (Mohri, 2009).

Finite State Transducers contain the following elements:

$$T = (\Sigma, \Delta, Q, \boldsymbol{E}, \imath, \boldsymbol{F}, \lambda, p)$$

where Σ refers to an input alphabet, Δ is the output alphabet, Q is the total number of states, $\boldsymbol{E}$ is a finite set of transitions, $\imath$ is an initial state belonging to Q, $\boldsymbol{F}$ is a set of final states belonging to Q, λ is the initial weight and p is the final weight function. Transitions between states are represented as

$$t = (p[t], l[t], w[t], n[t])$$

t being an element of $\boldsymbol{E}$, where p[t] is the source or the previous state, n[t] is the next state, w[t] is the weight of the transitions and l[t] is its associated label. For the purposes of speech processing as well as the design of this system, w[t] is a probability.

A path is a sequence of transitions between states. In order for a path to be labeled as successful it must represent a transition from $\imath$ to $\boldsymbol{F}$ and is labeled as $\boldsymbol{\pi}$. The weight of $\boldsymbol{\pi}$ is the product of the initial weight, the weights of the transitions and that of the final weight.

$$w[\boldsymbol{\pi}] = \lambda \otimes w[t1] \otimes \cdots w[tn] \otimes \rho(n[\ tn\ ])$$

with n being the number of transitions (Mohri, 2002).

Forsyth makes use of several characteristics of FST to make this model useful for accompaniment generation. One operation that is very useful is composition, which according to "Weighted Automata Algorithms" allows for the combination of several weighted transducers into a larger model. In order to use this operation, the input alphabet of one transducer must coincide with the output alphabet of the other one:

$$T1 = \Sigma*, \Delta*, Q1, I1, F1, E1, \lambda1, \rho1$$

$$T2 = \Delta*, \Sigma*, Q1, I1, F1, E1, \lambda1, \rho1 \text{ [22]}$$

thus, allowing for solving a greater problem by solving a series of smaller ones (Mohri, 2009).

Forsyth makes use of a Finite State Automata to model chord sequences and a single state FST to map melody notes at a time t to a chord at time t. Finite State Machines must include a finite set of symbols, or alphabet (chords and melody). This is calculated in two ways: binary templates representing each of the chords that will make up the library are used for matching labels to chords. The second way is to use an online vector quantization algorithm that iterates a set number of times to assign symbols from a codebook to each instance. The melodic inputs in this system are then quantized to create an input alphabet. (Forsyth & Bello 2013).

The next step is the creation of the chord sequence. If M is the transition probabilities from chord to chord, and T is the emission probability for each note that has a chord a Hidden Markov Model may be created, with each node being a single note in a melody paired via an FST to a probable chord. A sequence FST is used to improve this system, as it pairs chords with melody in context. The Viterbi Algorithm can then be used to calculate the fastest or easiest route to navigate a Hidden Markov Model and can provide the best possible match from a set of possible chord sequences. This is done by calculating the most probable path through the melodic input sequence and will output the best possible chord accompaniment for the specific input melody (Forney, 1973).

g. Metrics

One of the most important pieces of any model is the metric used to evaluate its performance. Simon and Morris evaluate their system Mysong with subjective evaluations. The automatic chord sequence is subjectively rated against a 'Chord Audition Tool' where the participants were presented with several options of chords for their melody. Four accompaniments were created, two with Mysong and two with the Chord Audition Tool and the results were evaluated by trained musicians (Summit et AL, 2008).

While a chord sequence might be considered as a difficult thing to rate regarding it's 'correctness', quantitative measures do exist to determine both the performance on a chord-note pairing, as well as for the overall sequence. Accuracy was measured in Forsyth's system as a number of correct chord symbols over the total number of chords generated given a strict system of rules for accompaniment. Per Symbol Distortion is a measurement that can also be used to take into account that some chords can more dissonant than others. This measurement can be done by measuring the Euclidean distance between the predicted symbol and that of the testing midi sequence (Forsyth & Bello 2013) or by creating a map of related chord notes that takes into account notes shared between chords, such as C and Am (Bertin-Mahieux, 2010).

Dissonance is another measurement that may be applied to individual notes. According to music theory, different intervals are more or less dissonant than others. Forsyth measures each notes dissonance against a predicted chord by including it as part of the chord tone and measuring the weights assigned to each interval with the root of the chord. Thus, if an A chord is paired with A#, the dissonance measurement would be quite high (Forsyth, 2015).

In order to evaluate the chord sequence, the PRI or Predictive Information Rate can be used. It is an attempted measurement of how well a symbol in a sequence represents the future given the symbols that come before it. Forsyth describes it as an "…information-theoretic quantitative metric that attempts to describe the degree to which a sequence achieves a proper balance between determinism and randomness" (Forsyth, 2016, p 68) and can be computed by the following formula

$$p = H(A^2) - H(A)$$

In this formula, A is a two-dimensional transition matrix ($A = [a_{ij}]$). The rate of entropy H(A) is calculated by taking the log difference between a stationary reference distribution π. The stationary distribution of A is calculated by finding the left eigenvectors with an eigenvalue of 1. Once this is done, we can compute the entropy rate by calculating the log difference between a and A^2. The predictive information rate can then be calculated using the equation above (Forsyth, 2016).

In their 2008 paper 'Evaluating and Visualizing Effectiveness of Style Emulation in Musical Accompaniment' Chuan and Chew propose a set of six metrics as a means of assessing accompaniment generated by their system. Three of these metrics deal with percentage of accuracies. Correct rate refers to melody notes that are found within the chord tones (Chuan and Chew, 2007).

The same chords metric refers to a comparison between the actual chord sequence and those used in training. They propose a third percentage metric that measures all chords parallel, subdominant or dominant to the original chord identity. If the original chord is C, and Am or even G chord would count as a hit.

The first of the distance metrics focus on the neo Riemannian distance between a generated chord and its ground truth. The average minimum is counted, with a maximum possible distance between any two chords of 7. The final two metrics deal with the distribution of the chords. The first measures the Half Step distance (mean squared difference between the interval distributions of two accompaniments. The second is the weighted version of the HS distance. This metric 'punishes' half step intervals between the sequences, as they are the most dissonant possible differences (Chuan, Chew, 2008).

