Musiplectics: Computational Assessment of the Complexity of Music Scores

Ethan G. Holder

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Science

Eli Tilevich, Chair
Amy Gillick
R. Ben Knapp

April 23, 2015
Blacksburg, Virginia

Keywords: Music Scores; Music Complexity Assessment; Novel Computing Domains; MusicXML

Musiplectics: Computational Assessment of the Complexity of Music Scores

Ethan G. Holder

(ABSTRACT)

In the Western classical tradition, musicians play music from notated sheet music, called a score. When playing music from a score, a musician translates its visual symbols into sequences of instrument-specific physical motions. Hence, a music score's overall complexity represents a sum of the cognitive and mechanical acuity required for its performance. For a given instrument, different notes, intervals, articulations, dynamics, key signatures, and tempo represent dissimilar levels of difficulty, which vary depending on the performer's proficiency. Individual musicians embrace this tenet, but may disagree about the degrees of difficulty.

This thesis introduces *musiplectics*[1], a systematic and objective approach to computational assessment of the complexity of a music score for any instrument. Musiplectics defines computing paradigms for automatically and accurately calculating the complexity of playing a music score on a given instrument. The core concept codifies a two-phase process. First, music experts rank the relative difficulty of individual musical components (e.g., notes, intervals, dynamics, etc.) for different playing proficiencies and instruments. Second, a computing engine automatically applies this ranking to music scores and calculates their respective complexity. As a proof of concept of musiplectics, we present an automated, Web-based application called Musical Complexity Scoring (MCS) for music educators and performers. Musiplectics can engender the creation of practical computing tools for objective and expeditious assessment of a music score's suitability for the abilities of intended performers.

This thesis is based on research submitted for publication at ONWARD'15.

---

[1]musiplectics = music + plectics, Greek for the study of complexity

Dedication

To the memory of Coppertone Holder, the standard by which all other pets shall be measured.

Acknowledgments

I have not experienced anything in this world quite like the publication of my master's thesis. It has taken only my last two years of graduate school to accomplish, but it has brought me (kicking and screaming at times) further along in those two years than I could have possibly imagined. This process took its toll many times not only on myself, but on all of those close to me. Thus, I'd like to take a moment to graciously acknowledge and thank all of those that had a hand in both shaping my research and keeping my sanity.

I must first thank my parents, Mark and Laurel Holder, without whom I would not even be here. From a very young age, they instilled a love of learning in me and constantly encouraged me, even when I stumbled along my way. You are always a guiding light and a safe harbor in storms. I could write a thesis on this topic alone, but I'll suffice it to say I owe it all to you, and I can never thank you both enough.

My fiancée, my silly, my partner in crime, my goofball, my Baehr, my BMVHB, my coffee-holic, my chocolate monster, my running buddy, my unsung hero... You have all these names to me, but more than each, you are the love of my life Bryanna Baehr. You inspire me in new ways every day and make all of this worth while. I'm so thankful for all the love you give, especially on my worst days. And don't worry, my vows can't possibly be as long as this thesis.

To Logan Patterson, we've been great friends since Kindergarten, when life was so much simpler, all the way up to and beyond our college time. We even managed to live together in a dorm and an apartment for 3 years with two other hooligans. Whether it was bouncing research ideas off of each other and calling out some that weren't up to par or just planning our next hair brained scheme and gaming on xbox, we were together through it all. Although our motives were often simply so we could play more video games later, you've kept me sane and challenged me to perform at my very best, and for that I am always grateful.

To Chris Schmucker, with a name like Schmucker's, it's gotta be good. You've always been there for me when I just needed to vent and escape for a little while. We've also gotten into a fair amount of trouble all around the country from band trips in high school to concerts for other artists. You've even saved my life on at least one occasion. I owe you so much and more. Thank you always and stir the coffee good sir.

One person above all others has been instrumental along my research career. Dr. Eli Tilevich, you've been my advisor and even more to me since my time beginning research as a lowly undergrad when I just wanted a fun project to get some experience. You have guided me through this crazy process and taught me uncountable lessons along the way. I can't say that all of these lessons adhered to conventional wisdom, such as how to determine the true prestige of attending certain conferences based on weather patterns or how to play a great April fools joke on the faculty, but they were nonetheless entertaining and effective. Sometimes lessons could be more eye opening as well. I still remember how much I believed myself to be a fairly decent technical writer until we co-edited my first paper... I could go on and on here, recalling many late nights of research deadlines, but through it all you have made me a better researcher, a better student, and a better person. I will never forget that.

I'd also like to thank the esteemed members of my committee, Dr. Amy Gillick and Dr. R. Ben Knapp. Dr. Gillick worked tirelessly with me and Dr. Tilevich throughout our initial paper submission. Dr. Knapp additionally helped us refine our approach for an ICAT exposé. All have been invaluable contributors to this research work.

I also owe a great deal of thanks to Dr. Cliff Shaffer, a dedicated professor and reference, who has encouraged me to get involved in research in only my first semester as an undergraduate. I must also thank the numerous professionals that advocated the research process to me as both an undergraduate and graduate student, including Mr. Mack McGhee, Mrs. Andrea Bailey, Mr. Adrian Rodriguez, and Mr. Steve Choquette. I would like to additionally thank several professional mentors from my internships and campus positions, including Mrs. Susan Arnold-Christian, Mr. Frank Marullo, Mr. Jose Rodriguez, Mr. Raymond Parsons, and Mr. Salil Bapat.

I generously thank the members of my software engineering research group, without whom much of this research would not have been possible. I must express significant thanks to all of the other various professors and advisors who have worked with me over my years at Virginia Tech. Among all of these, I would be remiss if I did not expressly thank Mrs. Terry Arthur. Your guidance from the very first time I met you, before I even began classes, all the way through until my end as an undergraduate has been outstanding in every way. Last but not least, I must thank all of my other family and friends who pushed me to achieve as much as I have.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Which piano concerto is more difficult: Rachmaninoff's Second or Third? A newly appointed band director wonders if this new orchestral score is appropriate for a high school band, given that the clarinet and bassoon sections are quite advanced, while the flute and oboe sections are more novice. Music educators working on pedagogical guidelines for K-12 students are trying to decide whether a given piece belongs in the N or N+1 curricular level. A publisher wonders which audience to target when marketing new works, while the publisher's customers face great uncertainty when determining whether unfamiliar music matches their playing ability. Performers, band directors, educators, and publishers encounter these non-trivial questions throughout their professional careers.

Unfortunately, determining the relative complexity of music is a non-trivial cognitive task. Additionally, methods in the current state of the art depend solely on individual opinions, a process influenced by personal biases and lacking common criteria. In other words, the only way to answer these questions in a viable way is to carefully analyze music scores by hand, a tedious, error-prone, and time-consuming process. The stakeholders at hand would rather spend their precious time on more creative pursuits.

Can computing help decode these persistent and challenging questions? Is it possible to provide such technology in a ubiquitous and user-friendly way, accessible to any interested musician? To answer these questions, this paper presents musiplectics, a new computational paradigm, that systematically evaluates the relative difficulty of music scores, thus benefiting educators and performers. Two insights provide a foundation behind musiplectics. First, certain notes and other musical components, including intervals, dynamics, and articulations, are harder to play than the others. Second, automated computer processing can transform a prohibitively tedious, error-prone, and subjective process into a practical and pragmatic solution if exposed via an intuitive user interface. Hence, musiplectics fuses commonly accepted music tenets and novel computing paradigms, to objectively answer the questions above.

Musiplectics draws its inspiration from computational thinking [91]. The problem of estimating the expected performance efficiency of a given program has been studied in great detail. We have computational approaches that can predict the amount of computational resources that will be required to execute a program [71] [85]. By tallying the costs of individual instructions, one can estimate the overall cost of executing a program on a given platform. Analogously, individual musical components also have agreed upon costs, defined in terms of the difficulty they present to performers. By decomposing a music score into its individual musical components, one can use their unit costs to compute the total complexity of executing a score on a given instrument, as shown in Figure 1.1.

Figure 1.1: The process of decomposing code and music to extrapolate a conclusion.
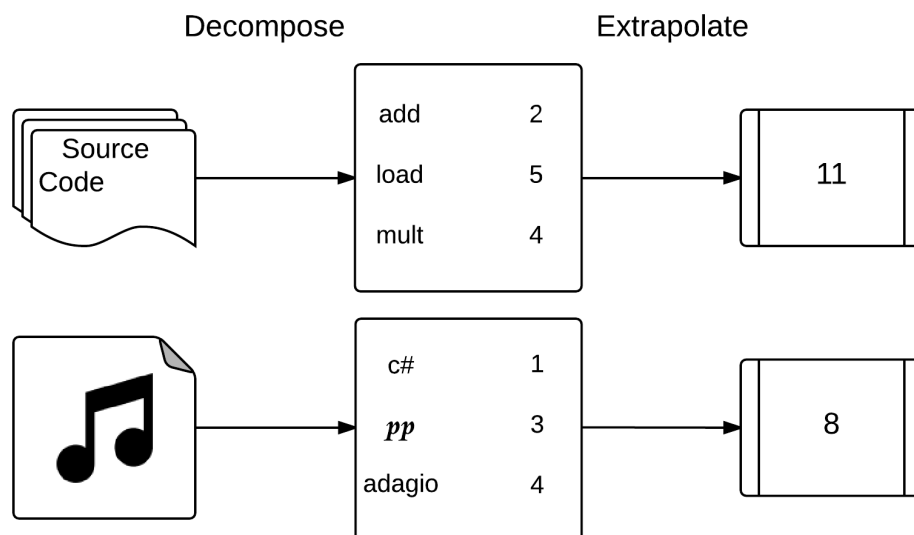
Although one can draw various analogies between music and computing, the intricacy of determining the complexity of music scores is most similar to estimating program performance on different computing devices. While the same compiled program can be executed on any computing device of a given architecture, the device's processing power ultimately determines the efficiency of the resulting execution, which can vary widely. The same applies to the complexity experienced by musicians with dissimilar levels of proficiency when performing the same piece on a given instrument. Although all musicians read a piece of sheet music and understand it similarly, the complexity of playing that piece is determined by a performer's proficiency. Musiplectics aspires to blaze a trail toward objectively assessing these complexities by creating a practical computational framework that can capture the subtle nuances of musical complexity.

The solutions presented herein are instrument agnostic. Nevertheless, to realize the concept of musiplectics, the reference implementation of the computational framework targets the

B♭ Clarinet. This exclusive focus on clarinet is reflective of my own music performance expertise, rather than of any limitations of the presented concepts.

1.1   Research Contributions

By presenting my work, this thesis makes the following contributions:

1. Musiplectics, a new application of computing that encompasses automated assessment of the complexity of music scores.

2. A computational model for music complexity; it assigns base levels of difficulty to an instrument's notes and intervals, with dynamics, articulations, key signatures, and note duration serving as multipliers that increase or reduce the base difficulty.

3. Musical Complexity Scoring or MCS, a concrete realization of the musiplectics-based model above, made publicly available online for teaching and experimentation.

4. A preliminary evaluation of the model's realization, which compares MCS to commonly accepted educational level guidelines for repertoire pieces.

5. An evaluation of music Optical Character Recognition (OCR) software as a means of converting sheet music into a computer-readable MusicXML format.

1.2   Definition of Terms

Due to the nature of this research, several disciplines intersect and bring with them specific definitions and connotations for certain terms. Here I outline such terms used within this paper that may conflict with definitions used in practice by outside diciplines.

First and most obvious are the terms "complexity" and "difficulty". Throughout this thesis, complexity and difficulty are utilized interchangably to refer to the level of play required to successfully reproduce or execute a piece of music by a performer. The intent for these terms is to correlate to the measurements specified in graded works. It is most analogous to the difficulty of an olympic dive being performed, where a higher score implies a higher degree of difficulty. Similarly, these terms do not encompass the measurement for what a performer actually does (how well the dive was executed). This is not necessarily true for the definitions of complexity and difficulty within the realms of psychology or philosophy, but for the purposes of this work, they are treated as identical.

Along the same vein, here the terms mental and cognitive, as used to describe a certain component of complexity, are also used interchangably. In other disciplines mental and

cognitive can separately refer somewhat analogously to affect and effect. In this thesis mental and cognitive are used in the same way to describe the components of complexity which are within a person's understanding (as in how difficult it is to simply understand exactly what a piece of music requires), as compared to mechanical or physical components (as in how difficult it is to execute the understood piece of music on a given instrument). Mental or cognitive compared to mechanical or physical is somewhat analogous to reading compared to writing.

1.3   Paper Roadmap

The remainder of this thesis is structured as follows. First, Chapter 2 presents background information on music to provide a base line for readers. Then, Chapter 3 highlights related work in this field. Next, Chapter 4 explains the basic model for determining music complexity. Afterwards, Chapter 5 details the proof of concept's design and details. This leads into the evaluation chapter 6 which shows how I have tested this system and what preliminary results I have uncovered. Chapter 7 interprets the results and discusses how the system compares to the state of the practice. Afterwards, Chapter 8 gives directions for future work. Finally, Chapter 9 presents my conclusions for this project.

Chapter 2

Background

This research is concerned with evaluating musical complexity. Music is a very specialized domain with its own unique set of concepts, terms, and conventions. Hence, for the benefit of the reader not familiar with Western music, I next present a brief refresher of the standard elements of music. The reader well-versed in this art can safely skip this chapter. Although I have striven to adhere to the standard definitions of the common musical elements, this chapter distills music theory down to the core concepts for the sake of brevity. For more advanced background information beyond what is provided here, there are many free resources available, such as MusicTheory.net [63] or the Society for Music Theory [78].

The fundamental building block of music is the note. Notes in music are similar to words in spoken language or tokens in code. The characteristics expressed by a sound require multiple types of visual symbols to express a note. In written music, the most important characteristics of notes are pitch, duration, dynamics, and articulation. In sheet music, notes are depicted as ovals, both with and without a line (stem) and flag attached to them.

Notes are placed into measures. A measure is a uniform unit of time that breaks up a music piece into smaller segments. These smaller segments endow the piece with a repeating rhythmic emphasis, analogous to meter in poetry.

Pitch is the most salient characteristic of any note. It is the frequency of the sound wave the note produces, or how high or low a note sounds to the listener. Pitches are not continuous, but rather fall on specific discrete steps or half steps within the range of audible frequencies.

An interval is the difference in pitch between two notes. Each interval's name encodes the distance between pitches with different levels of gradation. For example, a minor third, a perfect fourth, a major sixth, etc.

The pitch of a note can be determined from the note's vertical placement on the staff (the five horizontal lines and four spaces in between), the clef (the range of notes possible to represent on the staff), and the key signature (the pitches to be altered up or down a half

step). Accidentals can also similarly alter the pitch of a note up or down by half steps; but accidentals are not applied to the entire piece, only one specific measure at a time. Both accidentals and key signatures are represented by symbols called sharps or ♯, which raise the pitch a half step, flats or ♭, which lower the pitch a half step, and naturals or ♮, which undo the effect of the previously applied pitch modification. Web designers can draw an analogy from key signatures and accidentals in music to global and inline cascading style sheets (CSS) [16] in web page rendering.

