UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Programa de Maestría y Doctorado en Música

Facultad de Música
Centro de Ciencias Aplicadas y Desarrollo Tecnológico
Instituto de Investigaciones Antropológicas

Universo Tonal

Desarrollo de una aplicación de cómputo para el análisis musical armónico por medio de métodos matemáticos

TESIS

QUE PARA OPTAR POR EL GRADO DE:
MAESTRÍA EN MÚSICA, TECNOLOGÍA MUSICAL

PRESENTA:

CRISTIAN MANUEL BAÑUELOS HINOJOSA

TUTOR:

DR. FELIPE ORDUÑA BUSTAMANTE, (CCADET)

MÉXICO, D.F. JUNIO 2015

Índice general

Índice de figuras

Índice de tablas

Agradecimientos

Este trabajo fue posible gracias al gran apoyo y amor que me brindan mi familia y amigos, les agradezco infinitamente a cada uno de ustedes por ser parte de esta aventura. Gracias a mi amorcito Elisa Borrero, eres la luz que guía mi vida; a mis padres y hermana: Manuel Bañuelos, María de los Ángeles, Stephanie Carolina por su amor incondicional; a mis amigos y familia: Enrique Díaz, César Ramirez, Itze, Esteban, Fabián, Jesús, Jorge David, Rubén, Tania Varela, Otto, María Clara, Maby, así como todos mis compañeros de maestría con los que compartí buenos momentos en estos dos años. Un especial agradecimiento a mi tutor el Dr. Felipe Orduña, por su apoyo, dedicación y paciencia para guiarme. A todos mis maestros que me han preparado para este momento: Dr. Evgeny Korolkov, Dr. José Gurría, Dr. Roberto Morales, Dr. Pablo Rendón, Dra. Eunice Padilla, Mtro. Pablo Silva, Dra. Consuelo Carredano, Mtra. Irina Shishkina, Mtra. Olena Galitskaya, Dr. Alejandro Escuer, M. en I. Antonio López, Dr. Manuel Rocha y todos aquellos que me han compartido generosamente sus conocimientos.

A la UNAM y la Facultad de Música, por abrirme las puertas para realizar esta investigación. De igual manera a la DGEP y su apoyo económico por medio de la beca de maestría, la cual me permitió dedicarme de lleno a los estudios.

Le dedico este trabajo a todos los matemáticos que quieren adentrarse al mundo de la música, y a los músicos que están abiertos a nuevas posibilidades y que por medio del trabajo en conjunto podamos extender las fronteras del conocimiento.

Resumen

El análisis musical armónico asistido por computadora es un área de investigación reciente e interesante. En los últimos años ha adquirido relevancia debido a la gran cantidad de archivos y bibliotecas con partituras digitales que se encuentran en la Internet. Es de interés para la comunidad musical encontrar formas de extraer información estructural y de contenido de estos archivos. En este trabajo de tesis se desarrolló un modelo computacional capaz de tomar una partitura digital en formatos MIDI o MusicXML para realizar de manera semi-automática el etiquetado de acordes presentes en la pieza y obtener una clasificación de acuerdo con su posibles funciones tonales. Para esto, se implementaron representaciones matemáticas de acordes existentes en la literatura como el Círculo Cromático, el Tonnetz y la Banda de Moebius; y por medio de ellas fue posible llevar a cabo la clasificación requerida para el etiquetado. Asimismo, se desarrolló un modelo llamado Universo Tonal, capaz de tomar un acorde cualquiera y obtener todas sus posibles funciones tonales con respecto a las veinticuatro tonalidades mayores y menores, las cuales se representan en la visualización propuesta en este trabajo llamada Grafo Tonal. Por último, se implementó el sistema completo en un programa de cómputo que permite al usuario realizar el etiquetado, clasificación y visualización de acordes y regiones tonales. Finalmente, se probó con obras de J.S. Bach, F. Chopin y W.A. Mozart, se obtuvo un etiquetado correcto correspondiente al análisis de las piezas encontrado en la literatura. Se concluye que la aplicación de métodos matemáticos y computacionales para la extracción y clasificación de armonías a partir de una partitura digital es una herramienta que puede ser muy útil para el estudio de la música, debido a la automatización y simplificación de procesos repetitivos que ofrece.

Introducción

El estudio de la música es un proceso complejo que requiere una serie de habilidades y conocimientos diversos y especializados. Independientemente de si se es intérprete, analista, compositor o aficionado, para ser competente se necesita como mínimo una memoria hábil, capacidad de síntesis, conocimientos especializados sobre cómo se organizan los sonidos en el tiempo y sensibilidad estética, entre muchas otras habilidades. Dentro de esta cantidad de destrezas que requiere la música, existen tareas repetitivas que en lugar de fomentar la creación terminan por robar tiempo. Con la finalidad de simplificar el trabajo y los procesos mecánicos, se han desarrollado herramientas tecnológicas para ayudar al músico en sus tareas cotidianas. Por ejemplo, imaginemos a un compositor que creó una pieza para piano en *Do* mayor. Si por alguna razón quisiera transportarla a *Sol* mayor y quisiera hacerlo en papel, necesitaría una buena cantidad de tiempo y paciencia. En cambio, si utilizara un editor de partituras digital como *MuseScore, Sibelius* o *Finale*, bastarían un par de *clics* para resolver el problema. Como este, hay varios ejemplos de tareas repetitivas que han sido simplificadas en gran medida gracias a la introducción de herramientas computacionales para el estudio y la creación de la música. En el presente trabajo de tesis propongo un sistema para ayudar al músico a organizar y entender mejor el etiquetado de armonías en una partitura tonal, así como las regiones armónicas que dicha pieza musical visita. El sistema estará basado en técnicas de representaciones geométricas y matemáticas de acordes desarrolladas por investigadores matemáticos en los últimos años.

La ciencia y tecnología han avanzado al punto donde se pueden analizar enormes cantidades de datos de forma muy eficiente, clasificando y realizando inferencias acerca de las tendencias,

patrones y relaciones dentro de la información. Por medio de algoritmos se puede analizar un sistema de partículas; también es posible modelar sistemas con comportamiento caótico, e incluso analizar las tendencias globales del comportamiento de miles de individuos en la Internet. Parecería natural que en el estudio y análisis de las complejidades de las obras musicales se utilicen procesos similares para extraer información estructural de los archivos digitales que contienen música; sin embargo, esto no suele ser así, ya que las herramientas computacionales que se utilizan en la música suelen ser principalmente de edición, ya sea de audio o de partituras digitales, como *Sibelius, Finale, Pro Tools, Logic Pro, MuseScore*, entre otros.

En este trabajo se aborda el problema de etiquetar y clasificar los acordes que aparecen en una partitura digital por medio de la creación de diagramas que muestren las regiones tonales que surgen de ellos. Esto con la finalidad de que el músico tenga una herramienta que vaya más allá de las funciones de un editor de partituras y no solo pinte o dibuje notas en la pantalla, sino que extraiga de manera automática información de las funciones armónicas internas en la pieza.

Para abordar el problema planteado, se desarrolla un modelo matemático que a su vez se implementa en un programa de cómputo capaz de realizar etiquetado y clasificación de acordes de acuerdo con sus regiones tonales. Por un lado, se aporta un trabajo teórico interdisciplinario entre matemáticas-computación y análisis armónico musical, mientras que por otro, se crea una herramienta computacional que ayuda al estudio de la música.

El modelo propuesto se basa en la categorización de acordes a partir de una representación geométrica, la cual a su vez facilita su identificación en un lenguaje de programación. Investigadores que estudian la música con herramientas matemáticas, (TYMOCZKO, 2011; MAZZOLA, 2002), han desarrollado formas de representar acordes en espacios geométricos, los cuales son útiles porque ya que se cuenta con su visualización, es posible extraer información estructural oculta. Por ejemplo, como se mostrará más adelante, utilizando el llamado *Círculo Cromático* se verá que cada acorde mayor tiene la representación de un triángulo, mientras que a los acordes de séptima disminuida les corresponde un cuadrado. Esto es útil para la clasificación de acordes por medio de un lenguaje de programación, ya que si dibujamos un conjunto de notas, basta con que el algorit-

mo revise la figura para saber de qué acorde se trata. Además del *Círculo Cromático*, se mostrarán otras representaciones geométricas de acordes. Cada una de ellas proporcionará información específica; entre ellas se encuentran el *Tonnetz*, la *Banda de Moebius* y una nueva que propongo llamada *Grafo Tonal*. La representación de acordes en estos espacios es una manera de simplificar la traducción de conceptos musicales armónicos a un lenguaje de programación, permitiendo a la teoría matemática fungir como enlace entre conceptos y áreas.

En términos concretos, para crear el programa se implementan tres representaciones geométricas ya existentes en la literatura: el *Círculo Cromático*, el *Tonnetz* y la *Banda de Moebius*. Estas tres se utilizarán para la clasificación de acordes. Por otra parte, se creará una nueva estructura llamada *Universo Tonal* que tiene una representación que se definirá como *Grafo Tonal*. Estas servirán para organizar y clasificar dichos acordes de acuerdo con sus funciones tonales.

Muchos de los avances de investigaciones en cuanto a representaciones matemáticas de la música aún se encuentran en una etapa teórica y abstracta. Para que estos desarrollos puedan validar su utilidad en el trabajo cotidiano de un músico, es necesario que se tenga acceso a ellos; sin embargo, el lenguaje y notación que la Teoría Matemática de la Música utiliza suelen ser muy especializados y requieren conocimientos de geometría y álgebra avanzada, de nivel universitario. Por esta razón los trabajos de investigación mencionados suelen parecer lejanos y ajenos a la vida del músico, y con frecuencia terminan como datos curiosos. A pesar de que para la creación y desarrollo de las estructuras matemáticas musicales se requiere un conocimiento avanzado de matemáticas, los resultados visuales son lo suficientemente claros e intuitivos para que un músico pueda adoptarlos a su trabajo sin problema; siempre y cuando el músico no tenga que hacer cálculos de forma manual. Es conveniente que se le presente en la forma de una herramienta de exploración, en la cual pruebe introducir distintas melodías, acordes o partituras conocidas y desarrollar así una intuición visual de su representación.

Este trabajo se basa en la premisa de que la identificación visual de los acordes es una habilidad que puede extender nuestra manera de concebir la organización de la música. Por ejemplo, al evocar nuestro aprendizaje de las estructuras de acordes, es posible recordar la primera vez que se

nos dijo que un acorde mayor se componía de una tercera mayor y una tercera menor sobrepuestas. Lo más probable es que nuestro aprendizaje se basara en identificar su disposición en el piano, o quizá su representación en la partitura. Sin embargo, si el estudiante hubiese tenido una herramienta para visualizar acordes en espacios geométricos, la imagen mental habría complementado su entendimiento de la estructura musical. Es decir, en lugar de recordar que una triada mayor es una tercera menor y una tercera mayor concatenadas, podría simplemente recordar que una triada mayor es un triángulo en el círculo cromático.

La intención de este trabajo es crear una herramienta que permita al músico explorar las representaciones geométricas de acordes y regiones tonales de forma rápida e intuitiva, sin necesidad de realizar cálculos numéricos. El programa de cómputo aquí propuesto servirá para buscar y etiquetar acordes en una partitura digital; estos acordes se mostrarán automáticamente en las distintas representaciones geométricas mencionadas, de modo que el usuario podrá visualizar de forma fácil y sistemática las distintas regiones tonales que la pieza visita.

Al plantear la idea de una herramienta de cómputo que ayude al análisis armónico, especialmente en el etiquetado de acordes y su clasificación en regiones tonales, surgen dos preguntas muy importantes: "¿Qué piezas será capaz de analizar?" y "¿Quién es el usuario idóneo para este programa?". Para responder estas preguntas quiero mencionar mi experiencia de aprendizaje musical. Dado que vengo de una formación académica principalmente matemática y mi educación musical siempre fue paralela a esta, la forma estructurada de pensar y organizar las ideas que surgen de una visión científica del mundo influenció mucho mi forma de adquirir conocimientos musicales. Ocurría con frecuencia, que al enfrentarme a nuevos conceptos musicales no los lograba entender a profundidad por medio de la notación musical, sino que me era más intuitivo y natural traducirlas a un lenguaje numérico matemático. Por ejemplo, al intentar comprender la estructura de una escala mayor como *tono-tono-semitono-tono-tono-tono-semitono* y que puede ser transportada a diferentes notas fundamentales, fue mucho más intuitivo para mi pensarla como una serie de números $\{0,2,4,5,7,9,11\}$ que representan los intervalos en semitonos de las notas[1], y saber que

[1]Es importante aclarar que se presupone un temperamento igual para las escalas que menciono en este contexto.

transponer la escala sería sumar un número constante a todos los elementos. Por razones como estas, considero que existe un público tentativo para el cual la herramienta propuesta será de gran utilidad. Entre los usuarios ideales potenciales se encuentran: un científico con deseos de adentrarse en el entendimiento de conceptos musicales de armonía y funciones armónicas; por otra parte, un músico con afición matemática que busque nuevas formas de entender las estructuras musicales que maneja en su trabajo cotidiano. De acuerdo a la investigación de Orduña y Godinez, los perfiles mencionados para el usuario, son pertinentes para las tendencias que existen en la formación de profesionales en la tecnología musical:

> En general, parece que la tendencia es hacia el profesional que cuenta un gran conocimiento y manejo de tipo técnico (uso del equipo de audio), pero que carecen de un conocimiento musical sólido. Por otra parte; tenemos al músico tradicional con un bagaje musical sólido -solfeo, armonía, contrapunto, historia del arte, acústica, etcéterapero con una gran laguna en cuanto a conocimiento y manejo de tecnología se refiere. (ORDUÑA F., 2007, p.271-272).

Con respecto al tipo, estilo y periodo de música con los que este programa trabaja, de manera general se busca entender la *música tonal*. Dado que este es un concepto que engloba muchos compositores y periodos, el repertorio de obras se delimitará a aquellas con una organización armónica en las que los acordes presentes tengan funciones claramente tonales del tipo *Tónica, Subdominante o Dominante*. En este repertorio se incluirán principalmente a diversos compositores del periodo clásico y romántico. Entre ellos se encuentra gran parte del corpus de compositores como W.A. Mozart, Haydn, Beethoven, Chopin, Schubert, Haendel, Tchaikovsky, Schumann, Rachmaninov, entre muchos otros. Adicionalmente, se puede incluir en el repertorio mucha de la música popular contemporánea, ya que en gran manera el rock y el pop tienen estructuras armónicas que cumplen con las funciones tonales requeridas. El sistema no busca analizar obras de compositores que cada vez se alejan más de funciones claras tonales y buscan más bien texturas de sonido o efectos sonoros nuevos, como el caso de: Debussy, Stravinsky, Wagner y Mahler. Tampoco se busca analizar piezas con gran contenido polifónico, ya que el etiquetado de acordes y regiones tonales en este

tipo de piezas debe tomar en cuenta las nociones del contrapunto y movimiento de voces las cuales no se abordarán en este trabajo.

Como ejemplo de uso del programa, se analizan fragmentos del Preludio Op. 28, No. 20 F. Chopin, así como de la Sonata No. 16 K.545 W.A. Mozart y el Preludio No.1, Libro I, del Clave Bien Temperado de J.S. Bach. Se eligieron estas piezas ya que su contenido armónico es fácil de extraer debido a la disposición de los acordes, ya sea en forma de bloques, notas simultáneas claramente identificables, o presentados en forma arpegiada sencilla de visualizar. Estos ejemplos servirán para motivar al lector a elegir piezas tonales de su interés para analizarlas con las técnicas geométricas presentadas.