4. Methodology

The following chapter describes the approach to the problem stated in this thesis, regarding the use of speech processing techniques, such as Finite State Machines to create a system that can be trained using symbolic data. We outline the approach, step by step in the following section. We also include information on the composition algorithms used to combine the finite state machines into a transducer that can be decoded into a chord sequence. A description of the data set used for training and testing of the system is also included.

a. F - Creating the transducer (Note-Chord Pairings)

As explained in the literature review, a weighted Finite State Transducer has the following elements:

Σ	Finite input alphabet
Q	Finite set of states
I	Set of initial states
F	Set of final states
Δ	Finite output alphabet
p	Ergodic weight function

A single state Finite State Transducer **F** can be used to encompass every single note-chord pairing that occurs in the training data. Each arch of the transducer contains the following information:

Example from code:

```
arc0I = fst.Arc(1,1,(we[0]),0)

f.add_arc(0,arc0I)
```

From state = 0

Input symbol = 1 (I degree)

Output symbol = 1 (I major)

Weight = Weight vector corresponding to this particular arch (note-chord pairing).

Next state = 0

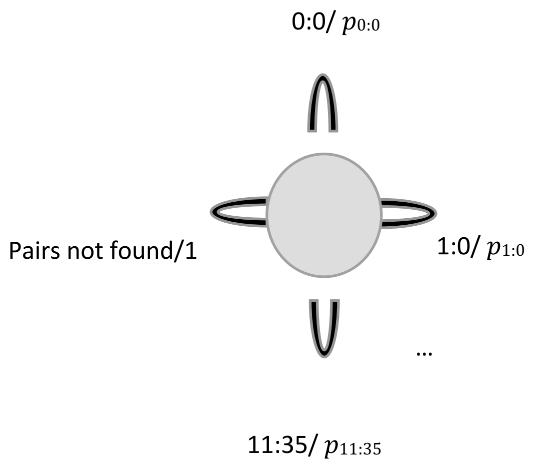


Figure 1: Single state FST F. Every arch begins and ends in the single state and represents mapping between each of the 12 input symbols (notes) and the 36 output symbols (chords) with the weight being a function dependent on the co-occurrence of the pairings found in the training set.

In this system, F is an ergodic finite state transducer that maps each of the 12 pitch classes to one of 36 possible chords. Each of the arches in the FST is one of the possible mappings, and the weight associated with each arch depends on the number of times each pairing is found in the training set.

Thus, we have for **F:**

Σ	0-11 pitch classes
Q	1 State (Initial and Final)
Δ	0-36 chords (12 major, 12 minor, 12 diminished, 1 chord not on dictionary)
p	Weight function representing co-occurrence of pairing in training.

Since we are using the shortest path algorithm to find the most likely chord sequence, each possible note-chord pairing is assigned an initial weight of p = 1. Thus:

If pair not found: weight = 1

If pair is found n times: weight = 1 / n

Every time a note-chord pairing occurs, weight is added thus decreasing the entire weight of the chord pairing. The more frequent a note-chord pairing is, the closer its arch weight will be to 0.

b. G - Creating the Bigram (Conditional Probability of C_t given C_{t-1})

As applied in Forsyth's dissertation (Forsyth, 2016) the FST **F** can be composed with a finite state acceptor that will model the conditional probability of a chord being Ct given the history of the previous symbols:

$$P(c_t \mid c_0^{t-1})$$

Where c_0^{t-1} is the sequence of symbols, or chords, from time 0 to time t-1.

If we look at this probability as an N-Gram, we can use the equation used by Forsyth 'Generating Musical Accompaniment Using Finite State Transducers' (Forsyth, Bello, 2013) to approximate the probability by counting the amount of times a certain sequence is found. According to 'Probabilistic Finite State Machines' "the n-gram approximation makes the assumption that the probability of a symbol depends only on the n-1 previous symbol" (Vidal et Al,2005). In this system we have implemented a first-degree N-gram which takes into account the previous chord:

$$P(a_t \mid a_0^{t-1}) \approx P(a_t \mid a_{t-N-1}^{t-1})$$

The second part of this equation can be computed as the number of times a certain sequence is found in the training set (Forsyth, 2016). If we have a symbol sequence $a_{t-N+1}a_{t-N+2} \dots a_{t-1}a_t$ that occurs C number of times, then we have

$$P(a_t \mid a_0^{t-1}) \approx \frac{C(a_{t-N-1}^{t-1} at)}{C(a_{t-N-1}^{t-1})}$$

This N-Gram was encoded into a Finite State Automata.

Thus, we have for **G**:

Σ	0:36 classes
Q	37 states
I	1 initial state
F	36 final states
Δ	is the same as Σ
p	Weight function dependent on co-occurrence of sequence in training.

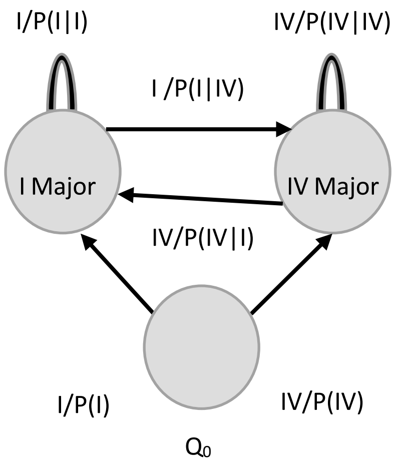


Figure 2: Example of bigram that represents two classes: I major and IV major. Initial state is Q_0 and both I major and IV major are final states. The FSA is initialized with a predetermined weight p = 1.

The bigram implemented in this system encodes the probability a chord given the previous chord, for example the probability of the sequence ImajImaj, ImajI#maj, ImajIImaj and so on.

Code example:

```
arcinitial1 = fst.Arc(1,1,big[0],1)
g.add_arc(0,arcinitial1)
```

From state = 0

Input symbol = 1 (I Major)

Output symbol = 1 (I Major)

Weight = Weight vector corresponding to this particular arch (bigram sequence initial state to I major state or State1).

Next state = 1

In this example we map the initial state to the first state that in this case represents the Chord State for I major. This state will be mapped to all other 35 states as well as to itself with the initial weight of 1 minus the weight function, which depends on the co-occurrence of the sequence in the training data set.

It must be noted that the input and output symbol sets are the same, thus making this FST a Finite State Automata that can in turn has the input alphabet corresponding to the output alphabet of the FST **F**.

c. T - Creating the Transducer

Once we have both **F** and **G,** by using the composition algorithm we can combine them into a single transducer that will can be conceptualized as a Hidden Markov Model. It will encode the following probability:

For a sequence of symbols A_0A_1… and a set of output symbols C_1C_2…:

G models $P(A_t \mid A_{t-1})$ and **F** models $P(C_t \mid A_t)$

Then by forming **G** o **F** the resulting transducer **T** will model the probability:

$$P(A_t \mid A_{t-1}) \, . \, P(C_t \mid A_t)$$

where **G** models the transition probabilities and **F** models the emission probabilities, thus creating a Hidden Markov Model:

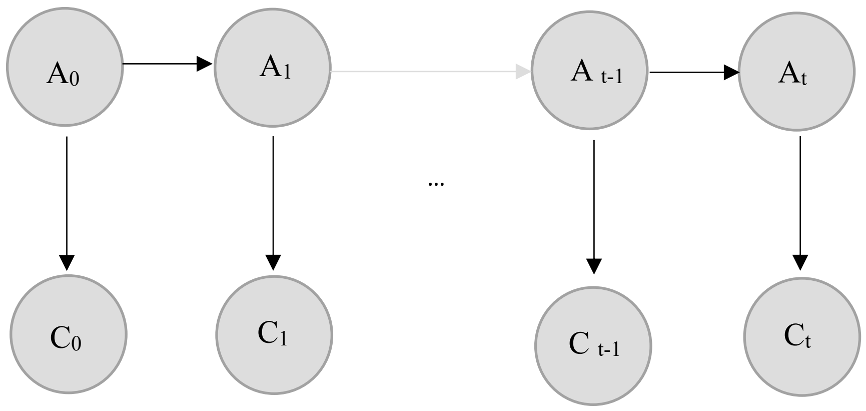


Figure 3. This is a Hidden Markov Model that represents the hidden states which in this case will be the probability of a chord dependent on the previous sequence of chords.

In order to use the composition algorithm, the input labels of **G** must be sorted to match the output labels of **F**, both of which represent the dictionary composed of 36 chords.

d. Computing and Decoding the C Transducer

Once we have the **T** transducer it can be used to compute a most likely path through it given a melodic sequence M. We can turn a series of melodic sequences into a linear chain Finite State Acceptor that consist of as many states as there are symbols minus one, and whose edges consists of the melodic symbols. It is an acceptor because only the original sequence of melodic symbols can be used to navigate this linear chain FSA.

The resulting FSA **M** can also be composed with the FST **T**, as the input/output symbols correspond to the input alphabet of **T.** The resulting FST is thus labeled **C** and contains all the possible paths that **M** can take through **T** as shown in 'Building a Harmonic Ecosystem' (Musick, Forsyth, Bittner, 2015)**.**

The shortest path algorithm finds the quickest, or least 'expensive' way to get through the Finite State Transducer (the path with the least amount weight). Since we have used a probabilistic means of assigning weights, the shortest path algorithm will yield a sequence of symbols that contain the most likely chord sequence for the input melodic sequence M, while the longest path will be mostly composed of non-occurring note-chord pairings in the training set that were assigned the arbitrary weight for p.

$$\mathrm{sequence} \ = path\ \mathrm{P(c|m)} = path\ \mathrm{P(m|c) \cdot P(c)}$$

where c is a member of Σ (dictionary of possible chords).

e. Dataset

This system was trained using symbolic data obtained from the Rock Corpus. The data set used, RS-200, consists of two hundred melodic transcriptions corresponding chord transcriptions done by two transcribers: Trevor de Clercq and David Temperley for a total of 400 harmonic analyses (de Clercq, 2011). These sets will be referred by the initials of the annotators: TDC and DT

The high quality of the transcriptions as well as the abundance of chord-note pairings make this dataset very suitable for the purposes of this thesis. It contains popular songs in the western tradition according to a Rolling Stone magazine article titles "500 Greatest Songs of All Time" (De Clercq, Temperley, 2011). It is also the dataset that Jon Forsyth used in the dissertation that serves as the inspiration for this thesis (Forsyth, 2016).

The system looks at a '.nlt' and a '.clt' file (melodic and harmonic transcription files respectively) and using the timing information encoded in the files assigns chords to each melodic event. This will in turn increment the weight for that particular chord-note pairing by 1 during the training stage. The end result is a sequence of chords of equal length to the sequence of notes.

It was necessary to adjust the information in the data set for the training of the machines. The data set provides a lot more information than was needed, so decided the information extracted relates only to scale degrees, therefore effectively transposing all songs into a single key. Transposed into the key of C, so every melodic line is represented symbolically in pitch class degrees limited to 1 octave. While the Rock Corpus provides a lot more melodic information, such as timing, MIDI notes and octave, the concern of this thesis is only with twelve notes mapped to 12 possible major, minor and diminished chords.

Every chord is relative to the scale, so we have 36 possible chords used in training (12 majors, 12 minors, 12 diminished) and a separate class for every chord not used (augmented, power chords, etc.) According to the way the dataset is constructed, there are several chords that for our purposes can be used as equivalent symbols, such as Eb major and D# major.

Furthermore, chords that feature 7ths and other embellishments such as 9ths, dominants as well as modified roots can be of use. The chord symbols used in this thesis are focused on the root, third and fifth of a chord. Below we have illustrated the chord equivalencies that have been made to the dataset:

Major

I Major	'I' or 'I6' or 'I64' or 'I7' or 'Id9' or 'Id7' or 'V/IV' or 'V42/IV' or 'V43/IV' or 'V65/IV' or 'V7/IV'
I# Major	'I#' or 'I#9' or 'bII' or 'bII7'
II Major	'II' or 'II6' or 'II65' or 'II7' or 'IId7' or 'IV/vi' or 'V6/V' or 'V6/v' or 'V65/V' or 'V7/V' or 'V/V'
bIII Major	'bIII' or 'V/bVI' or 'bIII6' or 'bIII64' or 'bIII7' or 'bIIId7' or 'bVId7/V')
III Major	'III' or 'III64' or 'V/vi' or 'V6/vi' or 'V65/vi' or 'V7/vi'
IV Major	'IV' or 'IV6' or 'IV64' or 'IV65' or 'IV7' or 'IV9' or 'IVd43' or 'IVd7' or 'V/VII'
IV# Major	'#IV' or 'bV' or 'bVb5'

V Major	'V' or 'V42' or 'V43' or 'V6' or 'V64' or 'V65' or 'V7' or 'V7b9' or 'V9' or 'V+11'
bVI Major	'bVI' or 'bVI6' or 'bVI7' or 'bVIb5' or 'bVId7'
VI Major	'VI' or 'V/ii' or 'V7/ii' or 'V7/II' or 'V7/II' or 'VI7' or 'VId9'
bVII Major	'bVII' or 'V/bIII' or 'V7/bIII' or 'bVII6' or 'bVII64' or 'bVIId7' or 'bVId7/ii' or 'bVI64/ii')
VII Major	'VII' or 'V7/III' or 'V/iii' or 'IV/IV' or 'IV64/IV' or 'V7/iii' or 'V7/III'