Another central characteristic of a note is its duration. Duration is simply the length of time a note is sustained. Although written music does not explicitly specify duration, it is inferred from the note's value (a fraction expressed by the note's stem and flags), the time signature (how many beats are in a measure and what fraction of a whole note gets a beat), and the tempo (how many beats are played in a given time frame).

For example, the value of a quarter note is $1/4^{th}$ that of a whole note. In three four time, the first number says there are three beats in a measure, and the second number says that a quarter note receives one beat in a measure. If the tempo is 120 bpm (beats per minute), then this example measure would last exactly 1.5 seconds (3 beats in a measure / (120 beats per minute / 60 seconds/minute)).

Dynamics refers to the volume of notes. Dynamics are specified with different Italian words, such as piano (quiet), forte (loud), and mezzo (medium), expressed by letter **p**, **f**, and **m**, respectively. Composer combine these letters to express a wide variety of dynamic levels, typically ranging from **ppp** (pianississimo, meaning very, very quiet) to **fff** (fortississimo, meaning very, very loud). There are also markings for gradually changing dynamics to louder (crescendo) or softer (decrescendo or alternatively diminuendo) that look like elongated greater than or less than symbols below the staff.

Articulation is how a note is played and linked to subsequent notes. The simplest analogy for articulations are how different letters or sounds are formed and connected in spoken language. For example, speaking "ta" and "la" have the same "a" sound once held out, but their initial articulations are different because the "t" and "l" sounds are uttered differently. There are many different articulations, but the main ones utilized in this work are as follows:

- Accent or >, which means to play the note louder than those around it.

- Staccato or ·, which means to play the note shorter than its full value, cutting it off early.

- Tenuto or – , which means to play the note slightly longer than its full value, in a connected manner to the following note.

- Marcato (a strong accent) or ∧, which means to play the note much louder than those around it, more than an accent.

- Slur or $\smile$, which means to separate only the first note, connecting all the following notes together.

Although musical notation possesses additional advanced characteristics, including timbre, further articulations, and elaborate music markings, we did not find these advanced symbols as common enough to be useful to consider in a general complexity model. When writing music, composers combine the concepts above to express the desired artistic impression the composition is intended to make on the listener. Hence, it is hard to distinguish which notational concept is more important than the other for artistic expression. Nevertheless, we have found the subset presented above as absolutely essential to realizing the ideas behind musiplectics.

Chapter 3

Related Work

Multiple prior research efforts study various facets of music. Although relatively few works concretely focus on analyzing music complexity 3.1, tangential studies into scanning and searching music 3.2 as well as classifying music 3.3 represent related research efforts. Each has tangible ties to this work's objectives, albeit some at a more abstract level than others.

3.1   Complexity Analysis

The most relevant works to musiplectics are those that also seek to analyze the complexity of music in some way. A representative example is presented in reference [28]. This work seeks to automatically generate or predict the difficulty of a piano music piece. The authors apply regression theory over a set of features that cause difficulties for that specific instrument. Musiplectics leverages similar musical concepts to build up an aggregate measure of difficulty or complexity. However, my approach has a wider range of applicability, both in terms of musical instruments and playing proficiency, as well as portability, by embracing uniform types of complexity parameters. In other words complexity parameters in musiplectics encompass the cognitive and mechanical complexities for a given instrument. So all instruments have the same types of complexity parameters, which may take upon vastly different values.

Similar to [28], [43] studies the complexity of playing guitar. Their work extracts features that determine difficulty when playing guitar. However, their focus lies more on the mechanical difficulties specifically associated with that instrument rather than broad ranging complexities of playing music on any instrument. Musiplectics also takes into account the playing proficiency of the player at hand.

On a related tangent from both [28] and [43], Laban Movement Analysis (LMA) [60] [12] [27] provides a means of analyzing musical performances as they occur. LMA was originally

created as a means of analyzing dance performances, but it has been used extensively in analyzing music performances as well [22] [21]. The main difference between LMA (as well as the related field of Labanotation [41] [49]) and the work presented here is when to analyze the complexity. LMA and Labanotation focus on the actual performance and the execution of a piece of music (or equivalently dance or many other physical motions), whereas Musiplectics focuses on the theoretical performance that has been specified by the piece of music. However, the categories studied in LMA have some rough analogues with both [28] and [43], and thus indirectly correlate with musical elements analyzed in Musiplectics.

Another related approach lies in [56]. The authors analyze the complexity of the rhythmic components of various pieces of music. They utilize the L-system to breakdown the rhythm of a piece into a tree structure and then apply tree analysis algorithms to generate a score. Although musiplectics avoids complex algorithms for examining the rhythmic structure of a piece, it considers a full array of the elements of music scores rather than simply rhythm alone.

State organizations, such as [87] in Virginia and [67] in New York as well as others, similarly analyze music scores by hand, an activity which we hope to automate. These organizations govern K-12 schools in their respective states. They also list music pieces and their respective difficulty grades for district, regional, or state competitions for each K-12 grade level. Other organizations, such as the Royal Conservatory Music Development Program [76], offer similar pieces and respective grades as part of their level requirements and assessment regulations. Unlike state organizations, Royal Conservatory publishes their pieces and scores to the public, a provision that enables leveraging them in evaluating this work.

The difficulty grading schemes in both types of organizations are analogous to the complexity of the piece, except that in these organizations pieces are graded subjectively by a group of people rather than a uniform algorithm. Additionally, the grades are typically listed as integer values between 1 and 10, thus lacking a sufficient level of granularity to express nuanced differences between musical pieces.

Further past works have specifically studied the complexity of music as experienced by listeners or observers. These works show great insights into defining complexity and in fact utilize very similar music elements as representative features to determine overall complexity. For instance, [57] relies on note pitches, intervals, and durations of notes as features on pieces of MIDI formatted music to determine the complexity experienced by listeners. Similarly, [81] presents an entire Ph. D. thesis on characterizing the complexity of music from the viewpoint of the listener. Other works have specifically utilized listener complexity in related studies [44] [72] [61] [69] [90] [18] [65] [80].

Several related works from [81] also formally define what complexity is in terms of listeners and what information can be gathered from it as a property of music [33] [74] [77]. While this thesis focuses on the complexity experienced by performers, such related works all provide a basis to build upon [55]. Future work could seek to compare the complexity experienced by listeners or observers to that of the performer to see what similarities (if any) arise.

3.2   Music Scan & Search

One area of research tangential to analyzing music complexity is scanning and searching for music. The overlap lies chiefly in the end use cases in both areas of research. Educators, performers, and other stakeholders, all find themselves needing to efficiently locate musical pieces that meet their respective requirements. Providing a complexity score is one means to improve the efficiency of searching for new music, since users can see complexity at a glance or even search by complexity to find a piece to prepare. [19] gives many reasons for why this is necessary, but there is a whole body of research related to music information retrieval that deals with similar problems [32] [23].

Another area of overlap between musiplectics and music scanning is in translation. From a high-level, reading in any form of music and writing out a related, different form is essentially translation, an area which I have previously explored in some depth [47] [45] [48]. There are many research efforts to translate forms of music into other languages or versions.

An especially interesting effort in this regard is [4]. The authors work to convert polyphonic music (music with several simultaneous notes on one or several instruments, such as a band playing together in harmony or one person playing piano or guitar for instance) into equivalent monophonic music. Their end goal is to reduce a large, expressive format to a more simple one for comparing pieces during a lookup. The reduction in complexity of chords down to single notes represents an interesting approach that could be leveraged with musiplectics, so as to generate a potentially less complex version of a given piece.

3.3   Music Classification

Music classification presents another area of potential research overlap. Much research has previously dealt with using computers to understand music and thus classify into various genres. While the efforts of music classification are not the same as determining complexity, the approaches taken to classify certain pieces via machine learning and statistical methods are important, because they present means of automatically analyzing music. At some point musiplectics could potentially apply similar machine learning concepts to interpret what makes music complex (rather than decomposing music scores into individual elements and calculating their summary complexity) as well as leverage genre classification as another source of potential complexity.

Cuthbert, Friza, and Friedland [31] for example focused heavily on using machine learning to classify different types of music. The authors can extract multiple features from a variety of input types and apply their theorem to successfully classify the genre of several inputs.

Similarly, [24] proposes an approach to classifying music files in MIDI format specifically. The authors form an approximation of the Kolmogorov distance using the normalized compression

distance between approximate string representations. They use this approximation as the main feature to classify numerous music files.

Chapter 4

Computational Model for Music Complexity

In Chapter 2 above, I discussed several fundamental music concepts. These concepts serve as the baseline elements that factor into the complexity score.

The most straightforward approach to calculating an overall score is to assign whole number weights to each element perceived to be especially important (i.e., notes, intervals, dynamics, articulations, key signatures, and note durations) and add all the weights up. This scheme, however viable, fails to adequately reflect the experience of playing music. At their core, notes and intervals present distinct difficulties on their own, whereas dynamics, articulations, key signatures, and note durations only modify those difficulties. For instance, a small interval may seem easy on its own, but with changing dynamics, with differing articulations, in a strange key, or at a high tempo, that interval could become much more difficult.

Hence, a more authentic approach to calculating an overall score is to only assign whole numbers weights to notes and intervals. These are still counted and summed up into a final score. However, dynamics, articulations, key signatures, and note durations become multipliers onto notes and intervals. Each dynamic, articulation, and key signature thus receives a multiplier weight that is a decimal (typically between 0 and 2). Those multipliers are applied to every occurrence of a note or interval.

Note duration is factored into the score as an average over all notes. The total amount of notes and associated duration in seconds is calculated at the end and applied as its own multiplier. The more notes in a given span of time, the higher the multiplier becomes.

This scheme captures the concept previously envisioned that makes duration add to overall complexity, except in the extreme case of playing excessively long notes. In cases of wind instruments, such as the main target of B♭ Clarinet, holding notes out for long durations may possess its own difficulty in providing adequate breath support, rather than the difficulties of changing finger positions and embouchure quickly.

Thus, the model adapts the note duration multiplier to be a multiplicative or fractional difference from one. If the average of notes per second in the piece is 1.5, then the multiplier remains 1.5. However, in the case of many long notes, the average of notes per second might be something closer to 0.5. In this case (when the average is less than 1), the fraction becomes its reciprocal, 2 in the example.

Users cannot directly change this parameter as it is built into the model. However, they can still influence the degree to which this parameter affects the overall score. To that end, the user can parameterize note durations with a multiplier value that increases or reduces the impact of the note duration parameter. For the cases when the user is content with the built-in setting, the model applies the default value of 1.

Chapter 5

Proof of Concept

In 5.1, I outline the basic software design of the proof of concept. Next, I provide some highlights of the extensible architecture in 5.2. Then, I describe the full details of the overall implementation in 5.3.

5.1 Design Overview

The complexity model presented above outlines the basic functionality of the approach. Nevertheless, the implementation's complete control flow involves several steps. The extended control flow based on the model can be seen in Figure 5.1.

From the top left, one can see the inputs to our a musiplectics system are Notation Apps and Pieces in PDF format. Many common notation applications, such as Finale Notepad [58] and Sibelius [11], support a universal format called MusicXML [59] [38]. MusicXML files can be imported, edited, and output back to MusicXML or other proprietary formats. MusicXML is the underlying representation on which the reference implementation operates.

Alternatively, the system can be extended to work directly with music pieces in PDF format. It can accept PDF files and transform them into their MusicXML representation, by means of music OCR (Optical Character Recognition) software. The reference implementation relies on free software from Audiveris [13] currently, but it could equivalently utilize other off-the-shelf OCR applications.

Once the MusicXML representation of the piece is obtained, the automated processor then takes it as an input. The processor computes the complexity score by going through a sequence of steps. First, it parses the piece to extract individual music elements to be analyzed. Then, it looks up the weights for each element as specified by the complexity parameterization.

Figure 5.1: The extended control flow showing the model forming the complexity score.
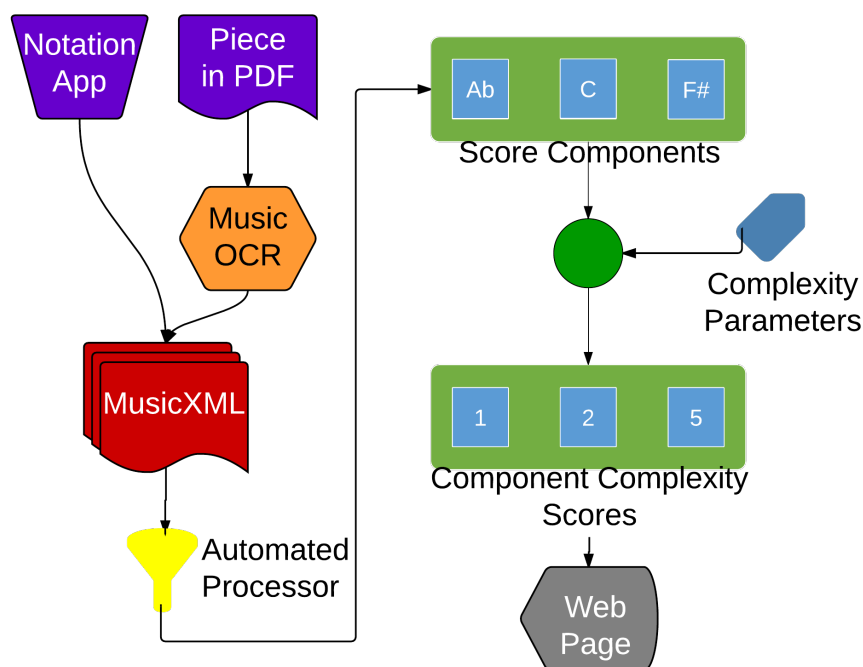
Recall that this system can be parameterized with different complexity weights to reflect dissimilar levels of per-instrument playing proficiency. Currently, these weights are specified in an xml file to facilitate tool integration. Because the weights are meant to be decided on by music experts, the design foresees the creation of visual editors to specify this information. These editors can easily save their inputs in some xml format. Similarly, individual users could create their own file representing their own personal weights and apply the system in the same manner. In the proof of concept, I experimented with 5 different sets of weights, which represent proficiency levels that range from that of a beginner up to a trained professional.

Using the specified weights, the system calculates the complexity by tallying up the difficulty values of each individual type of element, computed separately during a single pass over the structure. The final piece of functionality presents the calculated complexity in a user friendly fashion. To that end the system uses a web-based interface with specialized javascript libraries.

5.2   Architecture Highlights

One key design feature is ease of extendability at each level. Any application that generates valid MusicXML files can feasibly provide input to this system. Furthermore, the overall

process can be extended if other applications can generate the piece of music in PDF form or some intermediate that can be represented in PDF or MusicXML.

The system itself can be extended to operate only on specific selections of pieces or on batches of multiple pieces if necessary (thus showing the complexity of an entire performance). Additionally, the parameters that determine complexity can easily be changed on the fly based on future refinements. This provision enables individual performers or groups to set their own complexity levels to find a score even more relevant to their playing style.