Una obra musical engloba muchos aspectos, desde lo técnico hasta lo social, por lo que un análisis exclusivamente armónico representa solo un fragmento que no permite explicar a fondo la intención artística del compositor. Aun así, como en cualquier otro trabajo de corte científico, se apuesta a la idea de dividir el problema de estudio en fragmentos especializados, de modo que el producto final no pretende explicar la obra musical completa, sino dotar al usuario, músico o científico, de una herramienta que le ayude a analizar algunos aspectos armónicos de la obra. Este trabajo de tesis invita a otros investigadores, programadores y músicos a seguir trabajando en conjunto para encontrar nuevas formas de entender y representar los contenidos armónicos.

La presente tesis se divide en cuatro capítulos. En el primero se exponen los antecedentes históricos, teóricos y tecnológicos necesarios para abordar el análisis armónico. Se presentan las distintas notaciones usadas a través de la historia, así como sus implicaciones, ventajas y desventajas. Se describen también algunos de los métodos matemáticos y computacionales que se han utilizado en el análisis armónico. Por último, se expone una lista de aplicaciones de cómputo que abordan este tema. Para facilitar la comprensión, se ha evitado la utilización de notación matemática técnica, y los conceptos necesarios han sido explicados de forma coloquial. Sin embargo, en el apéndice *Antecedentes matemáticos* se muestran como referencia las definiciones formales de los conceptos matemáticos más importantes que se utilizan en la tesis, entre ellos definiciones de teoría de conjuntos, álgebra abstracta, geometría cartesiana y teoría de grafos.

En el segundo capítulo, se desarrolla el modelo matemático propuesto en este trabajo para automatizar el etiquetado de acordes y regiones tonales de las piezas a analizar. Se explica la creación del modelo *Universo Tonal* y se exponen las definiciones matemáticas de conceptos como, *Función Tonal, Tonalidad, Nota*, entre otros. Se describen detalladamente las ventajas de utilizar estas definiciones como base para la aplicación de cómputo.

En el tercer capítulo se abordan los conceptos técnicos necesarios para la implementación computacional de las representaciones geométricas y los algoritmos desarrollados. Se exponen las clases[2] necesarias en pseudo-código[3], de modo que el trabajo en abstracto sea independiente de un lenguaje o sistema operativo en particular. Se exponen los detalles de implementación necesarios para el análisis de obras musicales codificadas en archivos que se encuentren en formatos MIDI y MusicXML[4]. Una vez hecho esto, se comparan las ventajas y desventajas de desarrollar el sistema en distintos lenguajes de programación: Java, C#, Python, Javascript, Unity, SuperCollider, etc. Al final, se describe brevemente el diseño de la interfaz del usuario.

Para poner a prueba el programa desarrollado, en el cuarto capítulo, se analizan una selección de piezas tonales de diversos compositores. Se muestran las virtudes y limitaciones de los

[2]Una clase en el contexto de programación es una estructura de datos que representa un objeto abstracto con variables y funciones, las cuales también se conocen como métodos. Un ejemplo tradicional es definir la clase `Perro` y dotarla de las variables `raza` y `nombre`, y como métodos `ladrar`, `sentar`, esto permite referirnos a la idea de un perro, tan compleja o simplificada como se desee en el contexto de programación. En general podemos definir nuestras clases con las variables y métodos que necesitemos, en este trabajo necesitaremos definir `Acorde`, `Tonalidad`, `FunciónTonal` entre otras, las cuales se mostrarán en el tercer capítulo.

[3]Consiste en una manera de escribir en forma de texto la idea sobre la cual se desarrollará el código de programación. Al describir sólo las ideas, el pseudo-código no depende de ningún lenguaje de programación en específico, de modo que se puede utilizar para desarrollar en distintos lenguajes como serían: *Java, C++, C♯, SuperCollider*, entre muchos otros.

[4]MIDI inició como un protocolo de comunicación para sintetizadores y controladores, sin embargo más adelante sirvió como base del formato de archivos MIDI, los cuales contienen una serie de mensajes organizados en el tiempo para ser reproducidos por un sintetizador. Por otra parte, los archivos MusicXML están más enfocados en guardar información acerca de la partitura y no tanto en una secuencia de eventos. Good y Cunningham defienden la idea del uso de archivos MusicXML para la digitalización de partituras (GOOD y cols., 2001; CUNNINGHAM, 2004).

algoritmos y el programa de cómputo.

En la última parte, se hace una breve reflexión de los aportes de este trabajo y las líneas futuras de investigación, como los problemas de la identificación de fragmentos de unidad armónica, el análisis de partituras por métodos automáticos de reconocimiento gráfico o de imagen (tipo OCR, *Optical Character Recognition*), el uso de métodos de inteligencia artificial para la identificación de tonalidades, entre otros.

Este trabajo no consiste únicamente en una herramienta para el etiquetado de acordes y regiones tonales, sino que además propone que se exploren nuevas formas de organizar conceptualmente el contenido armónico por medio de estructuras geométricas. Un músico con formación de conservatorio puede realizar el etiquetado de acordes de una forma muy eficiente y rápida; sin embargo, las representaciones que se proponen en este trabajo no son fáciles de realizar sin una herramienta de cómputo, de modo que este elemento puede ofrecer al músico especializado una nueva perspectiva de las estructuras armónicas y tonales que tanto conoce. De este modo es importante aclarar que además del valor analítico, esta herramienta puede tener un valor didáctico para la comprensión de la armonía, aunque esto se encuentra fuera del enfoque del trabajo. Una vez terminado el modelo, sería interesante investigar las implicaciones del uso de este tipo de software en la educación musical. El objetivo final es presentar al mundo de la música, una herramienta nueva que pueda utilizarse y probarse, para saber si este tipo de organización matemática puede aportar algo al conocimiento musical.

Capítulo 1

Antecedentes

Para abordar este trabajo de investigación es necesario conocer los métodos y técnicas que se han utilizado tanto en el área de la música, como en los intentos de introducir conceptos matemáticos y computacionales para su estudio. En este capítulo haremos una breve revisión de los distintos tipos de notaciones para denotar acordes, así como sus funciones tonales con respecto a una tonalidad; se hará una revisión de distintas representaciones geométricas de acordes, entre ellos el *Círculo Cromático*, la *Banda de Moebius*, el *Tonnetz*; se mostrarán las líneas de investigación actuales acerca del análisis musical armónico computacional, al enunciar los resultados y avances más importantes del área; por último se muestran algunos de los programas computacionales comerciales que se encuentran disponibles para este tipo de estudio musical. Se concluye con una reflexión de las áreas que falta estudiar y desarrollar, para proponer en qué punto entra este trabajo de tesis para contribuir en el área.

1.1. Análisis musical armónico

Existen distintos tipos de análisis, algunos se enfocan más en la cuestión histórica y el significado con respecto al autor, algunas son estructurales de la pieza buscando tendencias globales y generales, mientras que otros son más enfocadas directamente al contenido musical, (SALZER, 1962; PISTON, 1978).

En este trabajo de investigación es de especial interés el componente armónico del análisis, ya que, por una parte en muchas obras tonales la forma en que los acordes interactúan y se presentan, es uno de los pilares que guía el flujo de la música; y por otra, los modelos matemáticos se prestan muy bien para describir las relaciones entre ellos.

Para un trabajo interdisciplinario como el presente, es necesario tener claros y bien definidos los conceptos de las áreas que intervienen, por esta razón, comenzaremos por hacer una breve revisión de las notaciones que se utilizan para representar acordes y conjuntos de alturas. Existen distintos tratados de teoría musical que ofrecen una forma sistemática de abordar la clasificación de acordes y reconocimiento de funciones tonales, desde muy antiguos como el tratado de armonía de Rameau hasta enfoques más contemporáneos como el tratado de armonía de Schoenberg, (RAMEAU, 1722; SCHOENBERG, 1978).

A pesar de la teorización de la práctica musical, se puede decir con seguridad que no existen reglas fijas que dicten notaciones y armonías correctas e incorrectas, existen prácticas y tendencias que se adoptaron en ciertos tiempos y regiones. En este sentido, los tratados musicales cumplen la función de documentos históricos que presentan una síntesis de dichas prácticas, las cuales nos servirán de referencia para abordar el tema del análisis armónico musical.

Cada estilo de notación de acordes nos permite referirnos a distintos aspectos de organizar la música, en algunos casos se da preferencia a las cuestiones improvisativas como en el *Bajo Cifrado* y la notación de *Jazz*, mientras que en otros, como la notación gradual se da preferencia a la estructura funcional del acorde. Cada notación hace una referencia implícita a estructuras matemáticas, en este sentido nos interesa estudiarlas para identificar en qué puntos podemos crear algoritmos que hagan uso de estas representaciones.

Es importante aclarar que el análisis armónico consiste en mucho más que etiquetado y clasificación de acordes, ya que en su forma más general, tiene que ver, sobre todo, con cómo se organizan las tensiones y relajaciones tonales a lo largo de la obra. Sin embargo, en este trabajo vamos a enfocarnos exclusivamente en la parte de etiquetado y clasificación, por lo cual es importante recordar las notaciones con las que estos acordes pueden ser clasificados.

1.2. Distintas notaciones de representación de acordes

Para poder representar los distintos acordes y regiones armónicas que se presentan en una pieza existen diversas notaciones, a continuación se presenta una breve recopilación de algunas de las más comunes, entre ellas: el bajo cifrado, notación por grados, notación jazz/pop, pitch classes y algunas de las notaciones numéricas. Se expone en cada una de ellas sus ventajas e implicaciones, tanto musicales como estructurales.

Bajo cifrado

La notación de bajo cifrado, o bajo continuo, surgió como una herramienta de escritura en el periodo barroco para indicar las armonías sobre las cuales el músico debía improvisar de acuerdo con señalamientos que aparecen en la partitura. Este cifrado permaneció también como herramienta de análisis incluso después de que pasó el auge del estilo barroco, (COOK, 1987). La notación consiste en especificar la nota más grave del acorde a realizar, a la cual se le colocan indicadores numéricos y simbólicos que especifican los intervalos que se deben construir sobre el bajo. Si no se agrega nada sobre la nota, significa que se construya una triada diatónica correspondiente a la armadura en uso sin importar el registro ni la posición melódica. En el caso de que se presenten otros símbolos como sostenidos, bemoles, signo de suma, entre otros, se alterará la nota en el intervalo indicado. Para conectar dos acordes consecutivos, el intérprete debe ser muy hábil y conocer a profundidad un conjunto de reglas muy precisas, entre las cuales están: la prohibición de intervalos de octavas o quintas paralelas, restricciones acerca de las duplicaciones de las notas y la disposición de la separación de las notas entre sí. Como referencia al uso de la notación del bajo cifrado ver el trabajo de ALDWELL y cols. (2010), así como el de CHRISTENSEN y ROBINSON (2002).

En teoría, con este sistema se puede expresar cualquier colección de notas, sin embargo sólo funciona eficientemente para un sistema armónico basado en escalas diatónicas, ya que si tratamos de expresar conjuntos de notas que funcionen en un contexto atonal, el bajo cifrado se encon-

trará con relaciones interválicas que son difíciles de expresar, o en su defecto, con una saturación de símbolos que harán ineficaz este sistema de notación. Por otra parte, no es muy útil desde el punto de vista del análisis, ya que no da información alguna de acordes equivalentes; por ejemplo en la notación no hay forma de saber que el acorde *Do mayor* en estado fundamental está relacionado con su primera inversión. Esto es distinto a decir que una persona que es hábil en la lectura de bajo cifrado no puede ver fácilmente la relación entre un acorde y su inversión, la situación es que la notación como tal, en sus símbolos no contiene dicha información, lo que es muy importante ya que por medio de estos símbolos es como introduciremos la información musical a un sistema de cómputo.

El bajo cifrado sirvió como base histórica para otras representaciones simbólicas armónicas, como la de grados de acorde en cifrado romano, ya que esta mantiene la simbología de números para indicar los intervalos del acorde. En este trabajo lo utilizaremos principalmente como referencia histórica y elemento para comparación. En la figura (1.1) se muestran algunos ejemplos de realización de bajos cifrados. Actualmente existen investigaciones que buscan crear algoritmos para encontrar de forma automatizada la realización de bajos continuos, como se puede ver en el trabajo de NIITSUMA y cols. (2011), donde tratan de simular la lógica que un humano sigue al encontrar las armonías adecuadas para un bajo. En un sentido matemático, la realización de un bajo continuo implica resolver un sistema de ecuaciones, ya que encontrar la solución al problema consiste en encontrar unas condiciones iniciales, en este caso sería un acorde, y de acuerdo con ciertas restricciones y reglas, navegar un espacio de posibilidades, donde el compositor decide el camino final a seguir. Esta visión de la armonía como un sistema de ecuaciones puede verse en el trabajo de GIES (2009).

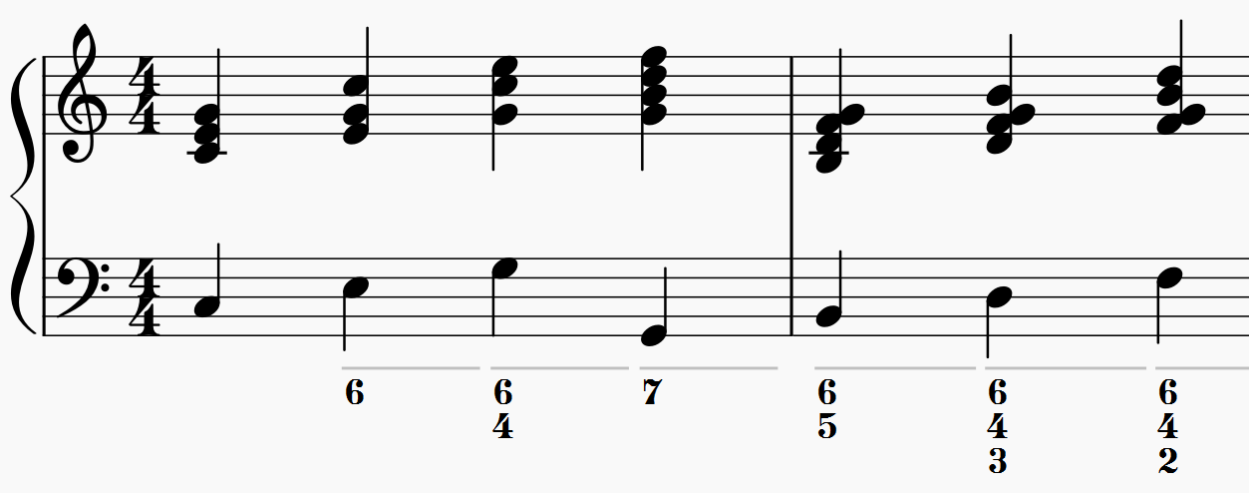


Figura 1.1: Ejemplo de bajo cifrado

Armonía gradual

Otro método de identificar las relaciones que distintos acordes tienen entre sí, es el utilizado en la armonía gradual por medio de la notación de números romanos. De forma similar al bajo cifrado, se indica con símbolos los intervalos que pertenecen al acorde, pero con la diferencia de que no se hace referencia a la nota del bajo, sino al grado del acorde, el cual en lugar de indicarse con el nombre de la nota, se denota por el número romano del grado de la tonalidad que representa. Por ejemplo en la tonalidad *Do mayor*, el acorde *V*, sin ningún otro cifrado, representa la triada del quinto grado, *Sol mayor*. Mientras que el bajo cifrado es una notación mecánica, la notación gradual implica un análisis estructural más profundo, ya que para una representación adecuada es necesario saber la tonalidad en la que se encuentra. Esto se aprecia en el hecho de que una misma colección de notas, puede tener distintas funciones tonales. Por ejemplo el acorde *(sol, si, re)* corresponde al quinto grado (*V*) en *Do mayor*, pero tiene la función tonal *I* en *Sol mayor* y *IV* en *Re mayor*. Del bajo cifrado se hereda también la numeración de las inversiones, como son I_6, I_4^6, I_7, I_5^4, I_3^4, I_2. Es importante notar que no se especifica la forma en que los acordes se conectan entre sí, a diferencia del bajo cifrado, la notación sirve puramente de análisis y no está diseñada para interpretación. En la figura (1.2) se muestra un ejemplo de acordes en la tonalidad *Sib mayor*. Para una descripción más detallada de este sistema ver los trabajos de ALDWELL y cols. (2010) y

KOSTKA y cols. (1995).