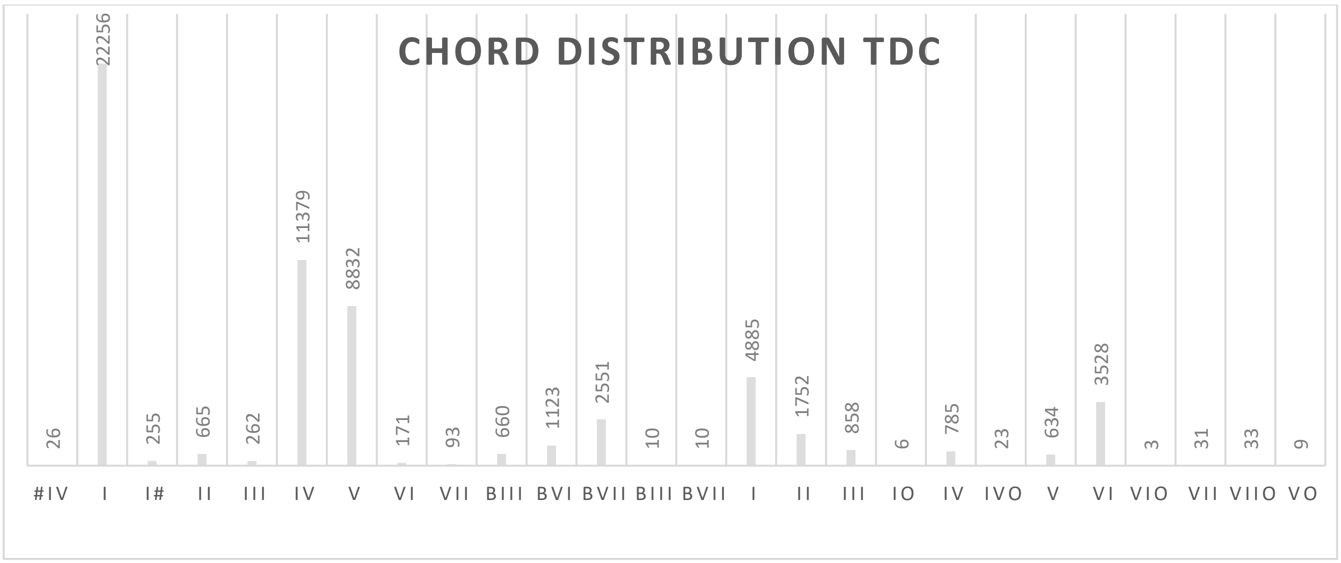
Minor

i minor	‘i' or ‘i42' or 'i6' or 'i7' or 'i9' or 'ii7/bVII'
ii minor	‘ii' or 'ii6' or 'ii65' or 'ii7' or 'ii9' or 'iih42' or ‘iih43'
biii minor	‘biii' or 'biii7'
iii minor	‘iii' or 'iii6' or 'iii64' or 'iii7'
iv minor	‘iv' or 'iv64' or 'iv65' or 'iv7' or 'v11'
v minor	‘v' or 'iv/ii' or 'iv6/ii' or 'ii/IV' or 'v6' or 'v64' or 'v65' or 'v7'
vi minor	‘vi' or 'ii/V' or 'vi6' or 'vi64' or 'vi7' or 'vi9'
bvii minor	‘bvii' or 'bvii7' or 'ii7/bVI'
vii minor	‘vii' or 'ii7/vi' or 'vii64' or 'iii6/V'

Diminished

io	'io' or 'viio/ii' or 'viix7/ii'
iio	'iio' or 'iih65' or 'iih7' or 'iio7'
iiio	'iiio' or 'iih65/ii' or 'iih43/ii'
ivo	'ivo' or 'viih7/V' or 'viio/V' or 'viix42/V' or 'viix43/V' or 'viix7/V'
vo	'vo' or 'viix7/vi'
vio	'vio' or 'vih7' or 'iih7/V'
viio	'viio' or 'iih7/vi' or 'viix42' or 'viih7'

It is worth noting that during the evaluation stage, the same chord correspondences have been made for assessing accuracy and note/chord plausibility. Several of the songs have key changes within the song. For this system, we have not taken into account the harmonic context extended to the chord sequences and are only concerned with individual chord-note pairings. We have transposed every chord to conform to a degree symbol and every note to a number between 0 and 11. Therefore, these changes do not affect the amount of training data.



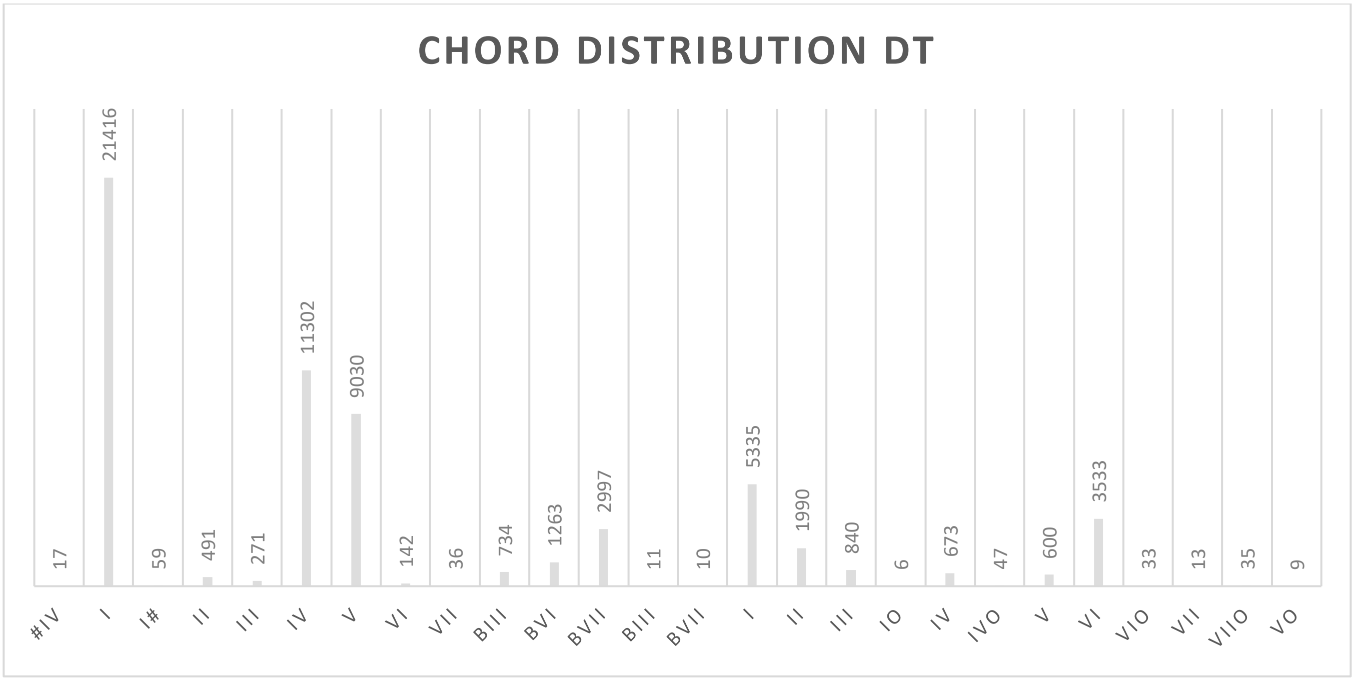


Figure 4a and 4b. Chord distribution of the training set. All chords have been conformed to scale degrees with the chord equivalencies applied to the training set.