Beyond the extension to the current process, the system can be easily scaled up to operate for many users on many music pieces. The current complexity parameters can be quickly substituted with an external database in MySQL [68]. Additionally, libraries of MusicXML pieces along with user uploaded files can easily replace pre-selected pieces currently available.

Even the backend software itself can be extended and modified in the future with relative ease. MusicXML files are parsed with heavy reliance on the Visitor Design Pattern [34] and judicious overloading [15]. In doing so the intricacies and redundancies of MusicXML are effectively abstracted away, and future additions to the model are easily accommodated.

Finally, the javascript libraries used on the frontend naturally facilitate creating a mobile-friendly web interface. While mobile experiences have not yet been fully optimized, they are still readily available even now. Furthermore, it would be straightforward to extend this interface into a mobile app itself by utilizing past translational research [45] [47] or PhoneGap [36] [5].

5.3   Implementation Details

The reference implementation spans several different code bases, most notably the automated processor backend and the web UI frontend. This style of two-tier architecture is common in both web and mobile applications. Figure 5.2 shows a diagram detailing the specific architecture for the reference implementation and showing how the frontend and backend pieces interact.

As shown, when a user first requests a complexity score from the frontend interface, that request is formatted as AJAX [35] and sent to the backend web server. The physical server receives the request and routes it to the appropriate running service (in this case, the backend). Once the backend receives the request, it fetches the complexity parameters and the piece in MusicXML format specified by the user. Then, the backend code generates the complexity score as discussed previously in the design overview. It formats the complexity score as well as some related data (the most complex measure number and form, note complexity including associated multipliers, and interval complexity including associated multipliers) for each part within the piece of music into JSON [30] data before sending the response back to the frontend. When the frontend receives the asynchronous response, it parses the JSON

data and displays the complexity score and related data in graphical format.

Figure 5.2: The architecture and flow of the backend and frontend code bases.
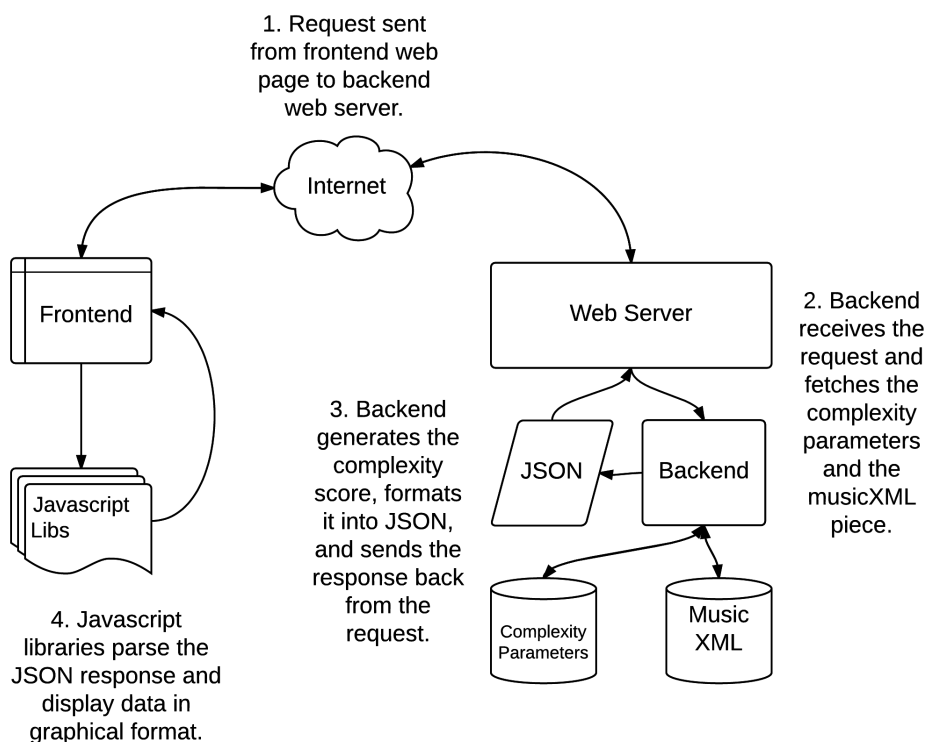
The backend code is written in Java [40] and spans 5137 source lines of code (SLOC) [64]. I've utilized the Apache Xerces 2.11.0 library for parsing xml files [6] and the JSON Simple 1.1.1 library [50] for constructing JSON [30] to pass along to the frontend. This Java backend is compiled and bundled up as an executable. It is then run via command line interface, specifying the complexity parameters and MusicXML to run against.

As mentioned previously, the backend utilizes the visitor pattern heavily when processing the MusicXML file. The visitor pattern eases the handling of the intricacies and redundancies of MusicXML by allowing different operations to be defined on the classes of elements present within MusicXML. It also abstracts away the requirement for a specific MusicXML structure.

The simplest way to demonstrate this abstraction is through the placement of specific xml elements within the MusicXML file. For instance, the elements "sound" and "direction-type" typically hold new "tempo" or "dynamics" elements as children, respectively. Obviously, tempo and dynamics must be tracked as part of the complexity score so their elements must be tracked, thus their parent elements must be tracked as well. However, "sound" and "direction-type" elements do not always appear in the same location within the MusicXML file. Both can occur as a child of a "direction" element or as a child of a "measure"

element. Therefore, the system needs to be robust enough to encounter these types of elements at varying depths within the MusicXML's tree structure. Visitor pattern succinctly accomplishes that goal.

The compiled backend code is currently deployed on a Unix-based Mac Mini [8] running OS X Mavericks (version 10.9.1) [7]. A lightweight Apache server [83] hosts the files and runs some limited PHP to facilitate easy communication between the frontend and backend portions. AJAX requests [35] are sent to the backend and JSON response text is sent back in the current implementation.

The frontend code is written in Javascript, HTML, and CSS and spans 3061 SLOC [64]. To minimize the amount of hand-written code, the implementation makes use of numerous industry-standard libraries, including jQuery 2.1.0 [84], Bootstrap 3.3.4 [70], Datatables 1.10.5 [79], D3 3.5.5 [17], and VexFlow 1.2.27 [26]. These libraries span various standard facets of system implementation, making it possible to focus on the novel, research-related issues of musiplectics when creating the proof of concept.

Both the frontend and backend as currently implemented are publicly available on GitHub [46] along with instructions as to how to set them up personally. The site also features links to the currently deployed version.[1]

---

[1]The current version is deployed at `http://mickey.cs.vt.edu/`

Chapter 6

Evaluation

The goal of this evaluation is to demonstrate that the reference implementation of musiplectics can be a useful tool for music educators. To that end, the system was run with different parameterizations on a set of music scores, used in educational settings. Music educators have also ranked these same scores by hand, producing a baseline for comparison. As expected, even though the general trends reported by the tool corresponds to that provided by music educators, some outliers were also observed. Some pieces turned out to have been much more complex on the relative scale than the pieces immediately preceding and following them in the rankings. These discrepancies can be explained either by immaturities of the implementation or by inaccuracies of ranking of music scores by hand. It is easy to overlook some really complex parts in the middle of a score when trying to assess its suitability for a given playing proficiency. While an automated tool analyzes scores in their entirety, producing the results based on an exhaustive analysis of each and every element.

I first specify the complexity parameters used in the experiments in 6.1. Then, I unveil the strategies one can follow to obtain the settings that represent a consensus among musicians in 6.2. Next, I describe the test pieces selected in 6.3. Finally, I discuss the accuracy of the OCR program tested for suitability in musiplectics in 6.4.

6.1   Clarinet Complexity Parameterization

Chapter 4 above explains the rationale behind complexity parameters. In particular, it differentiates between parameters, expressed as whole number weights and those which are decimal number multipliers. In that presentation, the parameters were not linked to any particular instrument. By contrast, in this Section, I discuss their specific application for a specific instrument, the B♭ Clarinet.

I decided to focus specifically on the clarinet, because it exhibits many forms of complexity

Table 6.1: Note ranges and weights for beginner B♭ Clarinet.

Note Range	Weight
G3-G♯4	1
A4	2
B♭4-C5	5
≥ C♯5	10

Table 6.2: Intervals, note ranges, and weights for beginner B♭ Clarinet.

Interval	Low Range	High Range	Weight
Unison	Anywhere	Anywhere	1
Second	G3-G♯4	G3-G♯4	2
Third	G3-G♯4	G3-G♯4	3
Fourth	G3-G♯4	G3-G♯4	4
Fifth	G3-G♯4	G3-G♯4	5
Any	G3-G♯4	A4-C5	8
Any	A4-C5	≥ C♯5	10
Sixth	Anywhere	Anywhere	9
Seventh	Anywhere	Anywhere	9
Octave	Anywhere	Anywhere	9
> Octave	Anywhere	Anywhere	10

and relates to many similar woodwind and brass instruments. Additionally, my own musical expertise favors this instrument above all others, making it possible to define my own complexity parameters with confidence and not requiring external confirmation.

The initial complexity settings for B♭ Clarinet were for beginners. Based on these values, complexity settings for other levels of B♭ Clarinet were later adapted, but those levels largely changed only the associated weights. Because only the beginner settings were utilized in the tests, they are the only ones discussed below.

Note complexities were broken up into the ranges and assigned weights, as shown in Table 6.1. Intervals were similarly broken up further and assigned weights, as shown in Table 6.2.

Dynamics and articulations were specified for those specific types mentioned previously in Chapter 2. Their weights can be found in Tables 6.3 and 6.4, respectively. Unlike dynamics and articulations, all possible major key signatures are specified with weights and shown in Table 6.5. Finally, the note duration modifier is kept at 1.0, so the note duration modifier works exactly as specified in 4.

Table 6.3: Dynamics and weights for beginner B♭ Clarinet.

Dynamic	Abbreviation	Weight
mezzo forte	mf	1.0
mezzo piano	mp	1.0
forte	f	1.1
fortissimo	ff	1.2
piano	p	1.3
pianissimo	pp	1.5

Table 6.4: Articulations and weights for beginner B♭ Clarinet.

Articulation	Weight
Slur	0.5
Normal/None	1.0
Accent	1.1
Staccato	1.2
Tenuto	1.2
Marcato (Strong Accent)	1.4

Table 6.5: The major key signatures and weights for beginner B♭ Clarinet.

Key Signature	Sharps/Flats	Weight
C	None	1.0
G	F♯	1.1
D	F♯, C♯	1.1
A	F♯, C♯, G♯	1.2
E	F♯, C♯, G♯, D♯	1.3
B	F♯, C♯, G♯, D♯, A♯	1.4
F♯	F♯, C♯, G♯, D♯, A♯, E♯	1.5
C♯	F♯, C♯, G♯, D♯, A♯, E♯, B♯	1.6
F	B♭	1.1
B♭	B♭, E♭	1.1
E♭	B♭, E♭, A♭	1.2
A♭	B♭, E♭, A♭, D♭	1.3
D♭	B♭, E♭, A♭, D♭, G♭	1.4
G♭	B♭, E♭, A♭, D♭, G♭, C♭	1.5
C♭	B♭, E♭, A♭, D♭, G♭, C♭, F♭	1.6

6.2 External Survey

Although the complexity parameterization presented in 6.1 may be accurate, one would not be able to validate them empirically, as they reflect one's subjective personal experiences and beliefs. However, musiplectics embraces this subjectivity, enabling individual musicians to specify the parameterizations that reflect their own individual understanding of their own or their students' proficiency.

As a logical consequence of the previous observation, it would be equally impossible to empirically validate the "correctness" of the computed complexity score of an analyzed piece. However, if stakeholders in a score can generally agree on its relative complexity, the resulting consensus can serve as a viable form of validation.

Based on this assumption, experts, musicians, and educators seem to have a vested stake in the results of these complexity scores. Therefore, I've begun to survey those related to B♭ Clarinet in an effort to ascertain their opinions. In the survey I ask simple questions about the complexity parameters already established, both how the parameters are implemented and the weights assigned to each. Once a statistically significant consensus has been reached or some threshold of responses have been given, the results of the survey will become the new complexity parameters. At the time of writing, neither condition has been met so my own parameters are in use, but it is important to note that I am striving to find an amicable means of determining these parameters.

6.3 Graded Music Pieces for Comparison

Based on the 2014 syllabus for B♭ Clarinet available from Royal Conservatory [76], I selected 2-4 pieces for each grade, 1-10. The pieces were chosen based on availability, so as to minimize the amount of companion book or subscription purchases required. In whole 32 pieces are used for the main comparison: 10 from Standard of Excellence [73], 7 from Clarinet Solos [9], 4 from Concert and Contest Collection [89], and 11 publicly available on IMSLP [75]. Each of these is listed with its author and grade in Table 6.6. These pieces are translated into MusicXML using the OCR process and subsequently passed through my system to obtain a complexity score for each.

Figure 6.1 displays the complexity score of each piece by associated grade. The pieces are in the same order as previously listed in Table 6.6, but are displayed here by grade for readability. Please note that pieces in all grades do have a complexity score, but those in grades 1-3 are less than 1000 and are not discernible in the graphic. Similarly, Figure 6.2 shows the average complexity of pieces by grade from Royal Conservatory. Again, scores for pieces in grades 1-3 are present, but are barely visible due to scale.[1]

---

[1]For more information, the full set of data can be found on our GitHub [46] under Documents/Data/.