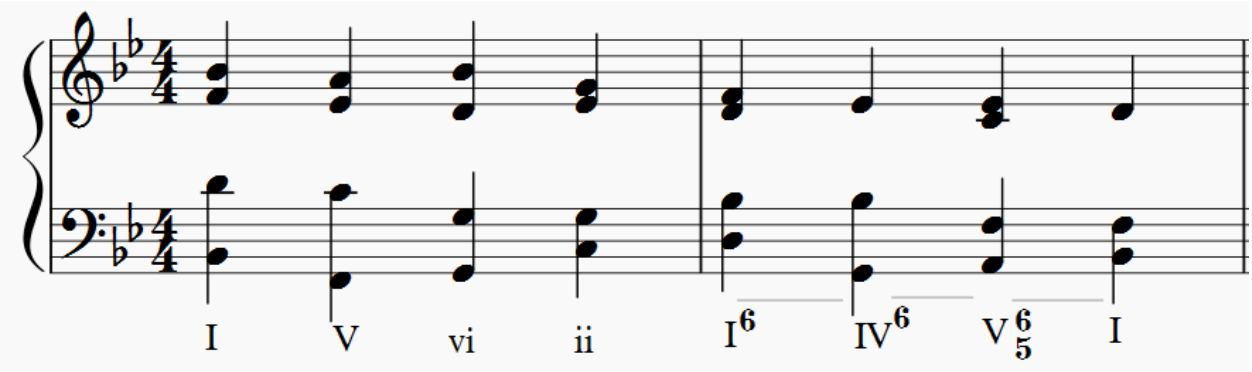


Figura 1.2: Ejemplo de acordes con sus respectivas representaciones de grados romanos

Notación de Riemann

Hugo Riemann desarrolló una variante a este tipo de notación, (RIEMANN, 1896). Funciona de manera similar a la de armonía gradual, con la diferencia que no sólo importan los grados romanos, sino que se hace una distinción acerca de si el acorde funciona como Tónica **T**, Subdominante **S** o Dominante **D**,

La notación gradual es la más importante para este trabajo, ya que funciona como una especie de coordenada que indica la pertenencia de un acorde en una tonalidad. Cada número romano representa una conjunto de acordes que permiten dividir el espacio armónico en secciones geométricas, las cuales se explicarán a detalle más adelante. En el segundo capítulo se utilizarán estos conceptos de clasificación de armonías para la definición de *Universo y Función Tonal*, con los cuales se establecen las bases para el programa de cómputo de análisis.

Jazz - Pop

La notación que usualmente se utiliza en el jazz y pop se distingue de las anteriores en que no tiene como prioridad un análisis estructural armónico, ni una conducción de voces específica. En ella, un acorde se denota por su nota fundamental en *notación alemana*[1] y el tipo, es decir si

[1]Donde las notas *Do, re, mi, fa, sol, la, si,* equivalen a C, D, E, F, G, A, B.

son de tipo *mayor, menor, aumentado, disminuido, dominante con séptima*, etc. Cada uno de ellos tiene un símbolo distinto que varía de acuerdo con versiones de notaciones, como se muestra en la tabla (1.1). Para indicar la nota del bajo, se colocan con una diagonal seguida por la nota más grave, esto puede tener varias funciones, como indicar una nota pedal o indicar la inversión del acorde. Por ejemplo el acorde Do menor se escribiría Cm o $C-$, mientras que su primera inversión sería Cm/E. Para una lista exhaustiva de los tipos de acordes clasificados de esta manera, ver los trabajos de AEBERSOLD (1974) y FRASER (1983).

Tipo de acorde	**Notación**	**Notas incluidas**
Triada Mayor	C	(Do, Mi, Sol)
Triada Menor	Cm,C-,Cmi,Cmin	(Do, Mi♭, Sol)
Triada Disminuida	C^o,Cdim	(Do,Mi♭, Sol♭)
Triada Aumentada	C^+, Caug, $C^{(\sharp 5)}$	(Do,Mi, Sol♯)
Séptima menor	Cm^7,$C-^7$,Cmi^7,$Cmin^7$	(Do,Mi♭, Sol, Si♭)
Dominante con séptima	C7	(Do,Mi, Sol,Si♭)
Séptima Mayor	$Cmaj^7$,$C\Delta^7$,Cma^7,CM^7	(Do,Mi, Sol, Si)

Tabla 1.1: Notación de jazz

Junto con la notación de grados romanos, la notación de Jazz es de gran utilidad ya que nos ayuda a tener un diccionario o base de datos de acordes los cuales se clasifican de acuerdo con la tonalidad que se requiera. A la hora de programar las cuestiones de análisis armónico, primero se revisa qué acorde es en "valor absoluto" (es decir su notación en *Jazz/Pop*), para después pasar a sus valores relativos, con sus respectivos grados tonales de acuerdo con su función tonal que tienen en un contexto dado. Por ejemplo si tomamos el acorde *(sol, si, re, fa)* tendría la notación G_7 o Sol_7 en valor absoluto, pero de acuerdo con la tonalidad en la que se encuentre cambiará su grado o función, en *Do mayor* sería V_7, mientras que en *Fa mayor* sería V_7/V. Por lo anterior se concluye que ambas notaciones se complementan para tener una visión estructural más amplia de los acordes utilizados en la música tonal.

Pitch Classes

La notación de *Pitch Classes*, o clases de alturas, fue promovida y desarrollada en parte por Allen Forte, (FORTE, 1973) con la intención de sistematizar el análisis y composición de la música atonal y serial. La idea consiste en asignar un número a cada tipo de nota $do = 0$, $do\# = 1$, $\cdots$, $si = 11$, sin importar la octava. De modo que el acorde Do mayor *(do, mi, sol)*, se convierte en la colección de notas [0,4,7]. Sin embargo más allá de la notación, una de sus contribuciones fundamentales es el concepto de *forma primaria* de un acorde. En ella se reduce un acorde a su numeración más compacta. Para visualizar esto, es útil dibujar un punto por cada nota del acorde en un círculo [2], dos acordes tienen la misma forma primaria si tienen la misma figura geométrica, sin importar la orientación ni giro[3]. En la figura (1.4) se muestran dos acordes equivalentes por forma primaria *Do mayor* y *Sib menor*.

Esta notación numérica viene en conjunto con toda una teoría y métodos para manipular y transformar las ideas melódicas por medio de operaciones y es muy útil para la música atonal y serial, sin embargo en su reducción tan drástica se simplifica demasiado el objeto de estudio, al grado que un acorde mayor es indistinguible de uno menor. Por lo tanto esta notación nos sirve como referencia histórica en cuanto que fue una de las primeras basada en traducción numérica de la música utilizada a gran escala, sin embargo no la implementaremos en código de momento, ya que nos interesan notaciones que nos permitan preservar el contenido armónico de las piezas desde un punto de vista tonal.

[2] Cada punto representa una nota, comenzando en la parte inferior y se sigue el sentido contrario a las manecillas del reloj $0, 1, 2, \ldots$ hasta cerrar el ciclo.

[3] A este diagrama se le conoce como *Círculo Cromático*, en referencia a la escala cromática que se utiliza para referirse a las doce notas en las que se divide la octava en una afinación con temperamento igual.

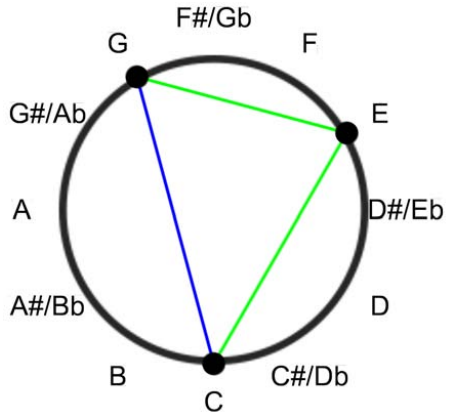


Figura 1.3: Acorde Do mayor en el círculo.

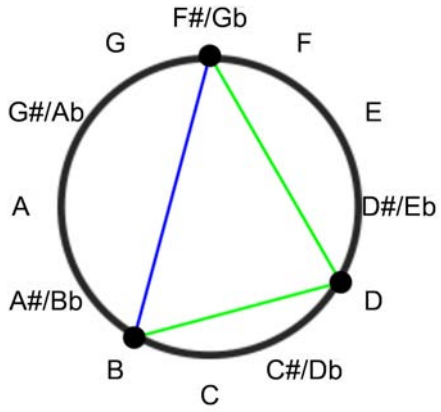


Figura 1.4: Acorde Si menor en el círculo.

1.3. Representaciones matemáticas utilizadas en el análisis musical

Varios investigadores han aplicado técnicas y métodos matemáticos en el estudio de la música. Abordar el estudio de la misma desde un punto de vista matemático resulta muy útil para nuestros fines, ya que trabajar con la armonía y teoría musical, requiere comprender una serie de patrones y estructuras que organizan las notas en el tiempo. En especial, la *Teoría Matemática de la Música* (TMM) es un área de las matemáticas aplicadas cuyo objeto de estudio son precisamente dichos patrones y ofrece herramientas teóricas para entender mejor la estructura de la música. Algunos de los principales trabajos modernos en este tema son los de: TYMOCZKO (2011); MAZZOLA (2002); ANDREATTA y AGON (2003); LARGE (2011); BOON y DECROLY (1995), entre otros. Dado que la especialización matemática requerida para utilizar estos métodos es muy alta, son pocos los músicos que se benefician de sus resultados, que suelen ser muy complicados de implementar. Un ejemplo de esto lo tenemos en el trabajo de Yust, donde aplica una serie de diagramas llamados *Prismas Tonales* para analizar la obra "Ondine" de Ravel, sin embargo, el resultado matemático es bastante complicado de forma que en vez de simplificar el análisis, termina con un nivel de complejidad mayor, (YUST, 2013). Por esta razón, en este trabajo se propone la creación de una aplicación de cómputo dedicada a los músicos, donde se implementen conceptos especializados de la TMM en el análisis armónico musical. Con esta herramienta, un usuario sin entrenamiento

matemático profesional podrá incluir en su trabajo cotidiano, nuevas formas de estudiar y analizar las obras musicales, con énfasis en el etiquetado de acordes y su clasificación en regiones tonales.

La TMM ha aportado al conocimiento musical en distintas áreas como la acústica, estudio de melodías, modelos de cognición, teoría de contrapunto y estructuras armónicas. En esta investigación, nos enfocamos exclusivamente en las estructuras armónicas, por esta razón es conveniente hacer una breve revisión de algunos de los métodos y estructuras matemáticas que se han utilizado para representar y entender la armonía musical.

Pianola, *piano-roll*

En un sentido básico, la música se conforma de sonidos organizados en el tiempo, los cuales pueden aparecer simultanea o individualmente. La disposición de tipo *Pianola* es una representación gráfica de tiempo contra notas, donde se colocan con su altura adecuada cada una de las notas en su tiempo de inicio. En el eje vertical cada espacio corresponde a un semitono, mientras que el eje horizontal representa el tiempo en unidades que varían en cada implementación. Es una evolución de las máquinas de piano automático Pianola y las cajas de música de principios del siglo XX, donde se utilizaban rollos de papel perforado o cilindros metálicos, que al avanzar, activaban el mecanismo sonoro.

Un problema de la pianola para el análisis armónico, es que a diferencia de lo que ocurre en un pentagrama, aquí no existe el concepto de enarmónico, es decir, dado que se presupone un *temperamento igual*, las notas *do*♯ y *re*♭ ambas se identifican con el número 1. Esto representa un problema para acordes en los cuales su escritura identifica a la tonalidad a la que pertenecen. Por otra parte la pianola es muy adecuada para representar el contenido de archivos MIDI, ya que de igual manera no tiene información enarmónica. Esta representación es muy utilizada en secuenciadores y editores de audio (DAW)[4], ya que es una sencilla forma de visualizar las notas en el tiempo.

La representación visual de *pianola*, a pesar de no contener explícitamente información armóni-

[4]Digital Audio Workstations.

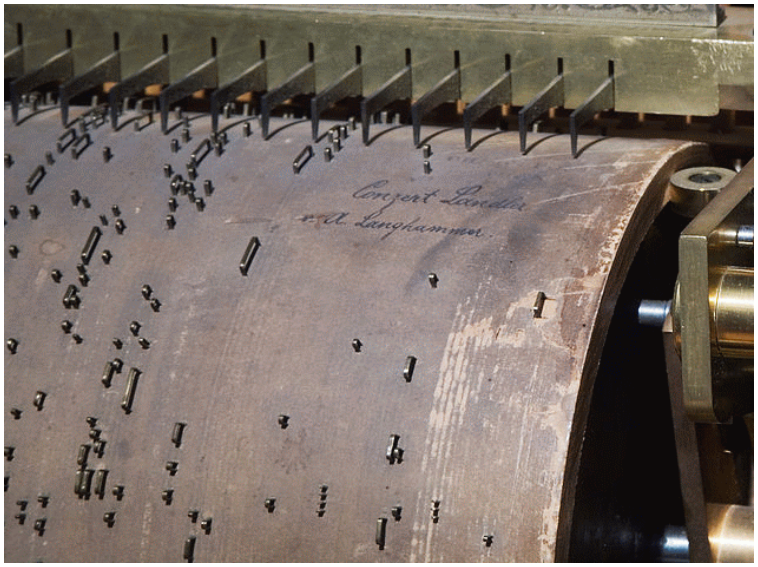


Figura 1.5: Mecanismo tipo pianola, (COMMONS, 2014b).