Pywrap Fst, available at (http://www.openfst.org/twiki/bin/view/FST/PythonExtension) was used for the implementation of the system. This is a wrapper for the OpenFst library which can be used for the creation, combination and optimization of weighted finite state transducers. OpenFst supports both the compose feature as well as the k-shortest path algorithm used in this thesis. This is an open library available at http://www.openfst.org/.

While this system is based on the work outlined in Forsyth's dissertation (Forsyth, 2016), we have taken a different approach in evaluation, which will lead us into new territory as we improve the system. Focusing on the dataset itself, we have explored 3 different training paradigms for the FST: the first one consists of training the system with two transcriptions per song, the second and third trainings were done on the TDC set and the DT set. Evaluation was done on all three combinations for further analysis.

Experiments and Results

All experiments done with this system were carried out with the same architecture:

1. Data is extracted from training data
2. **F** arch weights are populated according to note-chord pairs
3. **G** arch weights are populated according to two chord sequences
4. **G** input labels are sorted
5. **F** and **G** are composed into a new transducer **T**
6. Test melodic sequence is extracted from the same data set and **M** linear FSA is constructed.
7. **T** input labels are sorted
8. **C** is created from the composition of **M** and **T**
9. Sequence is extracted from **C** using shortest path algorithm

Thus, the input of this system consists of a linear FSA that represents a melodic sequence and the output is a viable chord sequence that contains as many output symbols (chords) as there are input symbols (notes).

The system was trained and evaluated according to the metrics outlined below. 4 fold validation was used to test the system: 150 songs were used to train the system and 50 were used to validate. The process was repeated 3 times, rotating the training and testing sets so each song was used at least once as a test melody

The sequence is then evaluated according to the following metrics:

a. Evaluation Method

To reiterate, the aim of this thesis is to create an automatic accompaniment system that outputs an accompaniment suitable for the western pop musical style. Therefore, it is necessary first to evaluate the number of chords that are suited for a specific melody, and then a comparison of accompaniment styles between real life examples and the system's output.

Accuracy: The first evaluation metric will be a direct comparison between the sequence the system generates, and the real sequence used for training.

$$Accuracy = \frac{Number\ of\ hits}{Number\ of\ note - chord\ pairings} \cdot 100$$

Divergence (Kullback-Leibler): a measurement of the distance between two distributions: P and the reference distribution Q. Forsyth used this measurement by creating two 2-dimensional transition matrices (P and Q), trained from the rock corpus sequence and the output sequence respectively (Forsyth, 2016).

$$D_{KL} = \sum_{i,j} P_{ij} \cdot log \frac{p_{ij}}{q_{ij}}$$

Dissonance: A measurement first proposed in (Chuan, 2011) that consists of assigning a dissonance value to each interval found within a chord. The dissonance values are as follows:

Interval	**Dissonance**
Unison	0
Minor second	90
Second	30
Minor third	15
Major third	12
Fourth	9
Augmented fourth	50

Every other interval found is inverted to conform to these values.

In this thesis we have the following dissonance values for each chord type:

Mayor	(major third, fourth, minor third)	36
Minor	(minor third, fourth, mayor third)	36
Diminished	(minor third, minor third, augmented fourth)	80

Once the interval for the chord is calculated the intervals caused by including the melody note as part of the chord (C+m) is calculated and summed to the total. SOURCE calculates the mean and standard deviation of the dissonance frame by frame. These measurements are then normalized with the mean and standard deviation of the chord sequence by itself to account for inherent dissonance found in the chord sequence.

$$mean_{C+m} = \frac{mean_{C+m}}{mean_C}$$

$$sd_{C+m} = \frac{sd_{C+m}}{sd_C}$$

The rock corpus dataset is comprised of 2 harmonic analysis per song created by the two authors. There are some discrepancies that arise which can have a negative impact on the accuracy of the system. According the ‘A Corpus Analysis of Rock Harmony’ (de Clercq and

Temperley, 2011) the harmonic analyses are in agreement 92.4% of the time according to the relative root of the song. This discrepancy can be seen in figure 5.

Three experiments were run with the system. The difference lies on which data set was used to train the FST **F**.

The first experiment run was done with both harmonic analyses used to train the transducer. It was evaluated against the first and the second set separately. The results in both accuracy and harmonic dissonance were low enough to merit a redesign of the system, which up to this point consisted of both sets used to train the transducer.

The second experiment run was done by using only the first set of transcriptions (Trevor de Clercq). It was also evaluated against the first and second sets.

The third experiment run was done by using only the second set of harmonic transcriptions, which feature a lot more chord variations. (David Temperley).

Notes (degrees)	7	7	4	4	4	4	5	5	5	2	2	2	2	2	0	4	7	7	7	0
Training TDC set	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I
Training DT set	I	bVII	I	bVII	I	bVII	I	bVII	I	bVII	I	bVII	I	bVII	I	bVII	I	bVII	I	bVII

Figure 5. These are the first 20 chords used for training the **F** transducer. This shows some of the discrepancies found between the two training sets that affect the note-chord pairings.

The results are as follows:

b. Experiment 1:

Accuracy	Both Sets	TDC set	DT set
FST Trained on both sets	12.78%	12.92%	12.63%

Kullback-Leibler Divergence

Pij	Qij on TDC set	Qij on DT set
T trained on both	1.4674	1.3758

Dissonance Mean and Standard Deviation

Dissonance	Mean	Standard Deviation
Training TDC set	2.281759	1.018
Training DT set	2.292622	1.014
FST Combined	3.305173	1.4676

c. Experiment 2:

Accuracy

	Against harmonic analysis 1	Against harmonic analysis 2
FST Trained on TDC set	20.664%	16.291%

Kullback-Leibler Divergence

Pij	Qij on TDC set	Qij on DT set
FST Trained on TDC set	1.21099	1.2698

Dissonance Mean and Standard Deviation

Dissonance	Mean	Standard Deviation
Training TDC set	2.281759	1.018
Training DT set	2.292622	1.014
FST Trained on TDC set	2.859885	1.3958

d. Experiment 3

Accuracy

	Against harmonic analysis 1	Against harmonic analysis 2
FST Trained on DT set	15.326%	18.192%

Kullback-Leibler Divergence

Pij	Qij on TDC set	Qij on DT set
T Trained on DT set	1.40178	1.38867

Dissonance Mean and Standard Deviation

Dissonance	Mean	Standard Deviation
Training TDC set	2.281759	1.018
Training DT set	2.292622	1.014
FST Trained on DT set	3.341586	1.547448

5. Discussion

Baselines:

In order to put these results into context, we have included some baselines with which to compare our system:

1. **Random sequence**

   A random generation of the sequence provided an accuracy score of 3.37 % when compared to TDC's set and a score of 3.16 % when compared to the DT set of harmonic analyses.