Figure 6.1: The complexity score of pieces by grade from Royal Conservatory. Everything to the right of a grade including that number represents a piece with that grade.
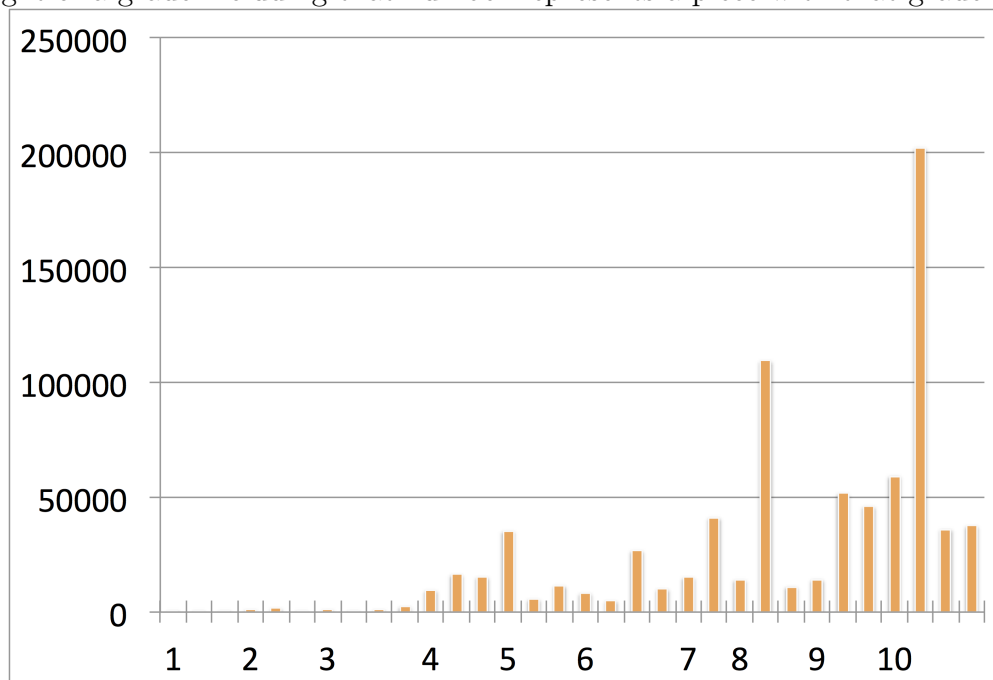

Figure 6.2: The average complexity score of pieces by grade from Royal Conservatory along with standard deviations.
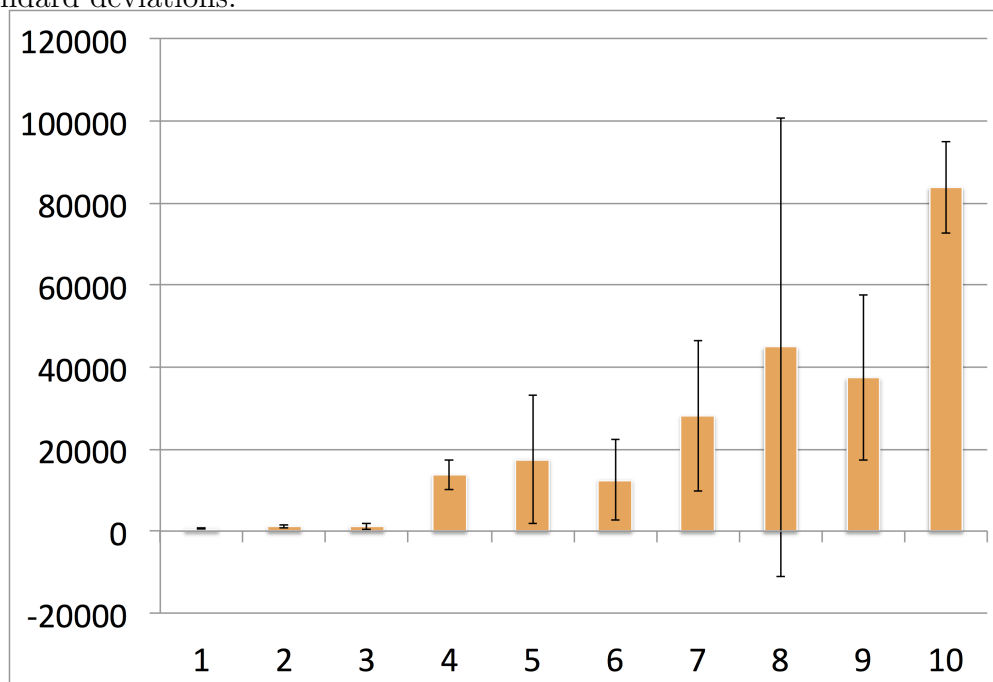
Table 6.6: The works from Royal Conservatory chosen for comparison along with their grade and composer (or book reference if no composer information was available).

Gr.	Title	Composer
1	Bingo	S.o.E.
1	Eerie Canal Capers	S.o.E.
1	Go for Excellence no. 61	S.o.E.
2	Alouette	S.o.E.
2	Grandfather's Whiskers	S.o.E.
2	Ming Court	S.o.E.
3	Just Fine	S.o.E.
3	Variations on a Theme	Mozart
3	Loch Lomond	S.o.E.
3	Theme from Symphony 9	Beethoven
4	Minuet in G	Beethoven
4	Gavotte	Gossec
4	Song without Words	Tschaikowsky
5	Humoresque	Dvorak
5	The Dancing Doll	Poldini
5	Hymn to the Sun	Korsakoff
6	Serenade	Drdla
6	Promenade	Delmas
6	Scherzo	Koepke
6	Nocturne	Bassi
7	Sonata Mvmt. 2	Hindemith
7	Scene and Air	Bergsen
8	Canzonetta	Pierné
8	Concerto Opus 36 Mvmt. 1	Krommer
8	Sonata Mvmt. 1	Saint-Saëns
9	Sonata Mvmts. 3 and 4	Hindemith
9	Sonata Mvmts. 2, 3, and 4	Saint-Saëns
9	Solo de Concours	Rabaud
10	Concerto no. 3 Mvmts. 1 and 2	Crussell
10	Concerto no. 3 Mvmts. 2 and 3	Crussell
10	Solo de Concours	Messager
10	Sonata no. 2 Mvmt. 1	Stanford

6.4   Optical Character Recognition (OCR)

The ability to use music scores in PDF format would greatly enhance the usability of our reference implementation. However, this ability hinges on the accuracy of the music OCR

software that can translate PDF into MusicXML. Hence, I empirically evaluated the accuracy of a widely used music OCR application to assess its suitability for this system.

In the experiments, I used freely available music OCR software from Audiveris [13]. I evaluated the reliability of this process as follows: find pre-matched PDF files with the correct MusicXML, convert the PDF files into a new MusicXML, and finally compare the correct and converted MusicXML to each other.

The pre-matched PDF and MusicXML files used are all from MuseScore [62] and are listed in Table 6.7. They were selected randomly from the list of single part pieces for clarinet. The MusicXML files came in .mxl (compressed MusicXML) format from MuseScore, and they were uniformly imported into Finale Notepad 2012 [58] to then export the uncompressed format for comparison. For this test I use only Audiveris for conversion.

Table 6.7: The works with matched PDF and MusicXML files from MuseScore chosen for testing OCR reliability along with their composer or arranger.

Title	Comp./Arr.
Dancing Clarinet	KRM
Mi Razon De Ser	Banda Central
Rudy	Jerry Goldsmith
The Hobbit: The Desolation of Smaug	Howard Shore
The Rose	Dan White

Figure 6.3: The comparison command for MusicXML files.

```
1    sdiff -B -b -s Original.xml OCR.xml | wc
```

I make use of a file differencing tool with counts of words, lines, and characters as shown in Figure 6.3, as well as a comparison of complexity scores to highlight the potential effects of the OCR process. I show the difference measurements for select pieces in Figure 6.4. I also present the average across pieces for each difference in the same figure. Figure 6.5 shows the difference expressed as the percentage of change from the values of the original MusicXML file. It also presents the average across pieces in the same figure.

Figure 6.4: The difference between matched and OCR generated MusicXML files in words, lines, and characters via sdiff as well as the positive difference in complexity score.
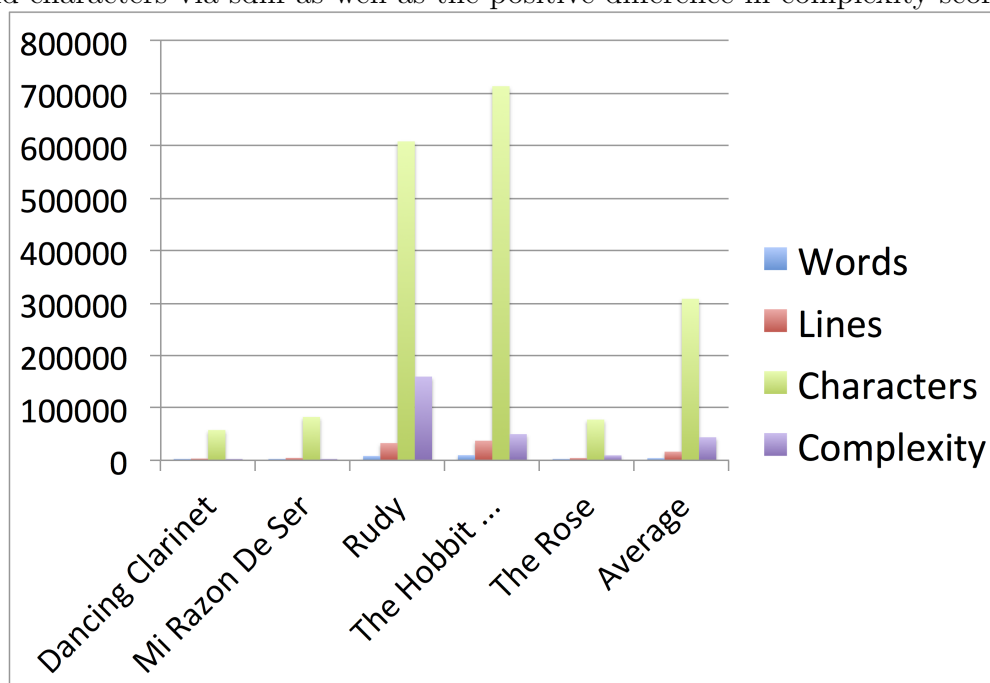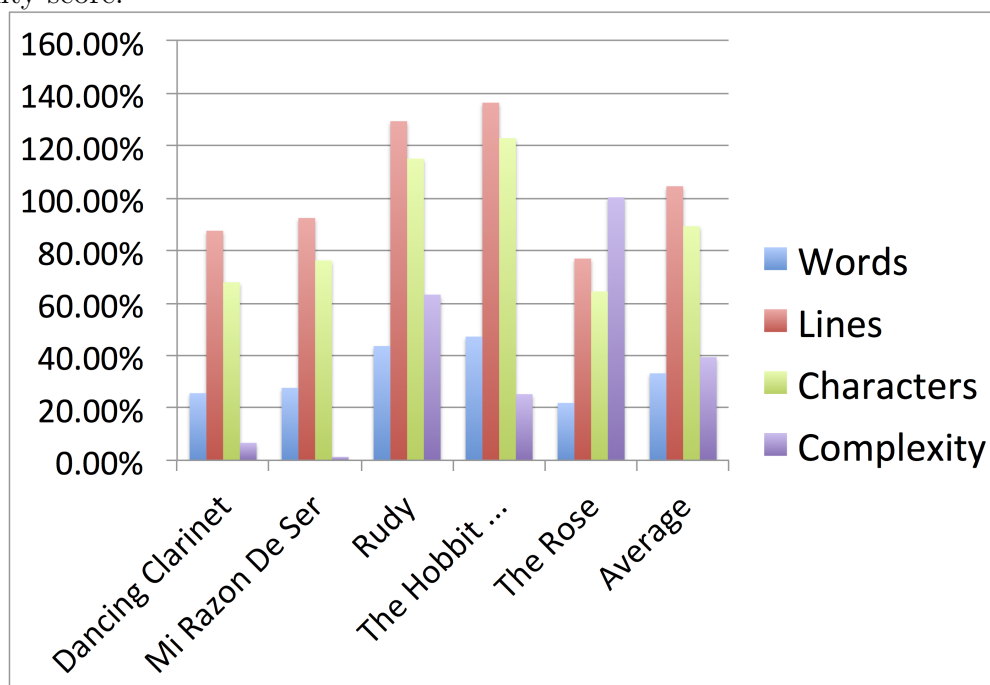

Figure 6.5: The percentage difference between matched and OCR generated MusicXML files in words, lines, and characters via sdiff as well as the positive percentage difference in complexity score.

Chapter 7

Discussion

I discuss my findings as follows: Section 7.1 covers the results from assessing previously graded pieces from Royal Conservatory; Section 7.2 talks about the results from comparing matched and OCR generated MusicXML files; and Section 7.3 discusses the findings with regards to the website's usability.

7.1 Manually Graded Pieces and Their Calculated Complexity Scores

Figures 6.1 presents the results of applying the reference implementation on 32 scores, which have been manually ranked by music educators as belonging to levels between 1 and 10. One would naturally expect the lowest and highest complexity scores to come from grade 1 and grade 10 pieces, respectively. Although the lowest score piece was a higher grade than expected, the second lowest score was indeed for a grade 1 piece, "Bingo", and the highest score was for a grade 10 piece, "Concerto no. 3," 2nd and 3rd Movements by Crussell. This outcome shows in one respect, that my results generally correlate with the expectation. Other discrepancies and outliers are to be expected given the subjective nature of grading pieces.

The graph reveals that even though the automatically calculated complexity scores follow the overall trend set by their manual rankings, there is a lot of noise in the calculated scores. This noise reflects the discrepancies between the manual (baseline) and automated (evaluated) rankings. Some outliers are worth examining in detail. In particular, the lowest computed complexity score of 514.19 was for "Variations on a Theme" by Mozart, a grade 3 piece, while the highest one of 202214.85 was for "Concerto no. 3," 2nd and 3rd Movements by Crussell, a grade 10 piece. Musicians would argue that music by Mozart is known to be deceptively simple. Hence, the low mechanical difficulty calculated by our tool may not

truly represent how human experts see this piece by Mozart, which is a well-understood exception in classical music. The Concerto looks deceptively hard due to the presence of a free-form cadenza that uses the very unusual 17/4 time signature and numerous consecutive tuplets, including both common 16th and 32nd note groups as well as unusual 6-tuplets. It is possible that performers would find this piece much less daunting once they make sense of these rhythms.

To further illustrate correlation with the expectation, Figure 6.2 shows the average complexity score of pieces by grade along with the standard deviation. Here, the complexity scores seem to more closely match what one would expect. The average complexity scores roughly increase and are within one standard deviation of consistently increasing. There are still outliers present, including grade 6 pieces having an average complexity less than grade 5 or 4 pieces, and grade 8 pieces having larger average complexity than those of grade 9. However, all of this evidence is simply a testament to the highly subjective nature of complexity assessment. It is likely that experts presented with the automated results would consider revising their ranking recommendations.

7.2   Matched and OCR Generated MusicXML

Another possible explanation for the discrepancies highlighted between graded pieces could be the inaccuracies of the toolchain, specifically music OCR. During the tests, I largely utilized Audiveris [13] for its simplicity and speed in generating MusicXML as well as its plethora of options for input image formats. However, some files were simply too poor of image quality for it to initially accept. With some manual effort to change image formats and attempts to improve the resolution of scanned images, Audiveris was finally able to catch the remainder of cases.

The process of converting PDF's to MusicXML is admittedly an imperfect means of generating accurate MusicXML representations. As Figure 6.4 shows, there were large differences between the test bed of matched and OCR generated MusicXML files. The difference in characters is the most alarming, however that could be explained by the limited features OCR is able to analyze with respect to the entire set encoded into the matched MusicXML. Nevertheless, the end result of the file differences leads to the associated differences in complexity scores.

Figure 6.5 underscores this difference by showing it as a percentage of the original measure of words, lines, characters, and complexity. While the piece "Dancing Clarinet" has only a 6.68% difference from the original complexity score, the piece "The Rose" has over 100% difference from the original complexity score. Both of these show over 20% difference in words and over 60% difference in lines and characters. Interestingly, the pieces "Rudy" and "The Hobbit: The Desolation of Smaug" both show over 110% difference in lines and characters, yet the change in underlying MusicXML only causes about 63% and 25% difference

in complexity for these pieces, respectively.

Therefore, it would seem that the process of using OCR to generate MusicXML from PDF files of music scores is simply not yet mature enough to handle the demands of complex music scores. This conclusion is strengthened by both outside research [20] and my experience with acquiring such software. The original intention was to deploy such OCR software at the beginning of the control flow from a web UI to allow users to easily upload PDF files without performing a manual conversion on their own. However, of the 5 different OCR packages I experimented with and nearly purchased, most did not even offer an option for batch execution. Those that did offer this option could not operate in this mode consistently or with a measure of reliability.