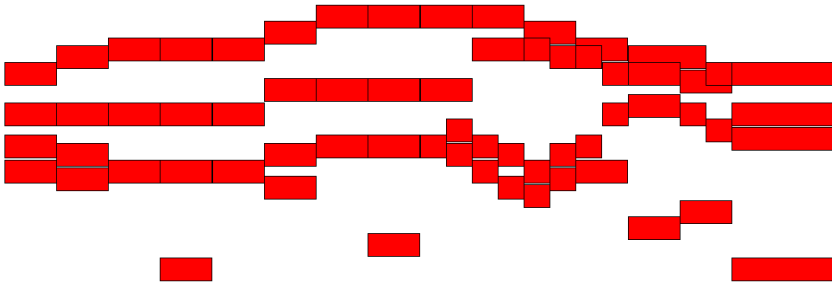


Figura 1.6: Representación tiempo-altura, tipo pianola.

ca, sirve como punto de partida para la organización temporal de las notas. Al tomar selecciones de eventos musicales consecutivos o simultáneos se puede extraer datos armónicos como la estructura del acorde, su inversión, su posición melódica, entre otros.

Espacio $\mathbb{N}^n$

Un acorde es un conjunto de n-notas, de modo que las armonías en una pieza musical se pueden representar como una serie de conjuntos de notas en el tiempo. Matemáticamente, a cada nota de una afinación de temperamento igual se le asigna un número de acuerdo con la numeración MIDI. Donde *do* central equivale al 60, *do*$\sharp$ y *re*$\flat$ a 61, *re* a 62, etc. Al traducir las notas a números, se

pueden realizar representaciones geométricas de los acordes, primero veamos cómo sería la de un intervalo. Para obtener su coordenada, se comienza a leer numéricamente el acorde de abajo hacia arriba. Por ejemplo el intervalo *(re,fa)* en notación numérica sería (62, 65), que se representa como un punto en el plano $\mathbb{N}^n$, como se muestra en la figura (1.7).

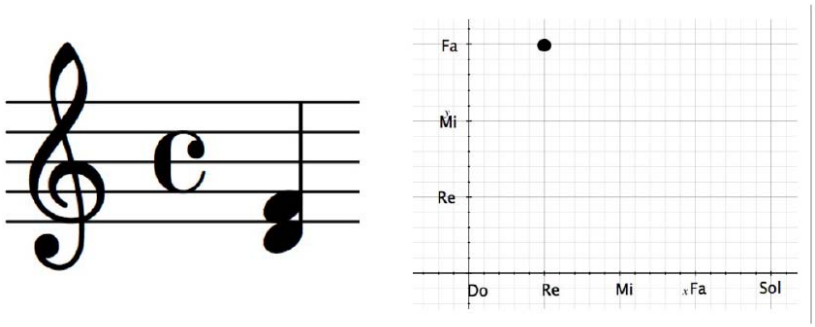


Figura 1.7: Espacio $\mathbb{N}^n$

De forma análoga, un acorde de tres notas *(do, mi, sol)*, tendría la representación (60,64,67) en un espacio de tres dimensiones $\mathbb{N}^3$. En general los acordes de *n*-notas se representan en un espacio $\mathbb{N}^n$ de n-dimensiones, a cada elemento de este espacio se le conoce como *n-adas*, (TYMOCZKO, 2011).

Uno de los beneficios de utilizar esta representación es que la noción de la conducción de voces entre acordes se visualiza como cercanía entre puntos. Por ejemplo, la idea del movimiento melódico de las voces significa moverse entre puntos cercanos en el plano. Estos espacios son en general de 3 o 4 dimensiones, dependiendo del número de voces del acorde, por lo que su visualización es complicada, sin embargo lo interesante de ellos es que nos permite medir distancias entre acordes. No es necesaria una representación gráfica para utilizarlos ya que sólo es necesario tener definidas las funciones que nos permiten calcular la cercanía entre acordes, estas se implementan en el programa de cómputo para ayudarnos en el etiquetado de acordes y de regiones tonales.

Existen miles de agrupaciones diferentes de notas y coordenadas geométricas, pero en general se clasifican de acuerdo con su estructura simplificada. Para dar una idea de la gran cantidad de colecciones que existen, si tomamos las 88 notas del piano y buscamos todas las combinaciones de 4 notas, con repetición, tenemos un total de $88^4 = 59,969,536$. Sin embargo, dada la equiva-

lencia por octavas, muchas de ellas son versiones transpuestas o repetidas entre sí, por lo que si queremos ver los distintos tipos de acorde, sólo debemos tomar el caso representativo de las 12 notas cromáticas, con lo que obtenemos $12^4 = 20,736$. Esta simplificación nos permite clasificar los acordes de forma más sencilla, ya que el espacio de trabajo se reduce de millones de posibilidades, a sólo miles.

La simplificación es aun mayor, ya que como se mencionó anteriormente, en la música tonal los tipos o familias de acordes que se utilizan son un conjunto muy pequeño de acordes. Los más usados son el acorde Mayor $(0,4,7)$, Menor $(0,3,7)$, Disminuido $(0,3,6)$, Aumentado $(0,4,8)$, Dominante $(0,4,7,10)$ y una lista que no pasa de cien distintos tipos. Todo esto se implementará en el segundo capítulo.

Enteros módulo doce y el Círculo cromático

Un concepto fundamental en el estudio de la armonía es la equivalencia por octavas, donde agrupamos las notas por familias: *do , do♯, re, re♯,...,la, la♯, si*. Para traducir esta idea musical a términos matemáticos es necesario introducir el concepto de *relación de equivalencia* y *clases de equivalencia*. Una relación de equivalencia es una forma de asociar elementos de un conjunto que comparten una característica o propiedad. En este caso podemos decir que dos notas están relacionadas entre sí están separadas por n-octavas. En esta forma, todos los *re* estarían relacionados entre sí, de igual forma todos los *fa♯*, y así para las demás notas, pero ningún *re*, estaría relacionado con algún *fa♯*. A cada una de estas familias de notas que se relacionan entre sí se le conoce como *clase de equivalencia*, de este modo para el conjunto de todas las notas, tendríamos doce clases de equivalencias. Una propiedad muy importante es que una relación de equivalencia induce una partición en el conjunto, es decir, todas las notas del conjunto deben pertenecer a una y sólo una clase y ninguna clase comparte elementos con otra. De acuerdo con lo anterior, el conjunto cromático de todas las clases de notas es:

$$(0,1,2,3,4,5,6,7,8,9,10,11),$$

y todas las demás son versiones transpuestas por octavas. Por ejemplo las notas

$$(0, 12, 24, 36, 48, 60, ...)$$

representan la familia de las notas *Do*, y así para cada una de las doce. A éste conjunto de familias se le conoce como: *Enteros módulo doce* y tiene la notación Z/nZ. Su representación geométrica suele ser un círculo como el que se ve en la figura (1.8), donde cada nota es un punto y se pintan las notas representantes a cada nota del acorde. El beneficio de utilizar esta representación geométrica es que sin importar las voces duplicadas, ni el registro de cada una de las notas, se tiene una representación única de acuerdo con su armonía. Por ejemplo, los acordes *(Do, Fa, La)*, $(60, 65, 68)$, $(60, 65, 68, 72)$, $(0, 5, 8)$ tienen la misma figura asociada. Las líneas representan el intervalo entre un par de puntos en el círculo y nos ayudan a visualizar mejor las estructuras que aparecen, por ejemplo, es interesante que los acordes mayor y menor ambos representan triángulos semejantes, mientras que los acordes de séptima disminuida son cuadrados y los acordes aumentados triángulos equiláteros. Los colores de las líneas sirven para identificar de manera visual los intervalos: segunda menor y el tritono son rojo, segunda mayor y séptima menor son violeta, terceras y sextas mayor menor son verde, por último los intervalos de cuarta y quinta justa son azules. Se eligieron estos colores para representar visualmente su disonancia percibida, rojo y violeta son los más disonantes, verde es consonancia imperfecta mientras que azul es consonancia perfecta[5].

Esta visualización es muy utilizada en programas de representaciones musicales, ha sido desarrollada y utilizada por diversos autores como un punto de partida para entender las estructuras interválicas de acordes y tonalidades desde un punto de vista geométrico. Por ejemplo, en las investigaciones de TOUSSAINT (2010) y TYMOCZKO (2011), se utiliza el círculo cromático para entender la conducción de voces entre acordes. Por otra parte en el trabajo de JOHNSON (2008), se utiliza para entender la estructura de acordes y escalas por medio de las figuras que aparecen al ser representadas en él. A su vez, en las investigaciones de MANDEREAU y cols. (2011), hacen uso de

[5]En algunos casos la cuarta se considera como una consonancia imperfecta, pero para este trabajo la entenderemos como el intervalo inverso de la quinta y la trabajaremos como consonancia perfecta

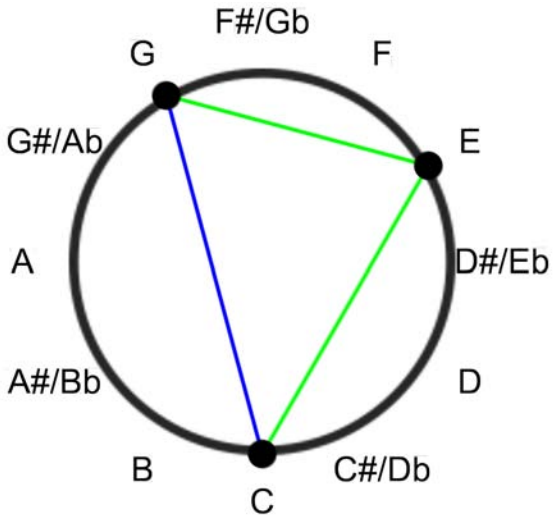


Figura 1.8: Círculo cromático

él para encontrar relaciones de homometría entre acordes.

El círculo cromático será una de las visualizaciones principales que estará presente en el programa de cómputo del que hablaremos en capítulos posteriores, ya que ofrece una forma visual de comparar si dos acordes son equivalentes y es muy útil para mostrar la conducción de voces entre armonías.

Banda de Moebius

Como se mencionó anteriormente, los pares de notas se pueden representar como puntos en un plano. Sin embargo, debido a la equivalencia por octavas y la correspondencia entre intervalos donde las voces se encuentran intercambiadas, por ejemplo si tomamos en cuenta que *(do, fa)* es equivalente a *(fa, do)*, el plano de dos dimensiones se puede reducir a un espacio representativo más pequeño. La *banda de Moebius* es una superficie de dos dimensiones que tiene un sólo lado y una sola orilla, la cual se muestra en la figura (1.9) y su representación en un plano se puede ver en la figura (1.10). Dado que cada punto es un intervalo de notas, para representar un acorde de más de ellas, necesitamos seleccionar dos o más puntos, según sea el caso. Para visualizar cómo es que la figura en el plano corresponde a la figura en tres dimensiones, imaginemos que el plano es una tira de papel rectangular y vamos a cerrarla en un cilindro pegando las orillas izquierda con

la derecha, sin embargo al examinar los elementos que se encuentran en dichas orillas, podemos ver que se encuentran en dirección contraria, por ejemplo, el punto que contiene el intervalo [FG] del lado superior derecho correspondería al [GF] en el lado inferior izquierdo, de modo que en lugar de cerrar la banda en un cilindro, debemos torcerla para que correspondan los elementos, de este modo obtenemos la figura (1.9). El uso de esta banda podría resultar interesante para la visualización de armonías, esto se implementa en el programa de cómputo para ver su utilidad en el uso cotidiano.

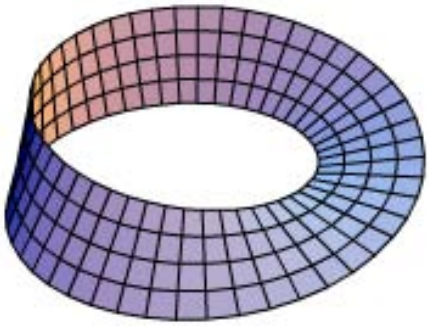


Figura 1.9: Imágen banda de Moebius, (COMMONS, 2014a)

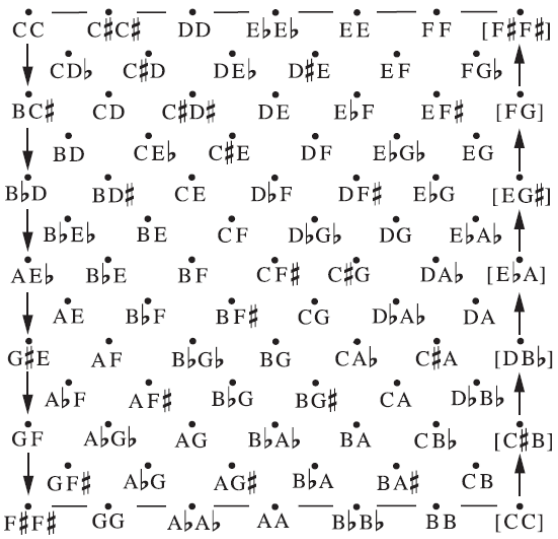


Figura 1.10: Plano de notas en dos dimensiones con orillas referenciadas que forman una banda de Moebius. (TYMOCZKO, 2011)

Grupo PLR - Tonnetz

Como parte de la teoría neo-riemaniana[6] se han definido algunas operaciones que permiten analizar ciertas relaciones entre los acordes. En términos matemáticos se trata de un *grupo* que relaciona los veinticuatro acordes mayores y menores por medio de tres operaciones: *P, L, R*. La primera operación *P* lleva una triada mayor(menor) a la triada opuesta menor(mayor) con la misma fundamental, es decir:

$$P(DoM) = (Dom).$$

$$P(Dom) = (DoM).$$

La segunda operación *L* funciona de la siguiente manera, si se le introduce una triada mayor, obtendremos como resultado la triada menor cuya fundamental se encuentra a cuatro semitonos arriba de la original. Por otra parte, si se le introduce una triada menor, se obtiene la triada mayor cuya fundamental se encuentra cuatro semitonos debajo de la original. Por ejemplo:

$$L(DoM) = (Mim),$$

$$L(Mim) = (DoM),$$

El nombre viene del inglés *Leading Tone*, dado que la nota fundamental del primer acorde se cambia un semitono hacia arriba o hacia abajo, según sea el caso.

La tercera y última operación *R* lleva una triada mayor a la triada menor que se encuentra tres semitonos debajo de la original, y en el caso de que se le introduzca una triada menor, obtendremos como resultado la triada mayor que se encuentra tres semitonos sobre la original.

[6]Conjunto de ideas que surgen de las teorías de Hugo Riemann (RIEMANN, 1896), donde entre otras cosas se busca relacionar a las triadas entre sí, sin necesidad de una referencia directa a una tonalidad y para ello se definen operaciones entre acordes, las cuales se veran a continuación (COHN, 1998).

$$R(Do\,M) = La\,m,$$

$$R(La\,m) = Do\,M,$$

En general la operación *R* lleva a una triada mayor a su relativa menor y una triada menor a su relativa mayor.

Es interesante notar que cada una de estas operaciones es su propia inversa, es decir, si se aplica dos veces seguida la misma operación, se regresa al acorde del cual se partió. Al aplicar estas tres operaciones en distintas combinaciones, se puede encontrar un camino entre cualquier par de triadas. En la figura (1.11) se muestra un ejemplo de camino entre triadas siguiendo estas operaciones, las cuales se realizan en la tabla (1.2).

$R(Do\,M) = La\,m$
$L(La\,m) = Fa\,M$
$R(Fa\,M) = Re\,m$
$L(Re\,m) = si\flat\,M$

Tabla 1.2: Ejemplo de operaciones del grupo *PLR*.