2. **Most popular chord**

   By generating a chord sequence that consists only of the most popular chord, which in this case is the I major, we get an accuracy score of 39.32 % for the TDC set and a score of 35.19 % for the DT set.

   The first of these baseline comparisons argues in favor of the system, as out of the 36 possible chord symbols, the chord sequence generated by the output are a lot closer to the real sequences. This is not the best guideline, because even with a set this big, chord sequences tend to favor a very few chords and this system awards equal weight to every chord.

The most popular chord baseline puts the results from the system into a better context. Using a sequence comprised of only the most popular symbol performs almost twice as well as our system. From the results of the experiment and the baseline comparisons we can make the following observations of our system:

a. Accuracy:

The first experiment introduced a lot of noise into the training of the transducer, as some chord sequences and especially chord note pairings were contradicted between sets 1 and DT set

The second experiment, which consisted of training the transducer with set number 1 was the most successful accuracy wise. The accuracy was lowered when compared to DT set as a consequence of the discrepancy between the annotators.

The third experiment which consisted of testing a transducer trained on the second set of harmonic analysis still performed better than the transducer on the first experiment. It was less successful than the transducer of the second experiment, however it still outperformed everyone when tested against the second set of transcriptions.

We can draw the conclusion that accuracy is impacted considerably by the training and testing sets used. The best result came from a transducer trained on TDC set and tested against chord sequences present in TDC set. The worst result came from a combined training FST when compared against the second set of transcriptions. The difference between the worst and best is approximately of 8 percentage points:

Accuracy	Both Sets	TDC set	DT set
Both Sets	12.78%	12.92%	**12.63%**
FST Trained on DT set	-	15.326%	18.192%
FST Trained on TDC set	-	**20.664%**	16.291%

b. Kullback-Leibler Divergence:

As stated before, the KL Divergence measurement can be used to gage how different two sequences of symbols are from each other. The sequences used for reference came from each of the training sets.

The best and worst scores mimic the results from the accuracy evaluation:

The least divergent transition matrix featured matrix trained on the sequence created by the finite state transducer trained on the first set of harmonic transcriptions: 1.21099
The most divergent matrix featured a transition matrix trained on both harmonic transcriptions when compared to the first set: 1.4674

It should be noted that the divergence scores do not vary greatly between one set of transcriptions and the other. This metric is a lot less harsh, as it takes into account changes in the sequence vs actual chord symbols.

c. Harmonic Dissonance:

When speaking of the evaluation of an automatic harmonic accompaniment, the question of dissonance is of importance. This measurement comports to the harmonic rules that govern western music. Most popular music favor note-chord pairings that have a low dissonance measurement (Mesz et Al, 2011). This can be seen in the prevalence of power chords, which comprise of the lowest possible harmonic dissonance measurement:

Interval	Dissonance
Root	0
Fifth	9
Octave	0

The dissonance measurements of this system were computed as a means and a standard deviation of a frame-wise measurement, as was done in Forsyth's work (Forsyth, 2016).

While one may expect the harmonic dissonance measurement of the second harmonic set, due to the use of seventh chords and other embellishments used in the analysis, the measurement was performed on the chord equivalencies. As such we do not see much difference between the two sets:

Dissonance	Mean	Standard Deviation
Training TDC set	2.281759	1.018
Training DT set	2.292622	1.014

This metric reflects the shortcomings of the system more clearly. All harmonic dissonance means are considerably larger than those found in the training sets. The least dissonance sequence was obtained by the training done with the first set of transcriptions:

FST Trained on TDC set	2.859885	1.3958

The most dissonant sequence was a product of the transducer that was trained on the second set of harmonic transcriptions:

FST Trained on DT set	3.341586	1.547448

This indicates that the second set of harmonic transcriptions yield a significantly more dissonant sequence of chords when combined with the input melodic sequence.

d. Persisting problems

Looking at the evaluations of the system it is clear that there is a lot of room for improvement. Although there is an accuracy ceiling of 92.4%, which comes from discrepancies between the two sets of harmonic analyses, an accuracy of 20.644% is far from optimal.

While the KL Divergence shows that the sequences are viable, and not infinitely separate from the reference sequence, the harmonic dissonance score shows that the result is far more dissonant than the original training set.

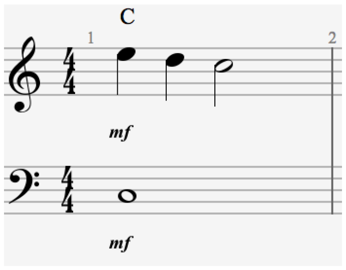
Following is an excerpt of the sequence that achieved the highest accuracy compared to the training set used in the FST. We have looked at two problem areas that can offer a path towards improving all three scores.

Notes relative to scale	G	G	E	E	E	E	F	F	F	D	D	D
Notes relative to scale	7	7	4	4	4	4	5	5	5	2	2	2
Sequence	I	I	I	I	I	I	I	VI	VI	V	V	III
Dissonance	**1.667**	**1.667**	**1.75**	**1.75**	**1.75**	**1.75**	**4.583**	**4.166**	**4.166**	**1.667**	**1.667**	**3.638**
Chords	I	I	I	I	I	I	I	I	I	I	I	I
Dissonance	**1.667**	**1.667**	**1.75**	**1.75**	**1.75**	**1.75**	**4.583**	**4.583**	**4.583**	**2.916**	**2.916**	**2.916**

Figure 6. This is an excerpt of the first 12 notes in the evaluation of the system. This particular sequence is a product of an FST trained on the first set of harmonic transcriptions and it is being evaluated against the same training set.

Problem 1: FST does not take into account the melodic context when encoding the F transducer

The output sequence generated by this system contains a lot of chords that do not fit into the traditional harmonization of the major scale the melodic input sequence belongs to.



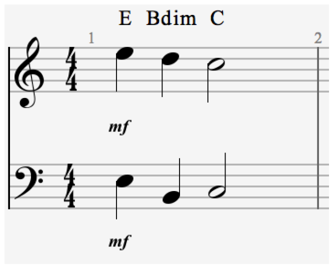


Figure 7a and 7b. 7a is an example of a usual harmonization of a passing tone. 7b is the harmonization this system has come up with.