While this process is obviously imperfect, its speed and automation allow us much more flexibility in generating MusicXML even outside of batch mode. This process is still absolutely necessary for comparing complexity scores of well-known works, since so many have not previously been rewritten into MusicXML. At this point there is no clear substitute for music OCR, but it is my hope that future efforts will strive to improve both the accuracy and reliability of this process so more research can be performed with sheet music.

7.3   Website Usability

I do not provide any empirical data so far on the performance of the deployed implementation, but it is publicly available for general use. In informal discussions with potential users, I uncovered many points of design that I discuss here.

First, the website must provide access to the matched PDF file as well as the complexity score from the MusicXML representation. The PDF file is made available so that there is no confusion about exactly what the piece of music is that is generating the score.

Second, I found that many users especially wanted to see what the most complex measure is in the piece as a means of determining whether the complexity score was due to one very difficult spot or a collection of many less difficult areas. Thus, I determine the complexity score for each individual measure in the pieces available. I not only show the measure number and associated score however, I also utilize VexFlow [26] to graphically represent this measure to further accommodate users.

Finally, the website also features the ability to run on music pieces with multiple parts or instruments. The relevant data for each part or instrument is extracted and even graphed against one another through D3 [17]. This comparison is not necessarily correct, given that each part or instrument is assigned a complexity score based off of parameters for B♭ Clarinet. Yet, it is still a valuable design point to show the general applicability of this approach to works that are not only for instruments besides B♭ Clarinet, but also for entire ensemble or orchestral pieces.

Figure 7.1: The main page for the website with user controls.

Figure 7.2: The table output of complexity data on the website.

Instrument Name	Total Complexity Score	Toughest Measure Number	Toughest Measure Complexity	Note Complexity	Interval Complexity
English Horn	3719	39	1084	1798	1921
Flutes	8272	37	1415	4153	4118
Harp	5872	26	1053	2486	3385
Oboes	6025	39	1655	2935	3089
Piccolo	2829	39	944	1457	1371
Timpani	16348	27	7771	14660	1687
Trumpets in C	201	38	109	103	97
Tuba	360	38	83	197	162
Viola	12852	27	1862	5610	7241
Violin I	4330	27	561	2091	2239

Showing 11 to 20 of 22 entries   Previous   1   2   3   Next

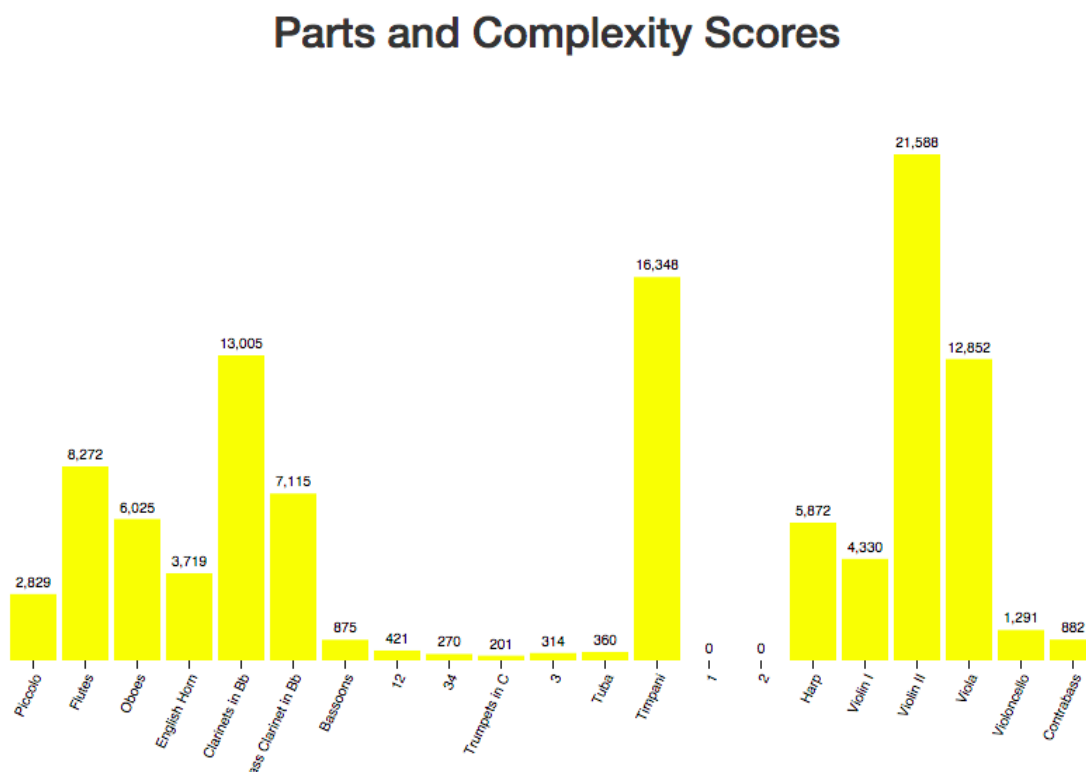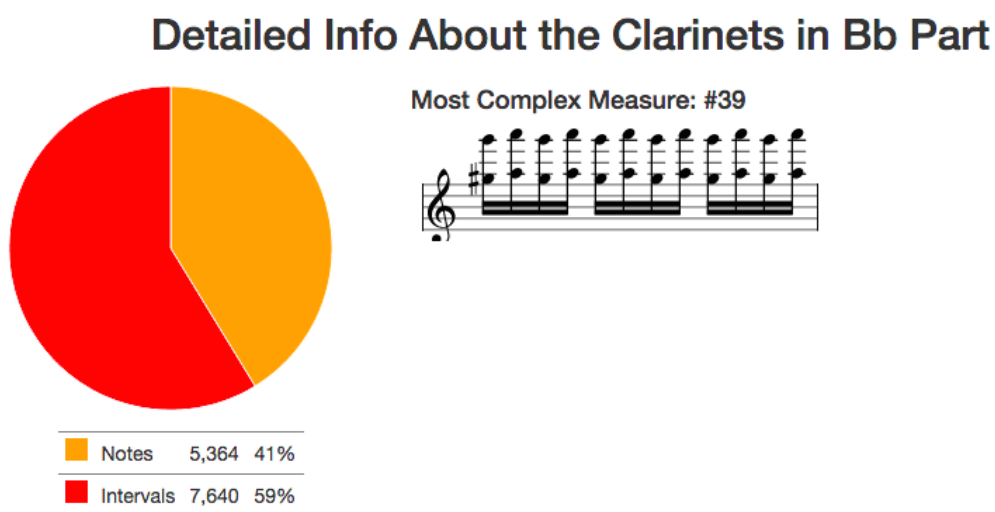Figure 7.3: The graph output of complexity data on the website.

Parts and Complexity Scores

Figure 7.4: The detailed output for one part on the website.

Detailed Info About the Clarinets in Bb Part

Chapter 8

Future Work

The following Subsections address planned future work in a number of different directions, including expanding complexity parameters in 8.1, mapping parts and instruments in 8.2, understanding complexity scores in 8.3, integrating scores with sources in 8.4, demonstrating this work in 8.5, including more input formats in 8.6, measuring complexity from physiology in 8.7, and broadening the approach in 8.8.

8.1   Expanding Instrument Complexity Parameters

As mentioned in 6.2, the current complexity parameters lack validation. One way to improve their accuracy is thus to gather a consensus from those with a stake in this complexity measurement, such as experts, performers, and educators. I have already begun the process of surveying these people for their opinions on complexity parameters for B♭ Clarinet. However, a notable direction for future work is to expand this survey and indeed the viability of the overall complexity score out to other instruments.

The reference implementation in its current state can run effectively for any instrument, and any complexity parameters can be utilized. In this way it is currently agnostic to what instruments are being played in the piece. I do not attempt here to generate complexity parameters for other instruments (besides B♭ Clarinet) both for brevity and for accuracy. If the approach of surveying stakeholders is practical and viable, then we must expand to utilize it further. If surveying will not work, then we must find another approach to validate complexity parameters.

The alternative to this arranged validation is to allow users to supply their own parameters each time without any set standard. Although this workaround would seem necessary to get customized complexity scores for a given playing level of multiple instruments, it requires potentially specifying all the parameters for an entire orchestra. It is unclear at this point

how necessary this feature is compared to a more streamlined process for using the program.

8.2 Mapping Separate Parts and Instruments

A related tangential point of future work to gathering more instrument complexity parameters is to separate out different parts and instruments so that each can have its own complexity parameters applied. As mentioned before, the reference implementation does not currently differentiate one part from another. Each part in a musical piece has the complexity parameters applied to it equally (as if each part in the piece was for the same instrument and playing level). It is simple enough to separate out these parts, but the problem becomes matching them to standard complexity parameters.

Within an orchestral or similar piece, the different parts can be named a variety of ways by referencing instruments, players, sections, etc. These can be specific or vague, such as "1st Chair B♭ Clarinet" and "High Brass", respectively. There is no widely accepted, practical standard for how these are specified.

However, we can still attempt to perform this matching. One trivial approach would be to simply keep track of all possible part names the reference implementation ever encounters and periodically update a table that matches the part name in the piece to a set of complexity parameters. A more elegant approach could be to apply some natural language processing techniques to attempt to automatically match the two or, at worst, provide a small subset of alternatives that a user could choose from when running the tool. Yet another alternative could simply be to allow the user to choose exactly which complexity parameters to use for each part at every run. Each of these has its drawbacks in efficiency, usability, and expressiveness. Nonetheless, this problem looms as we move towards more complexity parameters and remains an open area of research that we plan to address.

8.3 Understanding Complexity Scores

One potential issue users face is that to understand a complexity score requires referencing other complexity scores. For instance, the score of 1000 for some piece (or part equivalently) X cannot be meaningfully interpreted without knowing what that piece is, such as a Mozart symphony, or knowing other scores of pieces, such as a Beethoven symphony scoring only 500 so piece X is twice as complex.

One possible solution for this problem is to track the names of pieces (and parts) along with their complexity score in a database. Then, upon scoring some piece, the reference implementation can also output the closest scores of well-known pieces, thus providing a reference point. While keeping track of these scores in a database may also help speed up

computation by not repeatedly calculating the score for the same piece over and over, this introduces much more overhead if users are allowed to input any complexity parameters.

Another solution to this problem would be to scale the score down to some range of numbers, such as 0 to 100. Scaling would mean that no piece could have a complexity score greater than 100 or less than 0. While this may not directly solve the problem of understanding the complexity score, it does bound the possible scores and thus provide its own reference point.

This approach may be more useful for competition rankings so that the complexity score can be easily factored into the score for a performance. However, there is no simple way to scale all complexity scores down without knowing what would receive the highest possible score. Knowing the potential highest score becomes an even more difficult task with user supplied complexity parameters that can wholly change the resulting complexity scores. We are nevertheless investigating this currently to see how we could at least limit scores to some arbitrarily high value and scale based on that.

8.4 Integrating Complexity Scores with Sources

To make musiplectics more accessible, one direction for future work lies in integrating the complexity score into various music applications. For instance, we envision notation applications, such as Finale Notepad [58] or Sibelius [11], displaying the complexity score of a piece as it is being written so composers can readily see exactly how complex their piece is numerically. Similarly, we would like to partner with music sharing websites, such as International Music Score Library Project [75], that allow users to search, view, and download pieces of music. We envision the complexity score of a piece being available before downloading, or more importantly purchasing, the piece so as to give users some reassurance of what they are getting. This could also lead to users being able to search pieces by their complexity score (if they were pre-computed and stored somewhere) should the user need to find a piece to match his or her playing level.

8.5 Demonstrating Musiplectics for Art Exposé

Virginia Tech's Institute for Creativity, Arts, and Technology (ICAT) annually holds an exposé for new research and works of art that combine technology with the arts in a variety of ways. ICAT Day as it is called will serve as a formal demonstration for many potential stakeholders to see Musiplectics in action and get to interact with the reference implementation. In order to facilitate this interaction with the reference implementation, changes were made to the frontend web interface, turning it into a sort of game for spectators to play.

As shown in Figure 8.1, the game format of the web page allows users to assess the music piece in PDF format alongside inputting a guess for the most difficult measure in the piece.

On the left, an iframe displays the currently selected music piece in PDF format. On the right, a directed acyclic graph (DAG) shows each part within the piece connected to a central node for the name of the selected piece. Users may select a node in the graph, then supply their guess for the most difficult measure for that specific part. A user can specify guesses for every part or just some subset of the parts, and then click "Get Complexity Score" as usual to send the request.

The request sent to the backend is identical to before, but once the response is returned, the most difficult measure for each part is compared to the user's guess. Afterwards, output such as in Figure 8.2 is shown via an alert over top of the results.

During ICAT Day, I plan to informally assess the accuracy of the musiplectics approach compared to the expectations of potential stakeholders. While no formal data collection or analysis will be performed, this demonstration should provide critical insights and user suggestions. These can further allow us to improve the usability of the web page and accuracy of the overall approach.

8.6   Including More Input Formats

Yet another direction for future work is the expansion of the formats that can be input in general. At the moment MusicXML files are of course supported, and PDF files can be manually translated to MusicXML via OCR. As mentioned above, this process is not yet mature enough to be run automatically, but OCR in general can operate on many other formats, such as PNG, TIFF, and BMP images, so it would be trivial to expand to allow those inputs.

Beyond what OCR can handle, even more inputs can be translated into MusicXML. Software, such as NotationSoft [66], can translate event-driven MIDI files [10] into MusicXML. This type of translation could bring a wealth more of input since a large amount of music literature is stored in this fashion. In fact, efforts such as [29] are already taking place to digitize a large amount of publicly available music into MIDI. An extension to incorporate translated MIDI files would greatly expand the applicability of musiplectics.
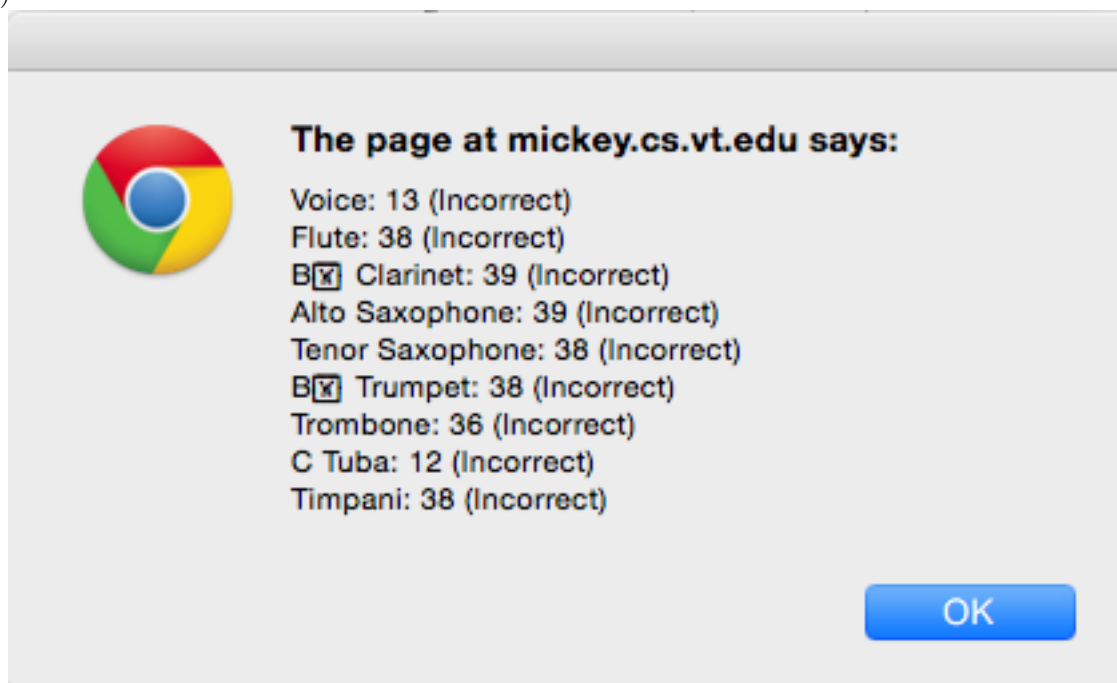
8.7   Measuring Complexity From Physiological Signals

Some prior work has focused on measuring physiological characteristics, especially in the field of human-computer interaction. Often these measurements have been used as indicators of emotional state [53]. In relation to music, these measurements have been used both to gauge an audience's reaction to a piece of music [25] [37] as well as a means for people to play their own music [54] [82]. We would like to leverage these types of works to incorporate

Figure 8.1: The modified input for the game version of the website.