Podemos encontrar una explicación detallada de estas operaciones, incluyendo las definiciones formales y explícitas, ver el trabajo de CRANS y cols. (2009). El grupo de operaciones *PLR* está diseñado sólo para triadas mayores y menores, sin embargo, en la investigación de POPOFF (2013), se pueden extender a acordes de cuatro notas.

Por otra parte, este grupo tiene una representación geométrica muy interesante. Para construir el diagrama se toma una nota inicial en el centro de un hexágono, por ejemplo *Do*, y se crean seis vértices. En el eje horizontal, se colocan las notas que se encuentran a una distancia de una tercera mayor: *La*♭ a la izquierda, *Mi* a la derecha. En el eje que se encuentra en un ángulo de 60 grados se colocan las notas que están a la distancia de una quinta justa, ascendente es hacia arriba: *Sol*, descendente es hacia abajo: *Fa*. Por último en el eje de 120 grados las notas que están a una tercera menor de distancia, ascendente hacia arriba: *E*♭, descendente hacia abajo: *La*. Al repetir esto para

todas las notas se crea un diagrama conocido como *Tonnetz*, en el que cada triángulo es una triada mayor o menor, algunas dispuestas hacia arriba y algunas hacia abajo. Las aristas que conectan a cada triada son las operaciones *P, L y R*, la orientación varía de acuerdo con la disposición de cada triángulo, si está con la punta hacia arriba siempre será una triada mayor, la operación de la base será *R*, la del lado izquierdo será *P* y el derecho *L*; por otra parte si el triángulo se encuentra con la punta hacia abajo la triada será menor, la operación del lado horizontal que es el de arriba, será *R*, el lado izquierdo será *L* y el derecho *P*.

El *Tonnetz* tiene una representación tridimensional en forma de una dona, también conocida como *toro*, esto se puede ver en la figura (1.13), (TYMOCZKO, 2011). Por otra parte el camino de operaciones que se muestra en la partitura de la figura (1.11), se puede visualizar en el *Tonnetz* de la figura (1.12). Este tipo de representaciones se han utilizado en diversas aplicaciones de análisis armónico y melódico, (HEDGES y MCPHERSON, 2013; BIGO y cols., 2013).

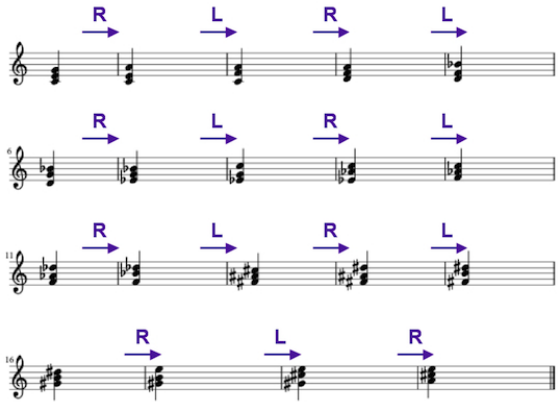


Figura 1.11: PLR pentagrama, (POPOFF, 2014).

Algunos investigadores prefieren una disposición del tonnetz en forma de espiral, donde se inicia en una nota fija, digamos Si bemol, y se comienza a girar de modo en dirección de quintas y cuando se vuelva a encontrar justo encima del *Si bemol* inicial debemos estar a una distancia melódica de una tercera mayor, (CHEW, 2008). La representación en forma de espiral no será utilizada en este proyecto ya que considero que la versión de rejilla que mostramos en la figura (1.12) es más fácil de visualizar.

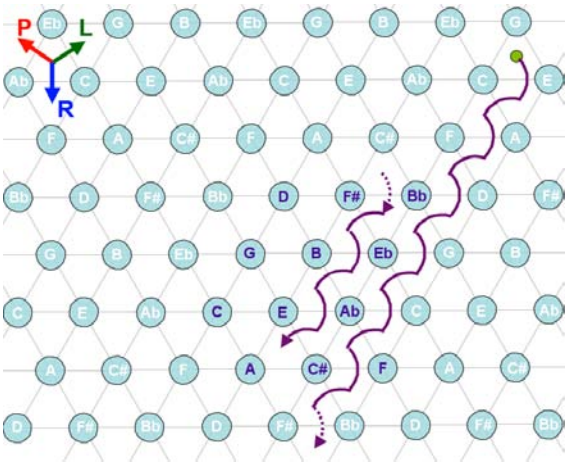


Figura 1.12: PLR tonnetz, (POPOFF, 2014)

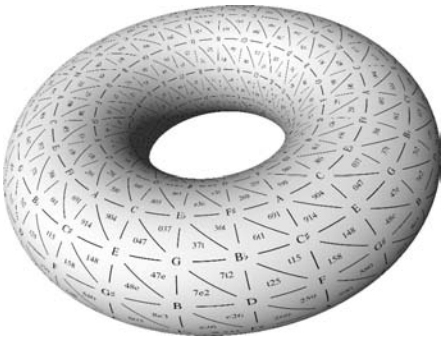


Figura 1.13: Representación del tonnetz tridimensional. (ENDOLITH, 2010)

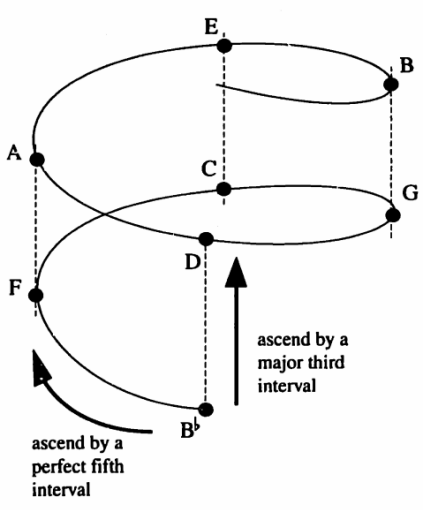


Figura 1.14: Representación del tonnetz en una espiral. (CHEW, 2000)

Teoría de grafos

La teoría de grafos permite representar las relación entre un número discreto de elementos en la forma de vértices y áristas. En estos diagramas, no importa la posición exacta de los elementos, ni las líneas, sino cuáles y de qué forma están conectados. En trabajos como el de WALTON (2010), se utilizan los vértices para identificar cada una de las veinticuatro tonalidades mayores y menores. Cada una de ellas se conecta con una arista si existe un acorde pivote, común, entre ellos. La gráfica resultante de este proceso se puede ver en la figura (1.15).

Esta representación armónica de grafos es la más importante para el presente trabajo, ya que utilizaremos una variante del trabajo de WALTON (2010), para construir un espacio que muestre la posición y función armónica de cada uno de los acordes de la pieza a analizar. Esto lo realizaremos por medio del modelo matemático que se desarrolla en el segundo capítulo.

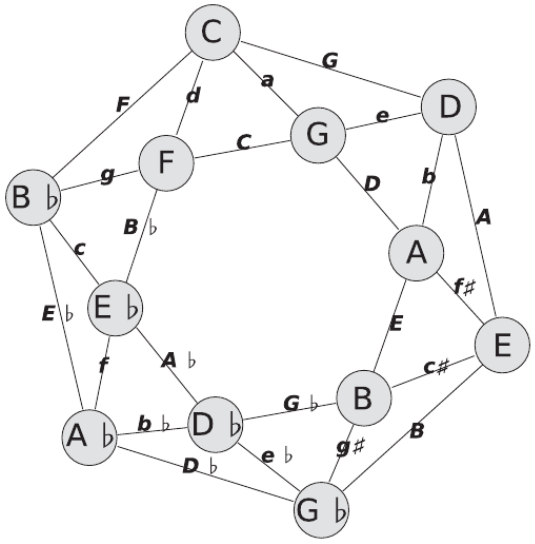


Figura 1.15: Tonalidades y acordes pivote en un grafo, de acuerdo con (WALTON, 2010)

Teoría generativa de la música tonal

Lerdhal y Jackendoff en 1985 desarrollaron un sistema de análisis y representación musical que establece una analogía entre la música y los lenguajes formales (LERDAHL y JACKENDOFF, 1985). Su idea básica es modelar cómo el humano percibe la música no como eventos aislados, sino como

agrupaciones coherentes que siguen una estructura similar a la de un lenguaje. Para analizar una pieza utilizan niveles jerarquicos que toman en cuenta cuestiones de rítmica, repetición y estructura armónica, entre otros. Este trabajo, a diferencia de las anteriores representaciones geométricas no busca necesariamente una visualización de las estructuras musicales, sino que busca dar sentido a lo que de otra manera serían eventos sonoros sin conexión. Podría ser relevante para nuestra investigación ya que si tenemos una serie de acordes, podemos utilizar los resultados de Lerdhal y Jackendoff para deducir las posibles funciones tonales, sin embargo no se implementa en esta versión del programa.

Ejemplos de visualizaciones

En las tablas (1.3) y (1.4) se muestran visualizaciones de algunos acordes en tres distintas geometrías: el *Círculo Cromático*, la *Banda de Moebius* y el *Tonnetz*, las cuales fueron elegidas para implementar en este programa de cómputo por ser las que preservan la geometría de los acordes bajo traslación e inversión. Cada renglón muestra un acorde distinto y se presentan en el siguiente orden *CM, Em, C+, Bo, Bo*7 *y GD7*. Como se mencionó anteriormente, cada tipo de acorde tiene una figura geométrica única asociada, la cual se preserva sin importar la nota fundamental, lo único que cambia es su orientación. Tomemos el acorde *CM* como ejemplo, su figura en el *Círculo Cromático* es un triángulo con la proporción de lados que se ve en la tabla (1.3); si quisiéramos construir el acorde *FM*, bastaría rotar el acorde hasta que el vértice que antes ocupaba la nota *C* ahora corresponda la nota *F*. En el caso del *Tonnetz* o de la *Banda de Moebius* las figuras se trasladan a la nota fundamental en la que se desee construir el acorde.

Una observación importante es la distinción geométrica de los acordes mayores y menores. A simple vista son muy similares ya que en el *Círculo Cromático* ambos presentan triángulos semejantes, sin embargo al analizarlos más detalladamente vemos que no se puede llegar de uno a otro por simple traslación, es necesario girarlos sobre alguno de sus ejes para conectarlos. De igual manera, en la *Banda de Moebius*, para cambiar de un acorde mayor a menor, es necesario invertir con respecto al eje horizontal, ya que los acordes mayores tienen su punta hacia arriba, mientras

que los menores hacia abajo.

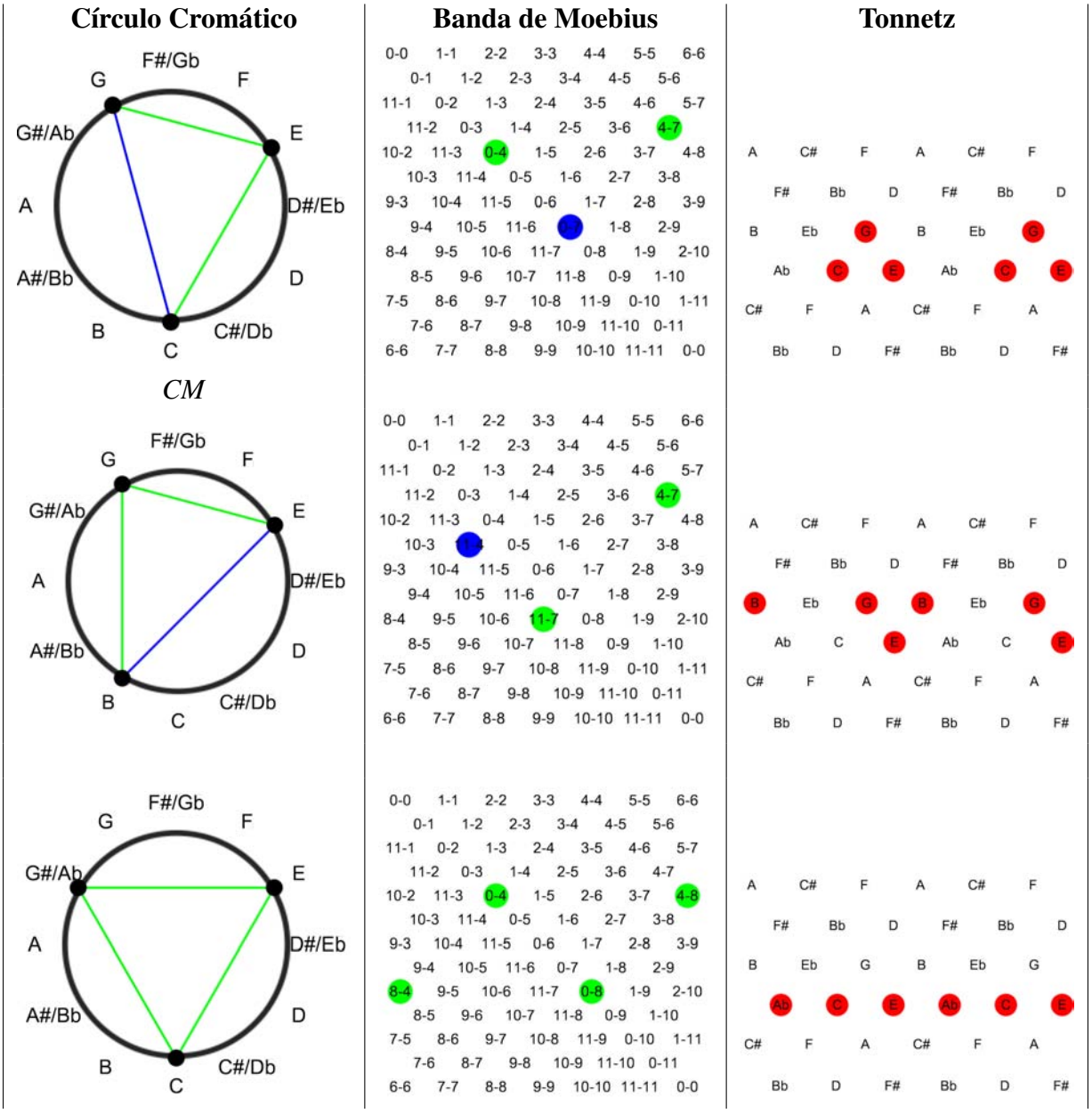


Tabla 1.3: Acordes y sus representaciones gráficas. *Do M, Mi m, Do +, Si dism*.