In figure 7a we can see that while the note B clashes with the C note, because it is on a weak beat the C chord is appropriate for the whole measure. In contrast on figure 7b, this system has assigned a chord per measure and does not take into account any sort of melodic context. The passing tone B is treated with the same importance as the C note. These kinds of melodic patterns and embellishments that should not change the harmonic analysis of the piece are all treated with the same weight of importance because this system does not address melodic context. Every single neighbor, passing tone and even trill generates a new chord which will not belong to the scale, thereby adding more dissonance to the sequence.

Furthermore, by excluding melodic context the system assigns the E major chord to the first note, as it is an e. This chord is not part of the major scale, as it includes a sharp fifth scale degree. If the three notes in the bar were evaluated as a whole, the system could be made to recognize that either an A minor or a C chord were the most probable options for this particular melodic sequence.

Problem 2. Chord sequence is not stable

One of the most easily perceived shortcomings of this system is the frequency that chords change in the output sequences. This is a product of the **F** transducer, that only takes into consideration a single note for mapping to all possible chords. Every note input into the system generates a probabilistic output without regard to melodic context. This results in a lot of instability in the chord sequence, which can be seen in figure 7b for example.

These two problems can further be explored through the analysis of a sequence that might be the product of a popular western song that did not form part of the training set. For this purpose, the following transcription of a simple harmonization for 'Ode to Joy' is included:

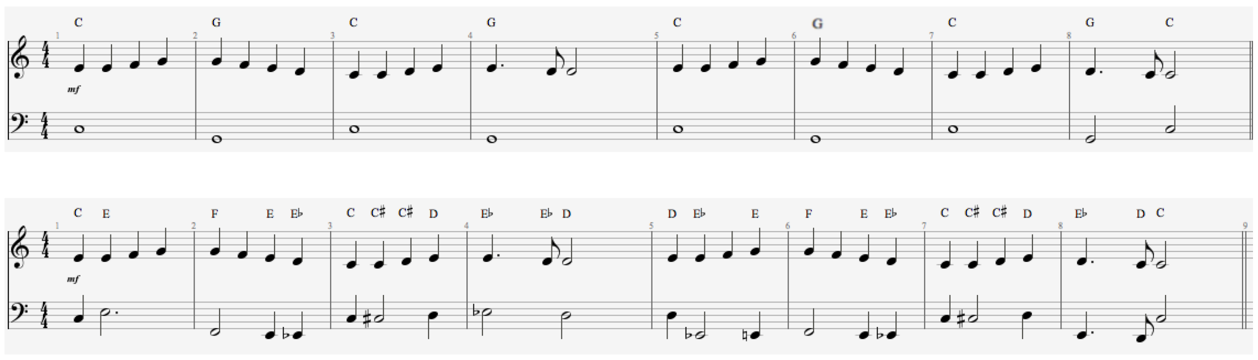


Figure 7a and 7b. 7a features a simple harmonization of the first eight bars to 'Ode to Joy'. 7b is harmonization done by the system trained on the first set of harmonic analyses.

As can be seen in figure 7b, there are a number of chords that are a lot more dissonant than should be desired. The passage starts with a plausible chord for an E note, which is in fact a C Major, however it quickly becomes a chord not available in the major scale: E Major. Dissonance becomes an issue specially in measures 3 and 7. Both are assigned an acceptable chord at the beginning and transition into dissonant chords with the next note. This type of problem can be solved by making it more likely that a chord will remain the same rather than change. Smoothing techniques discussed earlier can help in this regard.

It is interesting to note that the V chord (G) is not present in the output sequence. These chords seem to be substituted by major IV and major bIII. The first chord is present due to the number of note chord pairings that feature a major IV. This is another issue that must be solved in order to achieve a less dissonant sequence. Increasing the training data with more major V chord pairings can address this issue. The bIII can be addressed with smoothing techniques, as it always appears after a major II or a major III.

6. Conclusion

The goal of this thesis was to create a system that can be trained to create an automatic chord sequence that is viable in the field of western music. While the metrics used to evaluate the system have yielded less than optimal results, this system can serve as a basis for the creation of such a system.

Finite State Machines have been used for speech processing algorithms and this thesis has explored a way in which these concepts might be applied to the problem of automatic accompaniment systems. By using the composition feature, we have generated a Hidden Markov Model that encodes both transition and emission probabilities in a single Finite State Transducer. By composing an additional Finite State Acceptor from a sequence of melody symbols, we can decode this transducer and automatically generate a chord sequence. This system can be improved by including melodic context into the **F** transducer and applying smoothing techniques into the **G** FSA. Both of these techniques are used in the creation of speech to text technology and can be a next step in the further development of the system.

7. Future Work

In order to account for the persistent problems encountered in this thesis it is necessary to implement two changes. The first one will be to include melodic context into the transducer

F. Forsyth had some success by creating a transducer that encodes the probabilities of a second order pairing of chords (2016).

The input alphabet consists of 144 possible note combinations, rather than the 12 used in **F**. This has the advantage of providing some melodic context for the note-chord pairings and would decrease the amount of single note changes found in figure 7b. Each of these 144 note combinations would be mapped to an output dictionary consisting of all the chords found in the training data.

The second improvement to the system deals with the bigram FSA **G**. Given the fact that chord changes occur at a less frequent rate than note changes, we can assign an additional probability weight to **G** that favors a chord being repeated rather than changing to a different chord class. Another possible improvement might be to introduce timing into this system and making it impossible for chords to change inside a predetermined bar duration. For instance, if chord changes remained stable in the example of figure 7b, it would eliminate all instances of C#, thereby making the sequence a lot more consonant.

8. Annotated Bibliography

1. Alpern, A. (1995). Techniques for algorithmic composition of music. On the web: http://hamp. hampshire. edu/adaF92/algocomp/algocomp, 95, 120.

This paper outlines several techniques used in algorithmic composition. Stochastic models are discussed, which are the basis for my system.

2. Bertin-Mahieux, T., Weiss, R. J., & Ellis, D. P. (2010, August). Clustering Beat-Chroma Patterns in a Large Music Database. In *ISMIR* (pp. 111-116).

An approach to quantization of chordal information based on an online vector classification algorithm. Part of my research into design alternatives for the system.

3. Boulanger-Lewandowski, N., Bengio, Y., & Vincent, P. (2013, November). Audio Chord Recognition with Recurrent Neural Networks. In *ISMIR* (pp. 335-340).

An audio recognition system that relies on a recurrent neural network. Used in the literature review to explain the aptitude of RNNs to the problem of automated music generation.

4. Burns, K. H. (2004). Algorithmic Composition.

This textbook outlines how Hidden Markov Models have been used in computer aided composition and the difference between stochastic and rules based approaches.

5. Chadabe, J. (1997). Electric Sound: The Past and Promise of Electronic Music.

This book gives a historic overview of how algorithmic composition and computer music have evolved. From this source several approaches were outlined, such as stochastic methods for computer aided composition.