Figure 8.2: The potential output for the game version of the website (using Google Chrome [39]).

physiological measurements and biofeedback as a means of forming or validating complexity scores.

The knowledge of a performer's relative playing proficiency combined with their basic physiological traits while playing a piece could form a model for extrapolating the cognitive load or mental complexity being endured. This method can become a reliable means of parameterizing our system for individual players.

Additionally, future research could compare the physiological measures of mental complexity being endured to the expected complexity of the piece of music at any given time. It is our belief currently that performers tend to become more anxious (and thus endure higher levels of complexity and stress) directly before a particularly complex part of a song. Thus, we would expect to find a flip-flop pattern where mental complexity can be measured to increase first, and then the complexity score of the piece thus far will increase second, while the measured mental complexity then decreases (assuming the performer plays the selection of music correctly).

8.8 Broadening Musiplectics to Other Plectics

Past work in deciphering the human genome showed the promise of computers in being able to handle great levels of complexity and yield insightful results [1] [86]. While musiplectics does not require as much computational power as a task like deciphering human genes, we still draw parallels to the overall approach of decomposing something exceedingly complex for human and allowing computers to aggregate the small parts, thus providing valuable insights.

To that end, another potential area for future work in expanding the core approach of musiplectics to other areas of complexity measurement. Any well-defined task can be decompose into its principle parts. In music, this means separating a piece of music into the elements we described previously. As discussed above, in program performance estimation, this means separating source code into the low-level instructions that execute on a given machine. Analogously, certain activities could be defined in a similar fashion, such as cooking. In cooking, this would mean decomposing a dish into the steps and ingredients required to create the dish as well as assigning values or weights to each step in the process. This process is already performed to some degree to estimate the time required to create a dish, but it could similarly be extended to estimate the overall cost or the difficulty of preparing the dish.

There are many other genres of plectics that could apply this approach. It is not yet clear how formally defined a process must be to effectively make use of this approach, but it is our belief that tasks specified by both natural and formal languages [51] [42] [52] could benefit from these types of insights.

Chapter 9

Conclusions

This paper presented musiplectics, a new computational paradigm, that systematically evaluates the relative difficulty of music scores, thus benefiting educators and performers. Our hope is that musiplectics can improve the landscape of assessment of music scores. The work presented here unveils our first steps towards an objective and automatic approach to computing the complexity of music pieces. The contributions of this paper include our model for computing complexity scores and its concrete realization in our reference implementation. The automatically computed complexity scores of many well-known pieces and their respective manual grades demonstrate the promise of musiplectics to alleviate the burden of music complexity rankings, freeing musicians for more creative pursuits. In addition, future work directions present many exciting opportunities to apply computing to solve important problem in music arts.

Bibliography

[1] Initial sequencing and analysis of the human genome. *Nature*, 409(6822):860–921, 02 2001.

[2] ACM Digital Library. `http://dl.acm.org/`, Dec. 2014.

[3] Google Scholar. `http://scholar.google.com/`, Dec. 2014.

[4] J. Allali, P. Ferraro, P. Hanna, C. Iliopoulos, and M. Robine. Toward a general framework for polyphonic comparison. *Fundam. Inf.*, 97(3):331–346, Aug. 2009.

[5] S. Allen, V. Graupera, and L. Lundrigan. Phonegap. In *Pro Smartphone Cross-Platform Development*, pages 131–152. Apress, 2010.

[6] Apache Xerces. The Apache Xerces Project. `http://xerces.apache.org/index.html`, 1 2013.

[7] Apple Inc. OS X Mavericks 10.9.1 Update. `https://support.apple.com/kb/dl1715?locale=en_US`, 12 2013.

[8] Apple Inc. Mac Mini. `https://www.apple.com/mac-mini/`, 2015.

[9] J. Arnold. *Clarinet Solos*. Amsco Music Pub. Co, New York, 1939.

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

[11] Avid Technology Inc. Sibelius. `http://www.avid.com/US/products/sibelius`, 3 2015.

[12] I. Bartenieff. *Body movement: Coping with the environment*. Psychology Press, 1980.

[13] H. Bitteur. Audiveris. `https://audiveris.kenai.com/`, 1 2013.

[14] S. Blanchard and K. Cameron. SeeMore. *Leonardo*, 47(4):414–415, 2014.

[15] J. Bloch. Effective Java (the Java Series). 2008.

[16] B. Bos, H. W. Lie, C. Lilley, and I. Jacobs. Cascading style sheets, level 2 CSS2 specification.

[17] M. Bostock. D3 data-driven documents. `http://d3js.org/`, 2013.

[18] M. J. Burke and M. C. Gridley. Musical preferences as a function of stimulus complexity and listeners' sophistication. *Perceptual and Motor Skills*, 71(2):687–690, 1990.

[19] D. Byrd. Music-notation searching and digital libraries. In *Proceedings of the 1st ACM/IEEE-CS Joint Conference on Digital Libraries*, JCDL '01, pages 239–246, New York, NY, USA, 2001. ACM.

[20] D. Byrd and M. Schindele. Prospects for improving OMR with multiple recognizers. In *ISMIR*, pages 41–46, 2006.

[21] A. Camurri, S. Hashimoto, M. Ricchetti, A. Ricci, K. Suzuki, R. Trocca, and G. Volpe. Eyesweb: Toward gesture and affect recognition in interactive dance and music systems. *Computer Music Journal*, 24(1):57–69, 2000.

[22] A. Camurri, B. Mazzarino, M. Ricchetti, R. Timmers, and G. Volpe. Multimodal analysis of expressive gesture in music and dance performances. In *Gesture-based communication in human-computer interaction*, pages 20–39. Springer, 2004.

[23] M. A. Casey, R. Veltkamp, M. Goto, M. Leman, C. Rhodes, and M. Slaney. Content-based music information retrieval: Current directions and future challenges. *Proceedings of the IEEE*, 96(4):668–696, 2008.

[24] Z. Cataltepe, Y. Yaslan, and A. Sonmez. Music genre classification using MIDI and audio features. *EURASIP J. Appl. Signal Process.*, 2007(1):150–150, Jan. 2007.

[25] C. Chapados and D. J. Levitin. Cross-modal interactions in the experience of musical performances: Physiological correlates. *Cognition*, 108(3):639–651, 2008.

[26] M. M. Cheppudira. vexflow. `https://github.com/0xfe/vexflow`, 2010.

[27] D. Chi, M. Costa, L. Zhao, and N. Badler. The EMOTE model for effort and shape. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 173–182, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[28] S.-C. Chiu and M.-S. Chen. A study on difficulty level recognition of piano sheet music. In *Proceedings of the 2012 IEEE International Symposium on Multimedia*, ISM '12, pages 17–23, Washington, DC, USA, 2012. IEEE Computer Society.

[29] G. S. Choudhury, M. Droetboom, T. DiLauro, I. Fujinaga, and B. Harrington. Optical music recognition system within a large-scale digitization project. In *ISMIR*, 2000.

[30] D. Crockford. The application/json media type for JavaScript object notation (JSON). 2006.

[31] M. S. Cuthbert, C. Ariza, and L. Friedland. Feature extraction and machine learning on symbolic music using the music21 toolkit. In *ISMIR*, pages 387–392, 2011.

[32] J. S. Downie. Music information retrieval. *Annual review of information science and technology*, 37(1):295–340, 2003.

[33] B. Edmonds. What is complexity?-the philosophy of complexity per se with application to some examples in evolution. *The evolution of complexity*, 1995.

[34] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design patterns: elements of reusable object-oriented software*. Pearson Education, 1994.

[35] J. J. Garrett et al. Ajax: A new approach to web applications. 2005.

[36] R. Ghatol and Y. Patel. *Beginning PhoneGap: mobile web framework for JavaScript and HTML5*. Apress, Berkely, CA, USA, 1st edition, 2012.

[37] P. Gomez and B. Danuser. Affective and physiological responses to environmental noises and music. *International Journal of psychophysiology*, 53(2):91–103, 2004.

[38] M. Good et al. MusicXML: An internet-friendly format for sheet music. In *XML Conference and Expo*, pages 03–04. Citeseer, 2001.

[39] Google. Chrome. `http://www.google.com/chrome/`, 2015.

[40] J. Gosling. *The Java language specification*. Addison-Wesley Professional, 2000.

[41] A. H. Guest. *Labanotation: the system of analyzing and recording movement*. Psychology Press, 2005.

[42] M. A. Harrison. *Introduction to formal language theory*. Addison-Wesley Longman Publishing Co., Inc., 1978.

[43] H. Heijink and R. Meulenbroek. On the complexity of classical guitar playing: Functional adaptations to task constraints. *Journal of motor behavior*, 34(4):339–351, 2002.

[44] R. Heyduk. Rated preference for musical compositions as it relates to complexity and exposure frequency. *Perception and Psychophysics*, 17(1):84–90, 1975.

[45] E. Holder. Cloud twin: interactive cross-platform replay for mobile applications. In *Proceedings of the 2013 companion publication for conference on Systems, programming, & applications: software for humanity*, pages 119–120. ACM, 2013.

[46] E. Holder. MusicScoring. `https://github.com/xwsxethan/MusicScoring`, 2 2015.

[47] E. Holder, E. Shah, M. Davoodi, and E. Tilevich. Cloud twin: Native execution of Android applications on the Windows Phone. In *Automated Software Engineering (ASE), 2013 IEEE/ACM 28th International Conference on*, pages 598–603. IEEE, 2013.

[48] E. G. Holder. Native execution of mobile applications across platforms. Honors Bachelor's Thesis, Virginia Polytechnic Institute and State University, May 2014.

[49] A. Hutchinson. Labanotation. *Dance Books*, 1996.

[50] JSON Simple. json-simple. `https://code.google.com/p/json-simple/`, 2 2012.

[51] H. Kamp and U. Reyle. *From discourse to logic: Introduction to modeltheoretic semantics of natural language, formal logic and discourse representation theory*. Number 42. Springer Science & Business Media, 1993.

[52] R. J. Kate, Y. W. Wong, and R. J. Mooney. Learning to transform natural to formal languages. In *Proceedings of the National Conference on Artificial Intelligence*, volume 20, page 1062. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005.

[53] R. Knapp, J. Kim, and E. Andr. Physiological signals and their use in augmenting emotion recognition for humanmachine interaction. In R. Cowie, C. Pelachaud, and P. Petta, editors, *Emotion-Oriented Systems*, Cognitive Technologies, pages 133–159. Springer Berlin Heidelberg, 2011.

[54] R. B. Knapp and H. S. Lusted. A bioelectric controller for computer music applications. *Computer Music Journal*, 14(1):pp. 42–47, 1990.

[55] D. E. Knuth. The complexity of songs. *Commun. ACM*, 27(4):344–346, Apr. 1984.

[56] C.-Y. Liou, T.-H. Wu, and C.-Y. Lee. Modeling complexity in musical rhythm. *Complex.*, 15(4):19–30, Mar. 2010.

[57] S. T. Madsen and G. Widmer. Music complexity measures predicting the listening experience. *Entropy*, 1(4):1, 2006.

[58] Makemusic Inc. Finale Notepad. `http://www.finalemusic.com/products/finale-notepad/`, 3 2015.

[59] Makemusic Inc. MusicXML. `http://www.musicxml.com/`, 3 2015.

[60] V. Maletic. *Body-space-expression: The development of Rudolf Laban's movement and dance concepts*, volume 75. Walter de Gruyter, 1987.

[61] M. Mauch and M. Levy. Structural change on multiple time scales as a correlate of musical complexity. In *ISMIR*, pages 489–494, 2011.

[62] MuseScore BVBA. MuseScore share your sheet music. `https://musescore.com/`, 2015.

[63] MusicTheory.net LLC. MusicTheory.net. `http://www.musictheory.net/`, 2015.

[64] V. Nguyen, S. Deeds-Rubin, T. Tan, and B. Boehm. A SLOC counting standard. In *COCOMO II Forum*, volume 2007, 2007.

[65] A. C. North and D. J. Hargreaves. Can music move people? the effects of musical complexity and silence on waiting time. *Environment and Behavior*, 31(1):136–149, 1999.

[66] Notation Software Germany. Notation Composer. `http://www.notation.com/NotationComposer.php`, 2014.

[67] NYSSMA New York State School Music Association. NYSSMA New York State School Music Association. `http://www.nyssma.org/`, 2015.

[68] Oracle Corporation. MySQL the world's most popular open source database. `https://www.mysql.com/`, 2015.

[69] M. G. Orr and S. Ohlsson. The relationship between musical complexity and liking in jazz and bluegrass. *Psychology of Music*, 29(2):108–127, 2001.

[70] Otto and Thornton. Bootstrap. `http://getbootstrap.com/`, 2015.

[71] C. Y. Park. Predicting program execution times by analyzing static and dynamic program paths. *Real-Time Syst.*, 5(1):31–62, Mar. 1993.

[72] R. M. Parry. Musical complexity and top 40 chart performance. 2004.

[73] B. Pearson. *Standard of excellence : comprehensive band method, B clarinet*. Neil A. Kjos Music Co, San Diego, Calif, 1993.

[74] J. Pressing. Cognitive complexity and the structure of musical patterns. In *Proceedings of the 4th Conference of the Australasian Cognitive Science Society*, 1999.

[75] Project Petrucci LLC. International music score library project. `http://imslp.org/`, 2015.

[76] Royal Conservatory Music Development Program. Clarinet syllabus. `https://www.musicdevelopmentprogram.org/sites/default/files/files/S42_Clarinet%20Syl_MDP_2014_online_SECURED.pdf`, 2014.

[77] E. D. Scheirer, R. B. Watson, and B. L. Vercoe. On the perceived complexity of short musical segments. In *Proceedings of the 2000 International Conference on Music Perception and Cognition*. Citeseer, 2000.