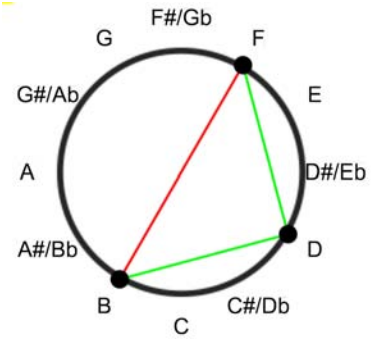
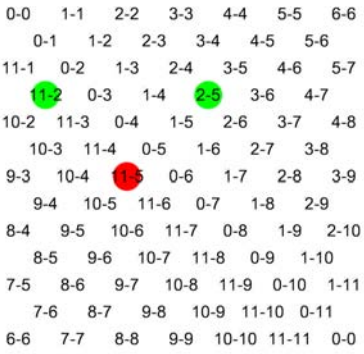
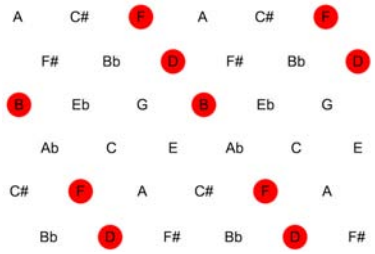
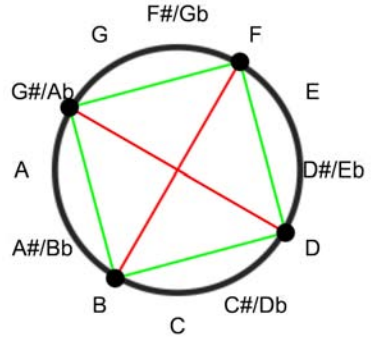
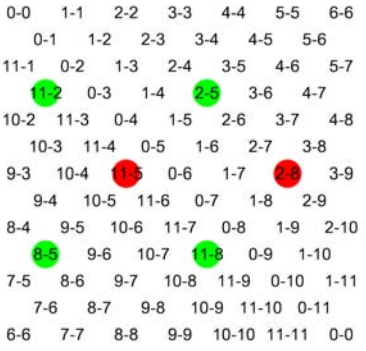
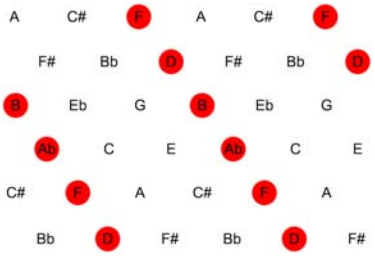
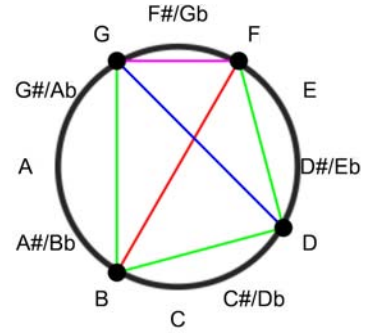
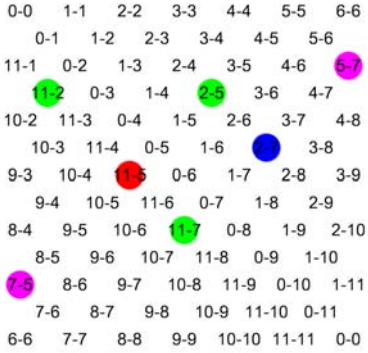
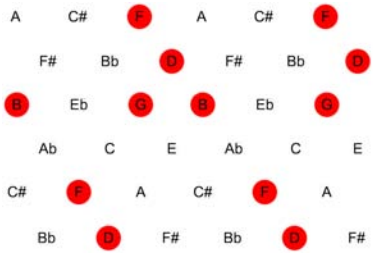
Círculo Cromático	Banda de Moebius	Tonnetz
		
		
		

Tabla 1.4: Continuación de acordes y sus representaciones gráficas. *Si dim* y *Sol Dom7*.

Propuesta de trabajo

Basado en los conceptos de traducción numérica de las notas y acordes, en el siguiente capítulo se propone una estructura matemática que organiza y clasifica las relaciones tonales de todos los posibles acordes, de modo que funciona como un mapa abstracto que permite conectar distintas regiones armónicas. A este espacio lo llamaremos *Universo Tonal* y es la base del programa de cómputo desarrollado en el capítulo tercero. A su vez, en el programa de cómputo, se implementan cada una de las geometrías aquí mencionadas: La Pianola/partitura, el plano/banda de Moebius, el círculo cromático, el tonnetz y el grafo de tonalidades, para probar su utilidad en el análisis de archivos musicales digitales en formatos MIDI y MusicXML.

1.4. Técnicas algorítmicas utilizadas

Existen trabajos que han abordado el análisis musical algorítmico desde distintos enfoques. Algunas investigaciones analizan archivos de audio y otras analizan archivos MIDI. Esta es una distinción importante ya que la diferencia es equivalente a si un humano analizara una pieza musical al escucharla directamente, contra analizar la partitura. Considero que el análisis de audio no es completamente eficiente de momento, ya que los métodos de extracción de información polifónica de distintos instrumentos actualmente no son muy confiables, en cambio los archivos MIDI y MusicXML resuelven en parte este problema, (CUNNINGHAM, 2004), ya que tenemos toda la información musical en un formato numérico muy accesible, del cual existen muchos repositorios de archivos digitales a los cuales se puede acceder fácilmente por medio de la Internet. Por esta razón nosotros realizaremos el análisis sobre archivos con estos formatos.

La extracción de información musical de archivos digitales es parte del área de investigación llamada *Music Information Retrieval*, donde se aplican distintos tipos de técnicas y métodos acústicos, algorítmicos y de inteligencia artificial para encontrar los datos buscados, (DE HAAS, 2012). Como ejemplos de algoritmos desarrollados para la extracción de contenido musical de archivos digitales tenemos varios ejemplos, entre ellos el trabajo de FUJISHIMA (1999), que extrae informa-

ción armónica de sonidos con texturas orquestales, o bien el de PARDO y BIRMINGHAM (2002), donde se crean algoritmos de optimización para detectar cuándo existe un cambio de acorde en la pieza y se comparan con un diccionario de armonías predefinidas. Por otra parte, en la investigación de DE HAAS y WIERING (2013), van un paso más allá, ya que les interesa la función armónica de cada acorde. Para extraer y analizar la información utilizan conceptos de *lenguajes formales* y análisis Schenkeriano, donde se propone una jerarquización de acordes de acuerdo con su función tonal. Su principal limitante es que sólo pueden trabajar con una tonalidad y su paralela a la vez, de modo que es imposible analizar movimientos de modulación. Es curioso que principalmente se han creado algoritmos para el análisis de corales de Bach, como en el trabajo de KRÖGER y cols. (2008), y música de *Los Beatles* como en las investigaciones de HARTE y SANDLER (2005) y SHEH (2003), como si fueran los emblemas de la música académica y la música popular, respectivamente.

Un punto común de los diversos sistemas de análisis armónico de un archivo de audio o MIDI es el problema de la notación que utilizan para los eventos musicales. Debido a que hay diversos tipos de notación dependiendo de la época y el género, no hay una simbología universal. Por otro lado, la implementación algorítmica de cualquier sistema requiere un lenguaje que pueda representar de manera única cada uno de los elementos y que no se preste a ambigüedades ni falsas interpretaciones. Este problema se aborda en el trabajo de HARTE y cols. (2005), en el cual establecen un sistema de notación que es fácil de leer por un músico con entrenamiento formal, pero es sencillo y sin ambigüedades en el contexto de programación. Primero contrastan y comparan los puntos fuertes y débiles de algunas de las notaciones armónicas más comunes, entre ellas el bajo cifrado, notación del análisis clásico (grados romanos) y la utilizada por el jazz y la música popular. La notación que obtienen es muy útil en el sentido de que cualquier acorde se puede expresar como una secuencia sencilla de letras y números.

Como ejemplo de las ambigüedades que pueden surgir y la necesidad de establecer claramente los significados de los símbolos, veamos un caso en el cual la notación es dependiente del contexto. En el caso de los grados romanos, I_7 representa un acorde de séptima mayor, mientras que en la

notación Jazz-Pop un CD_7 o simplemente C_7 es un acorde dominante de séptima con base en la nota *do*. En este caso el modificador 7 indica que el acorde tiene una estructura distinta, en el primer caso la estructura del acorde es $baseTonalidad + (0,4,7,11)$, mientras que un acorde de dominante en la misma nota sería $baseTonalidad + (0,4,7,10)$. El 7 puede corresponder a una séptima mayor o menor, dependiendo de la función tonal de cada acorde, asimismo se requiere saber en qué tonalidad se trabaja para que los números romanos se relacionen con un acorde en particular. En cambio en la notación de jazz, el acorde está identificado inequívocamente por su fundamental y su modificador, de este modo, en la tonalidad *Do mayor* el acorde I_7 se escribiría como CM_7 mientras que el dominante sería G_7.

Otros sistemas utilizados tratan de apegarse a convenciones y estándares, como es el caso de *Common Music Notation* (SCHOTTSTAEDT, 1997), sin embargo, dado su poco uso, en realidad terminan siendo todos casi igual de arbitrarios. Los trabajos de investigación que he encontrado no hacen mucho énfasis en la notación de los acordes y presuponen que el músico está familiarizado con todas ellas, sin embargo, se debe hacer una distinción acerca de cuándo se elige una notación por la facilidad de implementación en algoritmos, y cuándo se elige otra para facilidad de lectura del usuario. En este trabajo de tesis nos interesan los dos, aunque para distintas cosas. Por una parte, necesitamos una representación de los acordes en forma de datos para que la computadora pueda realizar los cálculos necesarios, sin embargo, para la interfaz gráfica del usuario necesitamos una notación que sea de fácil lectura, para que pueda visualizar los resultados, como lo sería la notación de grados romanos. Parte de los objetivos de este programa es que pueda ser utilizado por músicos aficionados a cuestiones científicas, o bien científicos con aficiones musicales. Debido a esta mezcla tan particular para que sea útil para ambos tipos de usuario, es necesario que las notaciones utilizadas sean precisas, es decir los términos musicales deben estar expresados en la notación habitual para un músico, como lo sería la notación de armonía gradual o de Jazz/Pop, por otro lado los conceptos numéricos y matemáticos, deben ser consistentes. Es imposible el predecir al usuario exacto que tendrá este programa, lo mejor que se puede hacer es denotar los objetos con las notaciones correctas y precisas para evitar confusiones.

Otra consideración importante que se hace en este tipo de investigación, es realizar suposiciones acerca de cuales son las progresiones armónicas más comunes y con base en eso realizar inferencias del significado funcional armónico de un fragmento musical. Esta idea aparece en el trabajo de YOSHIOKA y cols. (2004), donde se obtienen buenos resultados, sin embargo sólo son válidos para tonalidades mayores, sin modulaciones. En este sentido uno de los enfoques más completos es el implementar una biblioteca de patrones, reglas y relaciones armónicas que sirva como referencia para abordar el tema de una forma más general. En el trabajo de TAUBE (1999) se propone crear un *Núcleo Analítico*, donde se abstraen todos estos conceptos y se utilizan para el análisis de los corales de Bach.

Al intentar sistematizar y codificar las reglas de la música, es tentador querer hacer un tratado musical computacional que sirva para sentar las bases del análisis musical moderno, como lo hacen en el trabajo de MOUTON (1995). Sin embargo, sus resultados dejan claro que al tratar de implementar tratados clásicos de armonía, existen muchos huecos en cuanto a definiciones claras, lo cual dificulta la implementación rigurosa en un programa de cómputo.

Un punto común de los trabajos que intentan asignar funciones tonales a colecciones de notas verticales, es el concepto de una *matriz de tonalidades*, que es una tabla en la cual se enlistan todos los posibles significados de un acorde, en las distintas tonalidades mayor y menor. Esto permite tener una visión general de las distintas regiones por las cuales una pieza transita, (KRÖGER y cols., 2008; MOUTON, 1995).

Por último cabe mencionar el trabajo de David Cope, (COPE, 1996), por medio del análisis de cientos de datos MIDI agrupados por compositores, utiliza algoritmos para encontrar características de estilos individuales y poder sintetizarlas para recrear obras en un estilo similar.

Siguiendo los puntos fuertes de las investigaciones anteriores, el modelo matemático que se desarrolla en el segundo capítulo utilizará los conceptos de *Diccionario de acordes* y *Matriz de tonalidades*, ya que son lo suficientemente flexibles como para analizar un gran conjunto de piezas.

1.5. Aplicaciones de cómputo para el análisis armónico musical

Uno de los objetivos fundamentales de este trabajo es facilitar el análisis armónico de piezas musicales. Para ello se trabaja en la interdisciplina de las matemáticas, la música y la ingeniería de cómputo, con la intención de que el usuario final tenga una herramienta que complemente su experiencia musical y le ayude a comprender otras formas de analizar y estructurar una pieza. Con la intención de que el resultado de este trabajo sea innovador, es necesario hacer un breve recorrido y descripción de los programas computacionales que existen actualmente para abordar el problema del análisis y reconocimiento armónico de una pieza.

Así como se han desarrollado algoritmos para el análisis musical, también se han creado aplicaciones de cómputo que sirven como herramienta para músicos. Son variados los enfoques de las mismas y en esta sección se hace un análisis de las funciones y las limitaciones de algunas de las que existen actualmente. El análisis musical es un proceso complejo que requiere de la inteligencia e intuición humana para su uso correcto; a su vez, implica una serie de procesos repetitivos que pueden beneficiarse por una herramienta de cómputo. Entre las aplicaciones y ambientes computacionales que intentan satisfacer esta necesidad se encuentran: Open Music, Liquid Notes, Mapping Tonal Harmony Pro, entre otros. Cada uno tiene sus beneficios y desventajas, las cuales se describen a continuación.

Open Music

Uno de los ambientes de análisis y composición más completos es *Open Music*, un lenguaje de programación desarrollado por el IRCAM, que hace uso de estructuras matemáticas para organizar el contenido musical. Es altamente especializado y tiene una curva de aprendizaje muy alta (ANDREATTA y AGON, 2003), es un lenguaje de programación visual orientado a objetos diseñado para la composición musical. Es un ambiente muy complejo donde las partes musicales se ensamblan por medio de iconos conectados por líneas, se crean una especie de diagramas de flujo donde los nodos representan funciones y los aristas el flujo de los comandos y variables. En este progra-

ma se implementan clases que utilizan algoritmos matemáticos para organizar las ideas musicales; sin embargo, es altamente especializado y más que un software, es un ambiente de programación. Existen implementaciones de algorítmos de análisis armónico para OpenMusic, entre ellas tenemos la Harmonic Analysis creada por *Carlos Pérez*, (PÉREZ, 2015), de la Universidad Alicante, el cual es una adaptación de los algoritmos creados en (PARDO y BIRMINGHAM, 2002). También hay librerías que permiten hacer un análisis estadístico de la información melódica y armónica del archivo, como *SOAL*, (IRCAM, 2015).

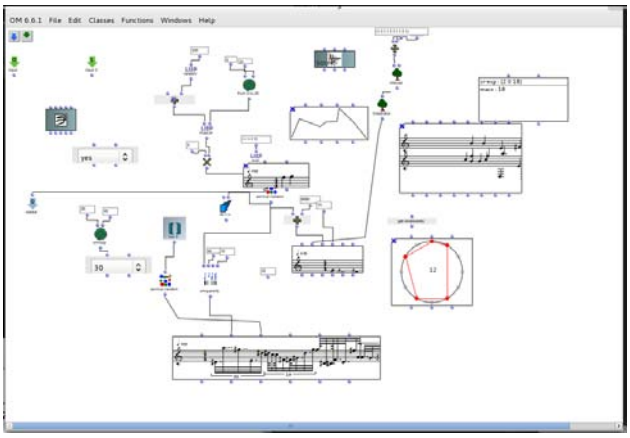


Figura 1.16: Ambiente visual de OpenMusic

La aplicación de cómputo elaborada en esta tesis podría implementarse en OpenMusic; sin embargo, el ambiente gráfico resultaría insuficientemente poderoso o versátil para crear las interfaces gráficas necesarias para un usuario no familiarizado con ambientes de programación. En las figuras (1.16) y (1.17) se muestran ejemplos de este ambiente gráfico.