6. Chen, K., Zhang, W., Dubnov, S., & Xia, G. (2018). The Effect of Explicit Structure Encoding of Deep Neural Networks for Symbolic Music Generation. *arXiv preprint arXiv:1811.08380*.

This is an article describing the use of deep generative models for the generation of music. This forms part of the literature review of the field of dynamic probabilistic models, which is the cutting edge of the field.

7. Chen, P. C., Lin, K. S., & Chen, H. H. (2013). Emotional accompaniment generation system based on harmonic progression. IEEE Transactions on Multimedia, 15(7), 1469-1479. Chicago

This paper describes an automatic accompaniment system that focuses on chord accompaniments rather than individual chords. It incorporates a valence and arousal curve. It was included as a review of the field.

8. Chuan, Ching-Hua, and Elaine Chew. "Generating an evaluating musical harmonization that emulate style." *Computer Music Journal* 35.4 (2011): 64-82.

This paper serves as one of the bases for the metrics proposed in the evaluation of the system. The authors propose several metrics for evaluation that take into account distances between chord distributions.

9. Chuan, C.H. and Chew, E. “A Hybrid System for Automatic Generation of Style-Specific Accompaniment,” Proc. of the 4th Intl. Joint Workshop on Computational Creativity, London, 2007

This paper contains information on metrics used to judge automatic accompaniment in the systems designed by the authors. It focuses on percentage of melody notes found within the chord ones generated.

10. De Clercq, Trevor, and David Temperley. "A corpus analysis of rock harmony." *Popular Music* 30.1 (2011): 47-70.

This paper describes the creation and use of the Rock Corpus Database. This is a collection of melodic and harmonic transcriptions taken from Rolling Stones ‘500 Greatest Songs of All Time’ and serves as the training/testing set used in this thesis.

11. Flowsson, Anders, and Anders Friberg. Algorithmic composition of popular music. Proceedings of the International Conference on Music Perception and Cognition. 2012.

This paper gives an overview of the algorithmic compositional process uses a concept titled global joint accent structure, that enhances the interaction of rhythm and melody based on patterns that are recognized by listeners

12. Forney, G. D. (1973). The Viterbi Algorithm. *Proceedings of the IEEE*, *61*(3), 268-278.

The Viterbi Algorithm finds the best suitable path through a transition matrix between several states. It is the basis for finding the best possible chord sequence from an array of choices.

13. Forsyth, J. P. (2016). Automatic musical accompaniment using finite state machines (Doctoral dissertation, New York University).

Doctoral dissertation providing information on an automatic accompaniment system using Finite State Machines and Transducers. This is the basis for this thesis and the starting point design wise of this system.

14. Forsyth, J. P., & Bello, J. P. (2013). Generating musical accompaniment using finite state transducers. In *16th International Conference on Digital Audio Effects (DAFx-13)*.

This paper focuses on a method for automatic musical accompaniment by using finite state machines to pair chords with melody notes. This is one of the main processes in the system.

15. Fujishima, T. (1999, October). Realtime Chord Recognition of Musical Sound: a System Using Common Lisp Music. In *ICMC* (pp. 464-467).

Pitch Class Profiles or Chroma are feature vectors that show the amount of harmonic energy classified in twelve pitch bands. This thesis will use these feature vectors for training of the HMMs.

16. Grout, D. J., & Palisca, C. V. (1996). A history of Western music (No. Ed. 5). WW Norton & Company, Inc.

A comprehensive history of music in the western tradition. This book was used for the general background and the historical development of chords and harmonies in conjunction with melodies.

17. Jurafsky, Daniel. "Speech and language processing: An introduction to natural language processing." *Computational linguistics, and speech recognition* (2000).

This book deals with the theory of Finite State Machines in regard to its applications in the field of language processing.

18. Isophonics Reference: Annotations Data Set http://isophonics.net/content/reference-annotations

Annotated data set containing chord changes over time for western music. Includes confidence levels pertinent to the successful design of the system.

19. Laitz, S. G., & Bartlette, C. (2010). Graduate Review of Tonal Theory. New York..

A review of tonal theory in the western tradition. This source will be used as a general information guide when creating binary chord templates and a justification for key transposition.

20. Mesz, B., Trevisan, M. A., & Sigman, M. (2011). The taste of music. *Perception*, *40*(2), 209-219.

Harmonic dissonance calculations is discussed in this journal based on the total intervallic dissonance. This is one of the evaluation metrics.

21. Mohri, M. (2009). Weighted automata algorithms. *Handbook of weighted automata*, 213-254.

Operations possible for Finite State Machines (specifically Transducers) are discussed in this manual. Will be pertinent to the design of my system.

22. Mohri, M., Pereira, F., & Riley, M. (2002). Weighted finite-state transducers in speech recognition. *Computer Speech & Language*, *16*(1), 69-88.

Discussion of how WFSTs can be characterized as Hidden Markov Models in speech recognition algorithms. Directly relevant to the design of my system.

23. Musick, Michael, Jonathan P. Forsyth, and Rachel M. Bittner. "Building a harmonically eco systemic machine: Combining sonic ecosystems with models of contemporary harmonic language." *ICMC*. 2015.

This paper describes the application of FSTs into a complete automatic accompaniment generator. It describes how speech processing technology can serve this purpose. It is a companion paper to 'Generating musical accompaniment using finite state transducers'

24. Pardo, B., & Birmingham, W. P. (2001). Following a musical performance from a partially specified score. Ann Arbor, 1001, 48109.

This paper describes a system for score transcription from a musical performance. It was included to be part of the review of the field as well as for ideas for MIDI transcription from audio signals.

25. Raffel, Colin (2017) Lakh Midi Data Set v0.1. Retrieved from http://colinraffel.com/projects/lmd/

MIDI Data Set for popular music linked with the Million Song Database. This will be used for constructing the training and testing data set.

26. Ron, D., Singer, Y., & Tishby, N. (1994, July). Learning probabilistic automata with variable memory length. In Proceedings of the seventh annual conference on Computational learning theory (pp. 35-46). ACM.

This paper deals with the difficulty of computing long sequence of events as part of Markov Chains. The concept of Variable memory length is discussed. It was reviewed as part of the review of the field for this thesis.

27. Simon, I., Morris, D., & Basu, S. (2008, April). MySong: automatic accompaniment generation for vocal melodies. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 725-734). ACM.

This paper describes MySong, a system which serves as a template for this thesis. It uses a Hidden Markov Model trained with popular music recordings to create a musical accompaniment, aimed at non-trained musicians who input a melodic audio input.

27. Vidal, Enrique, et al. "Probabilistic finite-state machines-part II." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27.7 (2005): 1026-1039.

This article provides information regarding finite state machines and the algorithms used in speech processing technology. This covers how hidden Markov Models can be created from N-Grams to generate strings of symbols.