[78] Society for Music Theory. MTO a Journal for the Society for Music Theory. `http://www.mtosmt.org/index.php`, 2015.

[79] SpryMedia Ltd. Datatables table plug-in for jQuery. `https://www.datatables.net/`, 2015.

[80] L. Steck and P. Machotka. Preference for musical complexity: Effects of context. *Journal of Experimental Psychology: Human Perception and Performance*, 1(2):170, 1975.

[81] S. Streich. *Automatic Characterization of Music Complexity: a multifaceted approach.* PhD thesis, Citeseer, 2006.

[82] A. Tanaka and R. B. Knapp. Multimodal interaction in music using the electromyogram and relative position sensing. In *Proceedings of the 2002 Conference on New Interfaces for Musical Expression*, NIME '02, pages 1–6, Singapore, Singapore, 2002. National University of Singapore.

[83] The Apache Software Foundation. Apache HTTP Server Project. `http://httpd.apache.org/`, 2015.

[84] The jQuery Foundation. jQuery. `https://jquery.com/`, 2015.

[85] V. Tiwari, S. Malik, A. Wolfe, and M. T.-C. Lee. Instruction level power analysis and optimization of software. *J. VLSI Signal Process. Syst.*, 13(2-3):223–238, Aug. 1996.

[86] J. C. Venter, M. D. Adams, E. W. Myers, P. W. Li, R. J. Mural, G. G. Sutton, H. O. Smith, M. Yandell, C. A. Evans, R. A. Holt, J. D. Gocayne, P. Amanatides, R. M. Ballew, D. H. Huson, J. R. Wortman, Q. Zhang, C. D. Kodira, X. H. Zheng, L. Chen, M. Skupski, G. Subramanian, P. D. Thomas, J. Zhang, G. L. Gabor Miklos, C. Nelson, S. Broder, A. G. Clark, J. Nadeau, V. A. McKusick, N. Zinder, A. J. Levine, R. J. Roberts, M. Simon, C. Slayman, M. Hunkapiller, R. Bolanos, A. Delcher, I. Dew, D. Fasulo, M. Flanigan, L. Florea, A. Halpern, S. Hannenhalli, S. Kravitz, S. Levy, C. Mobarry, K. Reinert, K. Remington, J. Abu-Threideh, E. Beasley, K. Biddick, V. Bonazzi, R. Brandon, M. Cargill, I. Chandramouliswaran, R. Charlab, K. Chaturvedi, Z. Deng, V. D. Francesco, P. Dunn, K. Eilbeck, C. Evangelista, A. E. Gabrielian, W. Gan, W. Ge, F. Gong, Z. Gu, P. Guan, T. J. Heiman, M. E. Higgins, R.-R. Ji, Z. Ke, K. A. Ketchum, Z. Lai, Y. Lei, Z. Li, J. Li, Y. Liang, X. Lin, F. Lu, G. V. Merkulov, N. Milshina, H. M. Moore, A. K. Naik, V. A. Narayan, B. Neelam, D. Nusskern, D. B. Rusch, S. Salzberg, W. Shao, B. Shue, J. Sun, Z. Y. Wang, A. Wang, X. Wang, J. Wang, M.-H. Wei, R. Wides, C. Xiao, C. Yan, A. Yao, J. Ye, M. Zhan, W. Zhang, H. Zhang, Q. Zhao, L. Zheng, F. Zhong, W. Zhong, S. C. Zhu, S. Zhao, D. Gilbert, S. Baumhueter, G. Spier, C. Carter, A. Cravchik, T. Woodage, F. Ali, H. An, A. Awe, D. Baldwin, H. Baden, M. Barnstead, I. Barrow, K. Beeson, D. Busam, A. Carver, A. Center, M. L. Cheng, L. Curry, S. Danaher, L. Davenport, R. Desilets, S. Dietz, K. Dodson, L. Doup, S. Ferriera, N. Garg, A. Gluecksmann, B. Hart, J. Haynes, C. Haynes, C. Heiner, S. Hladun, D. Hostin, J. Houck, T. Howland, C. Ibegwam, J. Johnson, F. Kalush, L. Kline, S. Koduru, A. Love, F. Mann, D. May, S. McCawley, T. McIntosh, I. McMullen, M. Moy, L. Moy, B. Murphy, K. Nelson, C. Pfannkoch, E. Pratts, V. Puri, H. Qureshi, M. Reardon, R. Rodriguez, Y.-H. Rogers, D. Romblad, B. Ruhfel, R. Scott, C. Sitter, M. Smallwood, E. Stewart, R. Strong, E. Suh, R. Thomas, N. N. Tint, S. Tse, C. Vech, G. Wang,

J. Wetter, S. Williams, M. Williams, S. Windsor, E. Winn-Deen, K. Wolfe, J. Zaveri, K. Zaveri, J. F. Abril, R. Guig, M. J. Campbell, K. V. Sjolander, B. Karlak, A. Kejariwal, H. Mi, B. Lazareva, T. Hatton, A. Narechania, K. Diemer, A. Muruganujan, N. Guo, S. Sato, V. Bafna, S. Istrail, R. Lippert, R. Schwartz, B. Walenz, S. Yooseph, D. Allen, A. Basu, J. Baxendale, L. Blick, M. Caminha, J. Carnes-Stine, P. Caulk, Y.-H. Chiang, M. Coyne, C. Dahlke, A. D. Mays, M. Dombroski, M. Donnelly, D. Ely, S. Esparham, C. Fosler, H. Gire, S. Glanowski, K. Glasser, A. Glodek, M. Gorokhov, K. Graham, B. Gropman, M. Harris, J. Heil, S. Henderson, J. Hoover, D. Jennings, C. Jordan, J. Jordan, J. Kasha, L. Kagan, C. Kraft, A. Levitsky, M. Lewis, X. Liu, J. Lopez, D. Ma, W. Majoros, J. McDaniel, S. Murphy, M. Newman, T. Nguyen, N. Nguyen, M. Nodell, S. Pan, J. Peck, M. Peterson, W. Rowe, R. Sanders, J. Scott, M. Simpson, T. Smith, A. Sprague, T. Stockwell, R. Turner, E. Venter, M. Wang, M. Wen, D. Wu, M. Wu, A. Xia, A. Zandieh, and X. Zhu. The sequence of the human genome. *Science*, 291(5507):1304–1351, 2001.

[87] Virginia Band and Orchestra Directors Association. Virginia band and orchestra directors association. `http://www.vboda.org/`, 2015.

[88] Visiv. SharpEye Music Scanning. `http://www.visiv.co.uk/`, 2015.

[89] H. Voxman. *Concert and Contest Collection for Bb Clarinet: Solo Part.* Rubank Publications, Chicago, Illinois, 1992.

[90] L. R. Williams. Effect of music training and musical complexity on focus of attention to melody or harmony. *Journal of Research in Music Education*, 53(3):210–221, 2005.

[91] J. M. Wing. Computational thinking. *Communications of the ACM*, 49(3):33–35, 2006.

Appendix A

Complexity Parameters Example

```
 1 <?xml version="1.0" encoding="UTF-8" standalone="no"?>
 2 <levels>
 3   <notes>
 4     <note-value>
 5       <range>G3-G#4</range>
 6       <difficulty>
 7          1
 8       </difficulty>
 9     </note-value>
10     <note-value>
11       <range>
12          A4
13       </range>
14       <difficulty>
15          2
16       </difficulty>
17     </note-value>
18     <note-value>
19       <range>Bb4-C5</range>
20       <difficulty>
21          5
22       </difficulty>
23     </note-value>
24     <note-value>
25       <range>
26          C#5-C10
27       </range>
28       <difficulty>
29          10
30       </difficulty>
31     </note-value>
32   </notes>
33   <intervals>
34     <interval>
```

```
35      <difference>
36      1
37      </difference>
38      <location1>
39      -1
40      </location1>
41      <location2>
42      -1
43      </location2>
44      <difficulty>
45      1
46      </difficulty>
47    </interval>
48    <interval>
49      <difference>
50      2
51      </difference>
52      <location1>G3-G#4</location1>
53      <location2>G3-G#4</location2>
54      <difficulty>
55      2
56      </difficulty>
57    </interval>
58    <interval>
59      <difference>
60      3
61      </difference>
62      <location1>G3-G#4</location1>
63      <location2>G3-G#4</location2>
64      <difficulty>
65      3
66      </difficulty>
67    </interval>
68    <interval>
69      <difference>
70      4
71      </difference>
72      <location1>G3-G#4</location1>
73      <location2>G3-G#4</location2>
74      <difficulty>
75      4
76      </difficulty>
77    </interval>
78    <interval>
79      <difference>
80      5
81      </difference>
82      <location1>G3-G#4</location1>
83      <location2>G3-G#4</location2>
84      <difficulty>
85      5
```

```
86      </difficulty>
87    </interval>
88    <interval>
89      <difference>
90      0
91      </difference>
92      <location1>G3-G#4</location1>
93      <location2>A4-C5</location2>
94      <difficulty>
95      8
96      </difficulty>
97    </interval>
98    <interval>
99      <difference>
100     0
101     </difference>
102     <location1>A4-C5</location1>
103     <location2>
104     C#5-C10
105     </location2>
106     <difficulty>
107     10
108     </difficulty>
109   </interval>
110   <interval>
111     <difference>
112     6
113     </difference>
114     <location1>
115     -1
116     </location1>
117     <location2>
118     -1
119     </location2>
120     <difficulty>
121     9
122     </difficulty>
123   </interval>
124   <interval>
125     <difference>
126     7
127     </difference>
128     <location1>
129     -1
130     </location1>
131     <location2>
132     -1
133     </location2>
134     <difficulty>
135     9
136     </difficulty>
```

```
137    </interval>
138    <interval>
139      <difference>
140      8
141      </difference>
142      <location1>
143      -1
144      </location1>
145      <location2>
146      -1
147      </location2>
148      <difficulty>
149      9
150      </difficulty>
151    </interval>
152    <interval>
153      <difference>
154      -8
155      </difference>
156      <location1>
157      -1
158      </location1>
159      <location2>
160      -1
161      </location2>
162      <difficulty>
163      10
164      </difficulty>
165    </interval>
166   </intervals>
167   <dynamics>
168     <mf>
169      1.0
170     </mf>
171     <mp>
172      1.0
173     </mp>
174     <f>
175      1.1
176     </f>
177     <ff>
178      1.2
179     </ff>
180     <p>
181      1.3
182     </p>
183     <pp>
184      1.5
185     </pp>
186   </dynamics>
187   <tempo>
```

```
188      1.0
189    </tempo>
190    <key-signatures>
191      <key-value>
192        <key>
193          C
194        </key>
195        <difficulty>
196          1.0
197        </difficulty>
198      </key-value>
199      <key-value>
200        <key>
201          G
202        </key>
203        <difficulty>
204          1.1
205        </difficulty>
206      </key-value>
207      <key-value>
208        <key>
209          D
210        </key>
211        <difficulty>
212          1.1
213        </difficulty>
214      </key-value>
215      <key-value>
216        <key>
217          A
218        </key>
219        <difficulty>
220          1.2
221        </difficulty>
222      </key-value>
223      <key-value>
224        <key>
225          E
226        </key>
227        <difficulty>
228          1.3
229        </difficulty>
230      </key-value>
231      <key-value>
232        <key>
233          B
234        </key>
235        <difficulty>
236          1.4
237        </difficulty>
238      </key-value>
```

```
239  <key-value>
240    <key>
241      F#
242    </key>
243    <difficulty>
244      1.5
245    </difficulty>
246  </key-value>
247  <key-value>
248    <key>
249      C#
250    </key>
251    <difficulty>
252      1.6
253    </difficulty>
254  </key-value>
255  <key-value>
256    <key>
257      F
258    </key>
259    <difficulty>
260      1.1
261    </difficulty>
262  </key-value>
263  <key-value>
264    <key>
265      Bb
266    </key>
267    <difficulty>
268      1.1
269    </difficulty>
270  </key-value>
271  <key-value>
272    <key>
273      Eb
274    </key>
275    <difficulty>
276      1.2
277    </difficulty>
278  </key-value>
279  <key-value>
280    <key>
281      Ab
282    </key>
283    <difficulty>
284      1.3
285    </difficulty>
286  </key-value>
287  <key-value>
288    <key>
289      Db
```

```
290      </key>
291      <difficulty>
292        1.4
293      </difficulty>
294    </key-value>
295    <key-value>
296      <key>
297        Gb
298      </key>
299      <difficulty>
300        1.5
301      </difficulty>
302    </key-value>
303    <key-value>
304      <key>
305        Cb
306      </key>
307      <difficulty>
308        1.6
309      </difficulty>
310    </key-value>
311  </key-signatures>
312  <articulations>
313    <normal>
314      1.0
315    </normal>
316    <staccato>
317      1.2
318    </staccato>
319    <tenuto>
320      1.2
321    </tenuto>
322    <slur>
323      0.5
324    </slur>
325    <accent>
326      1.1
327    </accent>
328    <strong-accent>
329      1.4
330    </strong-accent>
331  </articulations>
332 </levels>
```

Listing A.1: The full xml file of complexity parameters for beginner B♭ Clarinet.

Appendix B

External Survey

Default Question Block

To complete this survey, please first provide your contact information so we can record your response and indicate your preference for the complexity measurement methods suggested. We ask for some identifying information below to prevent duplicate responses and comply with institutional review board standards, but no identifying information will be kept with your response.

This survey will close at exactly 11:55pm on March 31st, and no late requests will be considered. Please submit your response by that time. Additionally, please do not attempt to submit multiple responses to this survey.

Email Address

Last Name

First Name

Are you over the age of 18 years old?

○ Yes

○ No

This survey will ask you about your preferences for the complexity measurements methods already implemented. The basic methodology currently is to set a point value between 1 and 10 for each note and each interval. Every occurrence of that note or interval receives that amount of points. Each point amount is then multiplied by a point value greater than 0 for the articulation, dynamic, duration (combination of tempo and note length), and key signature associated with the note or interval. Four total multipliers are applied (one each for articulations, dynamics, duration, and key signature). All the notes and intervals with associated multipliers are added together, and the total is the overall complexity measurement given to the piece of music.

The idea here is to reduce both the mental or cognitive complexity as well as the mechanical motions down to a single number for all of the musical elements listed above (notes, intervals, articulations, dynamics, duration, and key signature). By combining these simple building blocks, we can generate an accurate measurement for the overall complexity of a large piece of music.

The following questions will ask you about your overall thoughts on the methodology presented.

Do you agree with the overall methodology presented to measure the complexity of pieces of music?

For reference, that methodology is included again below:

The basic methodology currently is to set a point value between 1 and 10 for each note and each interval. Every occurrence of that note or interval receives

that amount of points. Each point amount is then multiplied by a point value greater than 0 for the articulation, dynamic, duration (combination of tempo and note length), and key signature associated with the note or interval. Four total multipliers are applied (one each for articulations, dynamics, duration, and key signature). All the notes and intervals with associated multipliers are added together, and the total is the overall complexity measurement given to the piece of music.