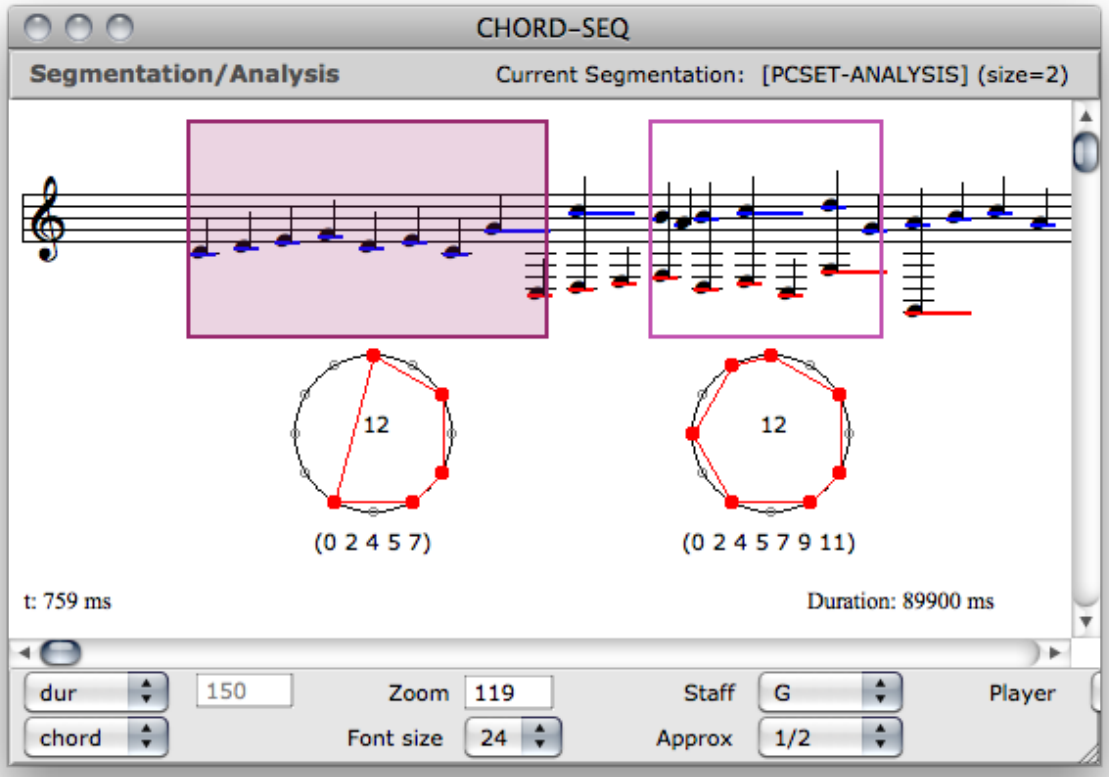


Figura 1.17: Análisis de segmentos en Open Music

MidiTonality

MidiTonality fue desarrollado de manera independiente, (SPIER, 2015). En él se utiliza la notación de tipo pianola para analizar las armonías que surgen de archivos MIDI, en cada paso de tiempo se clasifican los acordes posibles y se crea una tabla con colores donde se trata de inferir la región tonal que puede significar. Cuenta con mucha información y realiza análisis tanto de audio como MIDI, la desventaja principal de este programa es que la documentación no es muy clara y el resultado visual y estructural es tan saturado que los datos útiles terminan siendo casi encriptados, como se aprecia en la figura (1.18).

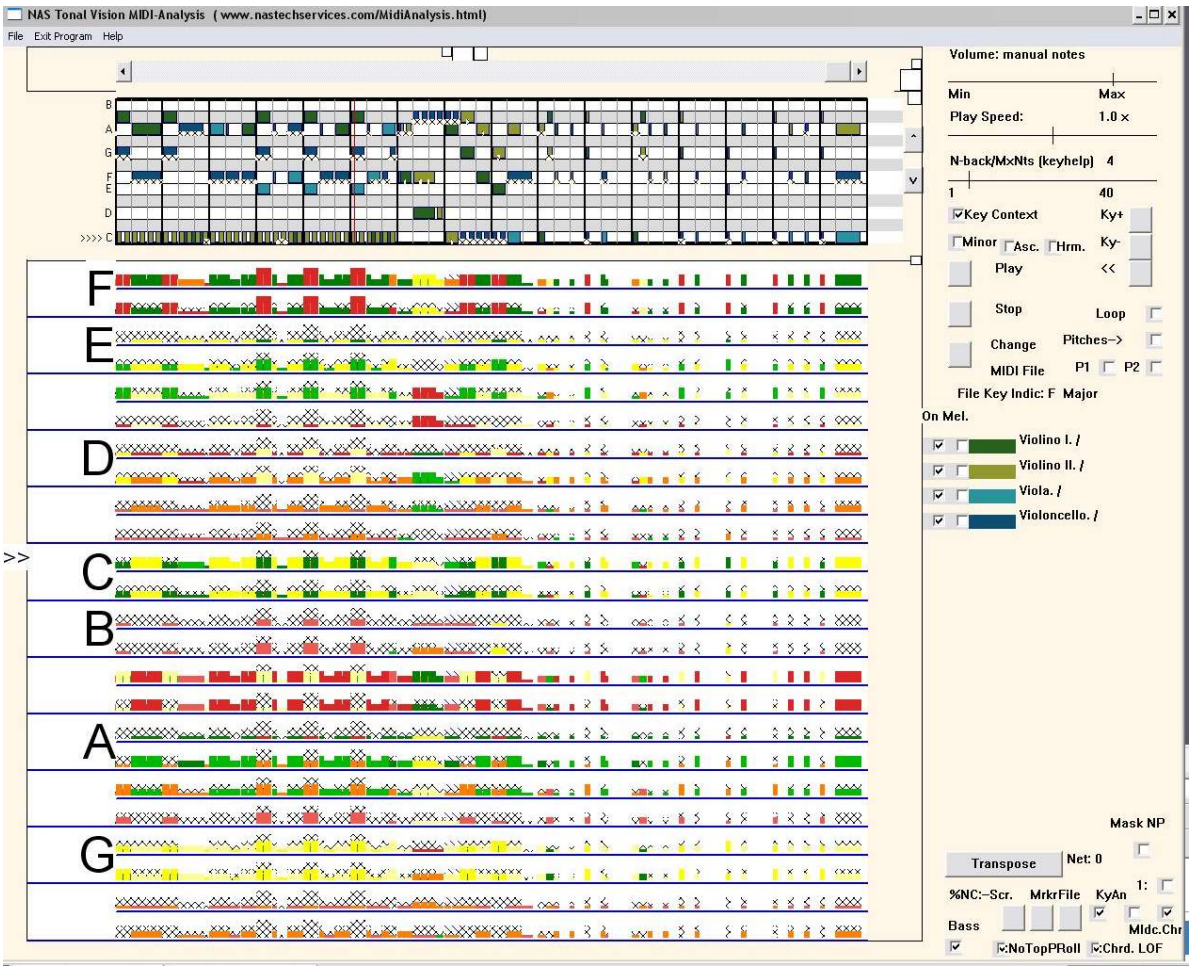


Figura 1.18: MidiTonality

Liquid Notes

Liquid Notes se promociona como un asistente armónico de composición, (RE-COMPOSE, 2015). Su principal función es tomar un archivo MIDI, analizar los acordes que aparecen y modificarlos de acuerdo con las intenciones musicales del usuario. Es decir, el software de manera automática segmenta el archivo y hace una estimación acerca de donde termina e inicia un acorde. Una vez hecho esto, le permite al usuario cambiarlo con opciones de *aumentar tensión*, *disminuir tensión*, o cambiar el acorde por alguno de otra región tonal como Dominante o Subdominante por medio de controles deslizables o botones. Utiliza la notación de pianola al igual que los anteriores. Una limitación fuerte que tiene es que sólo puede analizar acordes que aparecen en bloques de notas simultáneas, es decir “verticales”, cuando existen arpegios u otras formas de distribuir

el contenido armónico tiende a cometer muchos errores. Esto no causa demasiados problemas en muchas de las piezas de música popular, ya que los archivos MIDI suelen estar construidos con bloques de acordes. Considero que para este tipo de aplicaciones ayudaría mucho una estructura visual de tipo Tonnetz, para que usuarios no familiarizados con las regiones armónicas pudieran estructurar sus ideas musicales de forma visual.

Figura 1.19: Liquid Notes

Mapping Tonal Harmony Pro 5

Este programa no realiza análisis armónico, sin embargo tiene una visualización de las regiones armónicas que funciona como un mapa de exploración, (MDECKS, 2015). Primero se elige una tonalidad donde se trabajará y después se crea un diagrama de todos los acordes que tienen alguna relación tonal con ella, incluyendo dominantes secundarias y modulaciones a regiones cercanas. Cada vez que el usuario selecciona un nuevo acorde el programa le sugiere una serie de conducciones de voces para elegir la más adecuada para su intención musical. Me parece que esta visualización ayudaría mucho a programas como LiquidNotes y servirá de referencia para nuestro modelo de representaciones armónicas. En la figura (1.20), se muestra como ejemplo el mapa para la tonalidad *Do mayor*.

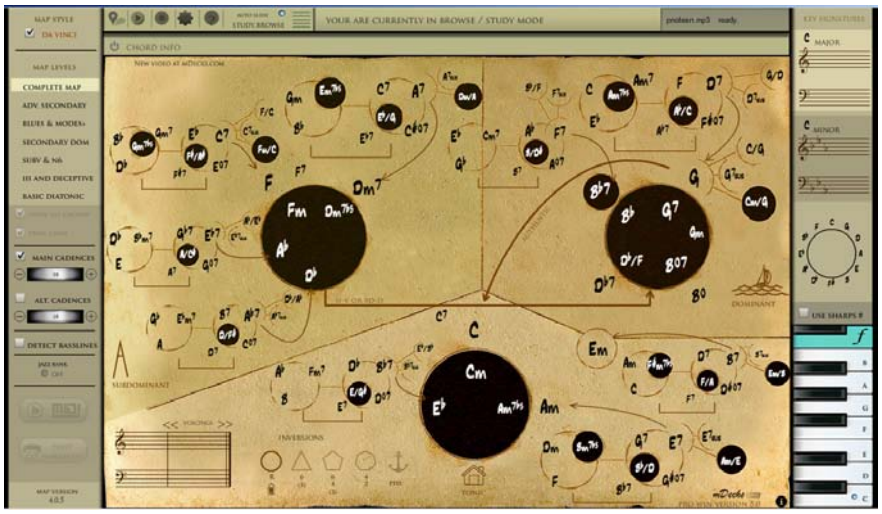


Figura 1.20: Mapping Tonal Harmony

iAnalyse 3

En iAnalyse, se busca organizar el análisis musical de modo que una partitura y una grabación de audio estén sincronizadas al interpretarse, es decir, por medio de calibraciones realizadas por el usuario se indican en la partitura ciertos puntos de tiempo que la música debe seguir, el resto se interpola automáticamente, (COUPRIE, 2015). En este programa también se pueden subrayar de distintos colores los temas y secciones, asignándoles un valor estructural del cual el programa

lleva un registro y permite visualizarlo por medio de colores y con la información sintetizada. Es interesante la función de compaginar la partitura con archivos de audio; sin embargo este programa no realiza cálculos automatizados de regiones tonales, sino que la información debe ser introducida por el usuario. Ver figura (1.21).

Figura 1.21: iAnalyse

Chord geometries

Este programa de cómputo es el que tiene más variedad de representaciones geométricas de acordes, (TYMOCZKO, 2015). Entre ellas se encuentran el: espacio lineal, geometría circular, espacio diádico, espacio diádico de clases de alturas, espacio triádico, espacio triádico de clases de alturas y espacio tetracórdico. En este programa se introducen los acordes en un teclado virtual, o por medio de comunicación con dispositivos MIDI, ya sean controladores físicos o de software. No está diseñado para el análisis de piezas, pero algunas de sus representaciones y visualizaciones son muy útiles para organizar la estructura armónica de las mismas; ver figura (1.22). Es interesante poder explorar las visualizaciones abstractas que surgen de un análisis matemático de la música, sin embargo, un problema que surge es que no todas ellas son prácticas para el uso cotidiano, ya que las armonías de una pieza muchas veces pertenecen a espacios de más de tres dimensiones, lo cual complica el trabajo. Creo que estas representaciones podrían usarse internamente sólo pa-

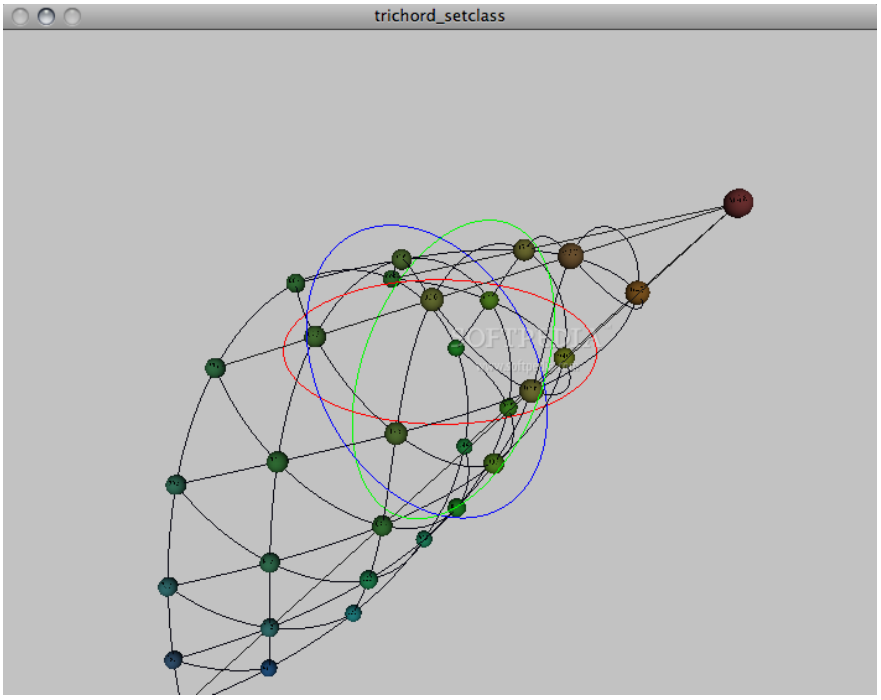


Figura 1.22: Chord Geometries

ra realizar cálculos numéricos, mientras al usuario se le presenta una notación tipo partitura con anotaciones de las etiquetas de acordes y regiones tonales.

Bibliotecas de análisis

Existen bibliotecas de código desarrolladas para analizar contenidos armónicos y musicales en formato MIDI o MusicXML, como Harmonia (JEKOVEC, 2015), Mingus (RHIJNAUWEN, 2015) y Music21, (MICHAEL, 2015). Sin embargo, ninguna de ellas es una aplicación ejecutable terminada, de modo que el usuario debe compilar el código, instalar la librería o aprender a programar. Para el fin de este trabajo, necesitamos una aplicación tipo *Standalone*, es decir que el usuario sólo deba abrirla para comenzar a analizar, sin necesidad de programar.