Strongly Agree	Agree	Neither Agree Nor Disagree	Disagree	Strongly Disagree
○	○	○	○	○

Do you agree with assigning point values to notes and intervals and multiplier values to articulations, dynamics, duration, and key signature?

Strongly Agree	Agree	Neither Agree Nor Disagree	Disagree	Strongly Disagree
○	○	○	○	○

If the methodology was changed such that articulations, dynamics, duration, and key signature each received point values instead of multiplier values, and each of these were added into the overall measurement along with notes and intervals, would you feel that the complexity measurement would become more or less accurate to your experience?

Much More Accurate	Somewhat More Accurate	Neither More Nor Less Accurate	Somewhat Less Accurate	Much Less Accurate
○	○	○	○	○

Do you agree that the complexity of both notes and intervals is affected by the articulations, dynamics, duration, and key signature or does this complexity only affect individual notes?

Strongly Agree (Both Notes and Intervals)	Somewhat Agree (Both Notes and Intervals)	Neither Agree Nor Disagree	Somewhat Agree (Only Notes)	Strongly Agree (Only Notes)
○	○	○	○	○

What do you like most about the methodology presented?

What do you like least about the methodology presented?

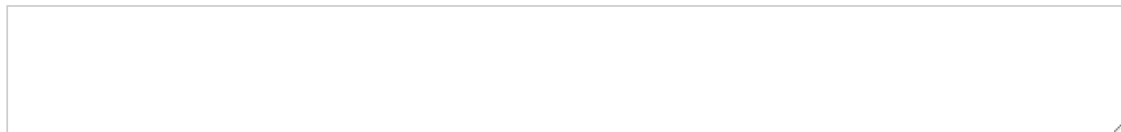The following questions will ask about your preferences for the point values of notes.

We have set the following point values for notes on the standard clarinet for beginning level musicians. Here, point values go from 1 to 10. 1 is the lowest or easiest and 10 is the highest or hardest. Please reference these when answering the next questions.

G3 to G4: 1
A4: 2
B4 to C5: 5
C#5 and above: 10
All others: 10 (default highest value)

For the standard clarinet, do you feel that these ranges of notes appropriately cover the correct ranges for beginners?

Strongly Agree	Agree	Neither Agree Nor Disagree	Disagree	Strongly Disagree
○	○	○	○	○

If you think there should be any additional ranges covered for beginners on standard clarinet, please list those ranges below:

Do you feel that the point value of 1 is accurate for the range of notes G3 to G4 on standard clarinet as a beginner?

○ Yes

○ No

What point value would you ascribe to the range G3 to G4 on standard clarinet as a beginner?

	1	2	3	4	5	6	7	8	9	10
G3 to G4										

Do you feel that the point value of 2 is accurate for the range of notes A4 on

standard clarinet as a beginner?

○ Yes

○ No

What point value would you ascribe to the range A4 on standard clarinet as a beginner?

	1	2	3	4	5	6	7	8	9	10
A4										

Do you feel that the point value of 5 is accurate for the range of notes B4 to C5 on standard clarinet as a beginner?

○ Yes

○ No

What point value would you ascribe to the range B4 to C5 on standard clarinet as a beginner?

	1	2	3	4	5	6	7	8	9	10
B4 to C5										

Do you feel that the point value of 10 is accurate for the range of notes C#5 and above on standard clarinet as a beginner?

○ Yes

○ No

What point value would you ascribe to the range C#5 and above on standard clarinet as a beginner?

	1	2	3	4	5	6	7	8	9	10
C#5 and above										

Do you feel that the point value of 10 is accurate for all other notes on standard clarinet as a beginner?

○ Yes

○ No

What point value would you ascribe to all other notes on standard clarinet as a beginner?

	1	2	3	4	5	6	7	8	9	10
All other notes										

The following questions will ask about your preferences for the point values of intervals.

We have set the following point values for intervals on the standard clarinet for beginning level musicians. Here, point values go from 1 to 10. 1 is the lowest or easiest and 10 is the highest or hardest. Please reference these when answering the next questions.

Unisons in any range: 1
Seconds in the range G3 to G4: 2
Thirds in the range G3 to G4: 3
Fourths in the range G3 to G4: 4
Fifths in the range G3 to G4: 5
Sixths in any range: 9
Sevenths in any range: 9
Eighths (octaves) in any range: 9
Any interval in the range B4 to C5: 8
Any interval in the range C#5 and above: 10
Any interval greater than an octave in any range: 10
All others: 10 (default highest value)

For the standard clarinet, do you feel that these intervals appropriately cover the correct ranges for beginners?

Strongly Agree	Agree	Neither Agree Nor Disagree	Disagree	Strongly Disagree
○	○	○	○	○

If you think there should be any additional ranges covered for beginners on standard clarinet, please list those ranges below:

Do you feel that the point value of 1 is accurate for the unison interval in any range on standard clarinet as a beginner?

○ Yes

○ No

What point value would you ascribe to the unison interval in any range on standard clarinet as a beginner?

	1	2	3	4	5	6	7	8	9	10
Unisons in any range										

Do you feel that the point value of 2 is accurate for the seconds interval in the range G3 to G4 on standard clarinet as a beginner?

○ Yes

○ No

What point value would you ascribe to the seconds interval in the range G3 to G4 on standard clarinet as a beginner?

	1	2	3	4	5	6	7	8	9	10
Seconds in the range G3 to G4										

Do you feel that the point value of 3 is accurate for the thirds interval in the range G3 to G4 on standard clarinet as a beginner?

○ Yes

○ No

What point value would you ascribe to the thirds interval in the range G3 to G4 on standard clarinet as a beginner?

	1	2	3	4	5	6	7	8	9	10
Thirds in the range G3 to G4										

Do you feel that the point value of 4 is accurate for the fourths interval in the range G3 to G4 on standard clarinet as a beginner?

○ Yes

○ No

What point value would you ascribe to the fourths interval in the range G3 to G4 on standard clarinet as a beginner?

	1	2	3	4	5	6	7	8	9	10
Fourths in the range G3 to G4										

Do you feel that the point value of 5 is accurate for the fifths interval in the range G3 to G4 on standard clarinet as a beginner?

◯ Yes

◯ No

What point value would you ascribe to the fifths interval in the range G3 to G4 on standard clarinet as a beginner?

	1	2	3	4	5	6	7	8	9	10
Fifths in the range G3 to G4										

Do you feel that the point value of 9 is accurate for the sixths interval in any range on standard clarinet as a beginner?

◯ Yes

◯

4/10/2015   Qualtrics Survey Software

No

What point value would you ascribe to the sixths interval in any range on standard clarinet as a beginner?

	1	2	3	4	5	6	7	8	9	10
Sixths interval in any range										

Do you feel that the point value of 9 is accurate for the sevenths interval in any range on standard clarinet as a beginner?

○ Yes

○ No

What point value would you ascribe to the sevenths interval in any range on standard clarinet as a beginner?

	1	2	3	4	5	6	7	8	9	10
Sevenths interval in any range										

Do you feel that the point value of 9 is accurate for the eighths or octaves interval

in any range on standard clarinet as a beginner?

○ Yes

○ No

What point value would you ascribe to the eighths or octaves interval in any range on standard clarinet as a beginner?

	1	2	3	4	5	6	7	8	9	10
Eighths or octaves interval in any range										

Do you feel that the point value of 10 is accurate for any interval greater than an octave in any range on standard clarinet as a beginner?

○ Yes

○ No

What point value would you ascribe to any interval greater than an octave in any range on standard clarinet as a beginner?

	1	2	3	4	5	6	7	8	9	10
Any interval greater than an octave in any										

	1	2	3	4	5	6	7	8	9	10
range										

Do you feel that the point value of 10 is accurate for all other intervals on standard clarinet as a beginner?

○ Yes

○ No

What point value would you ascribe to the sixths interval in any range on standard clarinet as a beginner?

	1	2	3	4	5	6	7	8	9	10
Sixths interval in any range										

The following questions will ask about your preferences for the multiplier values of articulations.

We have set the following multiplier values for articulations on the standard clarinet for beginning level musicians. Here, point values can be anything above 0. Lower is easier and higher is more difficult. Please reference these when answering the next questions.

No articulations (standard tonguing): 1
Staccato: 1.2
Tenuto: 1.2
Slur: 0.5

Accented: 1.1
Marcato or strong accent: 1.4
All others: 1.5

For the standard clarinet, do you feel that this range of articulations appropriately cover the correct ranges for beginners?

Strongly Agree	Agree	Neither Agree Nor Disagree	Disagree	Strongly Disagree
○	○	○	○	○

If you think there should be any additional articulations covered for beginners on standard clarinet, please list those ranges below:

Do you feel that the multiplier value of 1 is accurate for no articulations or standard tonguing on standard clarinet as a beginner?

○ Yes
○ No

What multiplier value would you ascribe to no articulations or standard tonguing on standard clarinet as a beginner?

    0   1   2   3   4   5

No articulations or standard tonguing					

Do you feel that the multiplier value of 1.2 is accurate for staccato on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to staccato on standard clarinet as a beginner?

	0	1	2	3	4	5
Staccato						

Do you feel that the multiplier value of 1.2 is accurate for tenuto on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to tenuto on standard clarinet as a beginner?

	0	1	2	3	4	5
Tenuto						

Do you feel that the multiplier value of 0.5 is accurate for slur on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to slur on standard clarinet as a beginner?

	0	1	2	3	4	5
Slur						

Do you feel that the multiplier value of 1.1 is accurate for accent on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to accent on standard clarinet as a beginner?

	0	1	2	3	4	5
Accent						

Do you feel that the multiplier value of 1.4 is accurate for marcato or strong accent on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to marcato or strong accent on standard clarinet as a beginner?

	0	1	2	3	4	5
Marcato or strong accent						

Do you feel that the multiplier value of 1.5 is accurate for all other articulations on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to all other articulations on standard

clarinet as a beginner?

	0	1	2	3	4	5
All other articulations						

The following questions will ask about your preferences for the multiplier values of dynamics.

We have set the following multiplier values for dynamics on the standard clarinet for beginning level musicians. Here, point values can be anything above 0. Lower is easier and higher is more difficult. Please reference these when answering the next questions.

MF: 1
MP: 1
F: 1.1
FF: 1.2
P: 1.3
PP: 1.5
All others: 1.5

For the standard clarinet, do you feel that this range of dynamics appropriately cover the correct ranges for beginners?

Strongly Agree	Agree	Neither Agree Nor Disagree	Disagree	Strongly Disagree
○	○	○	○	○

If you think there should be any additional dynamics covered for beginners on standard clarinet, please list those ranges below:

Do you feel that the multiplier value of 1 is accurate for MF (mezzo forte) on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to MF on standard clarinet as a beginner?

	0	1	2	3	4	5
MF						

Do you feel that the multiplier value of 1 is accurate for MP (mezzo piano) on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to MP on standard clarinet as a beginner?

	0	1	2	3	4	5
MP						

Do you feel that the multiplier value of 1.1 is accurate for F (forte) on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to F on standard clarinet as a beginner?

	0	1	2	3	4	5
F						

Do you feel that the multiplier value of 1.2 is accurate for FF (fortissimo) on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to FF on standard clarinet as a beginner?

	0	1	2	3	4	5
FF						

Do you feel that the multiplier value of 1.3 is accurate for P (piano) on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to P on standard clarinet as a beginner?

	0	1	2	3	4	5
P						

Do you feel that the multiplier value of 1.5 is accurate for PP (pianissimo) on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to PP on standard clarinet as a beginner?

	0	1	2	3	4	5
PP						

Do you feel that the multiplier value of 1.5 is accurate for all other dynamics on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to all other dynamics on standard clarinet as a beginner?

	0	1	2	3	4	5
All other dynamics						

The following questions will ask about your preferences for the multiplier values of key signatures.

We have set the following multiplier values for key signatures on the standard clarinet for beginning level musicians. Here, point values can be anything above 0. Lower is easier and higher is more difficult. Please reference these when answering the next questions.

C: 1

G: 1.1

D: 1.1

A: 1.2

E: 1.3

B: 1.4

F#: 1.5

C#: 1.6

F: 1.1

Bb: 1.1

Eb: 1.2

Ab: 1.3

Db: 1.4

Gb: 1.5

Cb: 1.6

Do you feel that the multiplier value of 1 is accurate for the key of C on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to the key of C on standard clarinet as a beginner?

	0	1	2	3	4	5
Key of C						

Do you feel that the multiplier value of 1.1 is accurate for the key of G on

standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to the key of G on standard clarinet as a beginner?

	0	1	2	3	4	5
Key of G						

Do you feel that the multiplier value of 1.1 is accurate for the key of D on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to the key of D on standard clarinet as a beginner?

	0	1	2	3	4	5
Key of D						

Do you feel that the multiplier value of 1.2 is accurate for the key of A on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to the key of A on standard clarinet as a beginner?

	0	1	2	3	4	5
Key of A						

Do you feel that the multiplier value of 1.3 is accurate for the key of E on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to the key of E on standard clarinet as a beginner?

	0	1	2	3	4	5
Key of E						

Do you feel that the multiplier value of 1.4 is accurate for the key of B on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to the key of B on standard clarinet as a beginner?

	0	1	2	3	4	5
Key of B						

Do you feel that the multiplier value of 1.5 is accurate for the key of F# on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to the key of F# on standard clarinet as a beginner?

	0	1	2	3	4	5
Key of F#						

Do you feel that the multiplier value of 1.6 is accurate for the key of C# on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to the key of C# on standard clarinet as a beginner?

	0	1	2	3	4	5
Key of C#						

Do you feel that the multiplier value of 1.1 is accurate for the key of F on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to the key of F on standard clarinet as a beginner?

	0	1	2	3	4	5
Key of F						

4/10/2015   Qualtrics Survey Software

Do you feel that the multiplier value of 1.1 is accurate for the key of Bb on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to the key of Bb on standard clarinet as a beginner?

	0	1	2	3	4	5
Key of Bb						

Do you feel that the multiplier value of 1.2 is accurate for the key of Eb on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to the key of Eb on standard clarinet as a beginner?

	0	1	2	3	4	5
Key of Eb						

Do you feel that the multiplier value of 1.3 is accurate for the key of Ab on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to the key of Ab on standard clarinet as a beginner?

	0	1	2	3	4	5
Key of Ab						

Do you feel that the multiplier value of 1.4 is accurate for the key of Db on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to the key of Db on standard clarinet as a beginner?

	0	1	2	3	4	5
Key of Db						

Do you feel that the multiplier value of 1.5 is accurate for the key of Gb on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to the key of Gb on standard clarinet as a beginner?

	0	1	2	3	4	5
Key of Gb						

Do you feel that the multiplier value of 1.6 is accurate for the key of Cb on standard clarinet as a beginner?

○ Yes

○ No

What multiplier value would you ascribe to the key of Cb on standard clarinet as a beginner?

	0	1	2	3	4	5

4/10/2015   Qualtrics Survey Software

Key of Cb					

Thank you for completing the survey and for providing your feedback!

If you have any final comments or feedback, please feel free to leave it below.

Survey Powered By Qualtrics