Visiones para el software

Con esta breve revisión de algunos de los programas de cómputo de análisis armónico disponibles actualmente, se puede ver que existe un hueco entre las aplicaciones más técnicas como Open

Music y las más comerciales como Liquid Notes, a las cuales les falta integrar las visualizaciones de mapas de regiones estilo Mapping Tonal Harmony y de Chord Geometries. Con este trabajo se implementarán nuevos algoritmos que permitirán llenar el vacío que las aplicaciones actuales han dejado, de modo que el usuario final pueda realizar operaciones complejas de análisis en una forma sencilla . En específico, los aspectos que este programa de tesis aborda que son nuevos para el análisis armónico musical por medio de computadora son:

1. Permitir al usuario seleccionar conjuntos de notas de un archivo MIDI o MusicXML y darle una etiqueta de que tipo de acorde es, así como las funciones tonales del mismo con respecto a todas las tonalidades.

2. Una vez que se tenga la selección de acordes que el usuario le interese analizar y visualizar, se puede visualizar las representaciones de estos acordes en el *Tonnetz*, la *Banda de Moebius*, el *Círculo Cromático* y el *Grafo Tonal*. La visualización es de forma interactiva, ya que al dar *play* al archivo MIDI, el acorde representado cambiará de acuerdo al compás y tiempo en el que se encuentre.

3. Permite al usuario reinterpretar las funciones tonales de una pieza y explorar los posibles resultados al hacer un cambio de tonalidad.

4. Obtener al final del etiquetado una visualización grafica del plan tonal de la pieza, por medio del *Grafo Tonal*.

Capítulo 2

Propuesta de modelo matemático: *Universo Tonal*

Para crear el programa de cómputo que se propuso en la introducción, necesitamos definir claramente las categorías musicales que se traducirán a código, para esto se construirán a continuación una serie de conceptos matemáticos que sirven de puente o enlace entre la información musical y los códigos de programación.

De manera global, necesitamos analizar conjuntos de números donde cada uno de ellos tiene asociada una nota musical, misma que puede adquirir distintas funciones tonales dependiendo del contexto. En la sección de antecedentes vimos algunas de las representaciones matemáticas que los objetos armónicos pueden tener, como el espacio $\mathbb{N}^n$, los enteros módulo doce y el Tonnetz , sin embargo, para este trabajo quiero proponer un modelo nuevo que ayude a organizar las tonalidades y sus relaciones en una representación donde se muestre de manera similar a un mapa. Se utilizarán las nociones de la teoría de grafos, ya que los diagramas resultantes muestran de manera adecuada las relaciones entre los acordes de acuerdo con las tonalidades a las que pertenecen por medio de vértices y aristas. A continuación construiremos y presentaremos paso a paso los conceptos y definiciones necesarios para esto.

2.1. Universo Tonal

Como en cualquier modelo matemático es necesario elegir las partes del sistema que se van a abstraer y analizar. En este caso, como nos interesa el análisis armónico, es necesario construir el sistema desde el nivel más básico que sea útil para nosotros. Por esta razón, comenzaremos por definir la traducción matemática de ciertos conceptos como *Nota, Acorde, Tonalidad, Función Tonal y Universo Tonal*. Al definir estos conceptos podemos sentar las bases sobre las cuales se implementa el programa de cómputo. Cada uno de los términos a definir incluye una serie de propiedades y funciones que permiten que se relacionen entre sí y ayudan a establecer de forma numérica y sistemática la forma en que realizaremos el etiquetado de acordes y regiones tonales.

A continuación presento una representación de las clases utilizadas en notación y diagramas UML-*Unified Modeling Language*, (RUMBAUGH y cols., 1998, 1991). Estos son utilizados en la programación orientada a objetos, donde se crean abstracciones de datos utilizando las estructuras llamadas *Clases*, las cuales tienen distintos métodos, funciones, variables y propiedades. Cada *Clase* se representa con una tabla dividida en tres partes verticales; en la parte superior se encuentra su nombre, en la parte media las variables o propiedades y en la parte inferior los métodos o funciones.

Nuestro elemento básico será el objeto que llamaremos "*Nota*", que consiste en un valor numérico de altura, una intensidad, un valor temporal de inicio y un valor temporal de final[1]. Nos interesa solamente la numeración de las mismas[2], que en una primara instancia sirve para representar una altura determinada, donde la nota *do* central es 60 y cada semitono representa sumar o restar una unidad:

$$Do = 0, Do\sharp = 1, \cdots Si = 11.$$

[1]Para fines de programación, las variables deben tener un *tipo*, puede ser `int` que representa un número entero, `float` un número de punto flotante, o decimal, `string` que representa texto, `boolean` puede tomar como valores *verdadero* o *falso*. Si al tipo de variable le sigue un par de corchetes cuadrados como por ejemplo `float[]`, esto significa que manejamos una lista de este tipo de variables.

[2]La numeración se apega a las convenciones utilizadas por el protocolo MIDI.

No entraremos en cuestiones acústicas como sus frecuencias y afinaciones. Supondremos una afinación de temperamento igual, para más información de temperamentos ver el trabajo de Sethares, (SETHARES, 2005).

De momento, la clase sólo requiere un método, que es calcular su duración, la cual se obtiene restando el tiempo final menos el tiempo inicial. Se puede ver el diagrama de la clase en la tabla (2.1).

Nota
+númeroDeNota: float +intensidad: float +tiempoInicio: float +tiempoFinal: float
+calcularDuracion(): float

Tabla 2.1: *Clase Nota*

Para poder analizar y construir los acordes es necesario agrupar un cierto número de notas en una secuencia a la que llamaremos "*Acorde*". No hay distinción acerca de la disposición temporal de las notas, podrían ser extraídas de un acorde de bloque, una melodía o un acorde arpegiado. Encontraremos más adelante que en términos de análisis musical armónico, algunas veces no es conveniente hacer distinción entre melodía y acorde, esto con el fin de poder extraer información estructural de arpegios y acordes quebrados. En la tabla (2.2) se muestran las variables y métodos de la clase. Principalmente toma un conjunto de notas y se encarga de encontrar su forma simplificada y equivalencia por octavas para clasificar el tipo de acorde que es. De igual manera tiene un método para comparar si es equivalente, en el sentido de estructura interválica interna, a algún otro acorde.

Para los fines de este trabajo, definiremos a una "*Tonalidad*" como una serie de notas y un conjunto de acordes que tienen asignados una función tonal con respecto a la tónica. Es importante aclarar que el concepto de tonalidad para esta implementación, se restringirá a escalas de siete notas en modo mayor o menor (armónico y natural), asimismo se presupone un temperamento igual o fijo.

Acorde
+fundamental: float +notasArmonia: float[] +notasTipo: float +tipoDeAcordeStr: string +inversiónStr: string -notasSimplificadas: float[]
+<< *constructor* >> Acorde(float[]) +intervaloDiatonico(): float +equivalente(Acorde): boolean +fundamentalString(): string +imprimirString(): string -setEstructura()

Tabla 2.2: *Clase Acorde*

Como ejemplo tomemos la tonalidad Do mayor, que consiste en el conjunto de notas y acordes:

$$\{do, re, mi, fa, sol, la, si\},$$

$$\{DoM, Rem, Mim, ..., DoM_7, \ldots\}$$

En el conjunto de acordes se incluyen todas las posibles combinaciones utilizados en una tonalidad, incluyendo dominantes con séptima, novena, acordes incompletos, etc. Todas estas se guardan en un diccionario de acordes, que sirve como lista y referencia para su clasificación. El utilizar un diccionario de este tipo puede que no sea suficiente para estilos donde hay un gran componente polifónico, como lo es el Barroco, ya que puede caerse en el error de tratar de etiquetar acordes con notas no estructurales, sino que incluye notas que se justifican por medio de cuestiones melódicas. Este sería el caso de incluir una nota de suspensión o retardo dentro de un acorde y tratar de forzar a que sea un acorde de novena o séptima, cuando en realidad no tiene esa función tonal. De igual manera es importante saber que cada periodo y compositor tienen distintos estilos armónicos y suelen utilizar los acordes de distintas maneras, por lo mismo un acorde de séptima disminuido puede tener funciones muy distintas usado por Bach o por Chopin. A pesar de esto, para un primer paso es útil poder clasificar todos los acordes de acuerdo con un diccionario, para trabajos

posteriores se puede filtrar, esto para ignorar en la selección de acordes las notas no armónicas y poder especializar el programa a distintos estilos y periodos armónicos, sin embargo esto sale del enfoque de este trabajo.

Cada uno de los acordes del diccionario de la tonalidad se asocia a una y sólo una de las notas de la misma, de acuerdo con la fundamental de cada acorde. Por ejemplo una muestra del diccionario de acordes para la tonalidad *Do* Mayor es:

$$\{CM, CM_7, CM_9 \ldots\}_I, \{Dm, Dm_7, \ldots\}_{II}, \{Em, Em_7, \ldots\}_{III} \ldots \{Bo, Bo_7, \ldots\}_{VII}$$

La lista de estos acordes se clasifica de acuerdo con su nota fundamental. En la tabla (2.3), se muestra el diagrama de la clase. La información que nos interesa es la base de la escala, es decir la fundamental, el tipo de escala (Mayor o menor), la lista de acordes pertenecientes a la escala (dada por el diccionario de acordes). Por su parte, las funciones que debe realizar consisten en encontrar el grado y el índice de una nota. Si se le introduce una lista de números (notas) existe un método que nos da el porcentaje de ellas que pertenecen a la tonalidad, esto es útil para cuando al programa no se le diga explícitamente la tonalidad de la pieza y se necesite hacer alguna inferencia, los conjuntos de notas con mayor porcentaje, serán los más probables a pertenecer a la tonalidad que se esté probando. Por último tiene una función para calcular la distancia entre otra tonalidad y la original, la cual será una de las más importantes para nuestras representaciones visuales ya que, como se verá más adelante, las tonalidades con distancia mínima son las más cercanas para modular. En matemáticas se conoce como *métrica* a la función que permite medir distancias en un espacio, esta función debe cumplir ciertas propiedades[3] para que esté bien definida, es importante distinguirla del concepto musical de métrica que se refiere a cuestiones rítmicas. Para definir una distancia entre tonalidades, asociaremos las tonalidades relativas menores/mayores como un sólo elemento, de modo que por ejemplo *Do* mayor y *La* menor natural representan el mismo elemento. Para encontrar la distancia entre un par de ellas se mide el número de notas en que difieren, por

[3]Para más información de la definición y ejemplos de métricas en geometría ver el apéndice *Antecedentes Matemáticos*.

ejemplo, la distancia entre la tonalidad *Do* mayor y *La* menor natural es 0, ya que comparten todas las notas y las asociamos al mismo punto. Por otra parte la distancia entre *Do* mayor y *Fa* mayor es 1, ya que sólo tienen de diferencia la nota *si*♭ y *si*. Esta métrica está basada en el trabajo de Walton, donde realizan un estudio de las modulaciones como operaciones en un grafo de tonalidades, (WALTON, 2010). Existen otras formas de medir distancias entre acordes y elementos musicales, algunas de ellas pueden encontrarse en el trabajo de Tymoczko, (TYMOCZKO, 2009).

Tonalidad
+baseEscala: float +tipoEscala: sring +listaArmonías: Acorde[]
+<<constructor>> Tonalidad(base, tipo) +gradoNota(float): float +distancia(Tonalidad): int +indiceNota(float): int +perteneceNota(float): string +porcentajeNotas(float[]): float

Tabla 2.3: *Clase Tonalidad*

Una vez definido el concepto de tonalidad, se puede hablar de una "*Función Tonal*". Es decir, si elegimos un acorde cualquiera, este puede tener distintas funciones tonales en cada tonalidad, la clase de programación que utilizaremos para este concepto funcionará como una especie de coordenada que llevará en registro de pares ordenados de las funciones tonales que cumple dicho acorde.

La coordenada se escribirá como un par ordenado (a,b), donde a indica el grado con respecto a la tonalidad principal y b es el grado que el acorde representa con respecto a la tonalidad a. Es importante notar que los valores de a y b corresponden a los grados romanos de la tonalidad, por lo tanto tienen un rango de 1 a 7, esto es diferente a la numeración que hacemos en el círculo cromático o la banda de moebius donde se utilizan los doce valores cromáticos. Por ejemplo, en el contexto de *Do mayor*, el acorde *Sol mayor* puede tener las coordenadas $(1,5)$ que es el quinto grado de *Do*, bien $(6,7)$ el séptimo grado de *La menor*. El concepto de *Función Tonal* es adecuado para describir las distintas regiones armónicas a las que puede pertenecer un acorde en una pieza.

Lo anterior en un sentido de música tonal donde los acordes siguen las funciones principales de Tónica, Subdominante y Dominante. El diagrama (2.4) muestra que esta clase sirve principalmente para llevar un registro de la función tonal del acorde y no contiene ninguna función de cálculos.

Función Tonal
+escala: float +grado: float
+imprimir(): string

Tabla 2.4: Clase *Función Tonal*

Por último, al conjunto de todas las tonalidades mayores y menores naturales lo llamaremos *"Universo Tonal"*, el cual es el espacio abstracto que se recorre con cada una de las piezas que se interpreten. Es nuestro espacio de posibilidades, y nos interesa encontrar las relaciones que existen entre dicho universo y cada una de las obras. Para poder explorar las regiones armónicas de una pieza musical, debemos definir una estructura geométrica que organice y relacione a las veinticuatro tonalidades mayores y menores. El primer paso será ser generar un grafo de veinticuatro vértices donde cada uno representa una tonalidad y dos de ellas están relacionadas de acuerdo con su distancia armónica. De este modo, definiremos al universo tonal como al conjunto $U = \{T_n\}_{24}$, donde cada T_i representa una de las tonalidades mencionadas, junto con la métrica entre ellas $d(T_i, T_j)$. Por lo anterior, el *Universo Tonal* contiene la lista de todas las tonalidades junto con algunas funciones. Entre ellas tenemos una que calcula el porcentaje de pertenencia de una lista de notas en cada tonalidad, a esto lo llamaremos *Vector de porcentajes*. Otro método de gran utilidad es el que calcula el grado y función tonal de una lista de notas. De este modo, la clase ofrece un sistema robusto para clasificar cualquier acorde dentro del universo de las veinticuatro tonalidades mayores y menores.

Las clases que definimos en esta sección sirven como base para la biblioteca computacional que se implementa en el tercer capítulo. Cada una de ellas representa una abstracción matemática de los conceptos musicales.

Universo tonal
+universoMayor: Tonalidad[] +universoMenor: Tonalidad[] +tonalidadActual: Tonalidad
+gradoSecundario(float[], Tonalidad) +asignarTonalidadActual() +obtenerTonalidad(base, tipo) +vectorPorcentaje(float[])

Tabla 2.5: Clase *Universo Tonal*

2.2. Pieza musical

¿Dónde se encuentra la música? Podríamos decir que en las ondas del aire, o bien en la partitura, o en una interpretación quizá. La realidad es que el proceso musical es muy complejo y no tiene una fuente ni receptor únicos. Para los fines de este trabajo, una pieza musical es un conjunto de eventos sonoros que ocurren en el tiempo. Más allá de la partitura, existe un lapso entre la emisión del primer sonido y la entrada de una segunda voz, así como el tiempo que tarda en detenerse la primera nota y entrar la siguiente. En fin, es una secuencia de información o mensajes que está ordenada cronológicamente y parte de esta información es precisamente la que nos proporciona el archivo MIDI de una determinada pieza. Este tipo de archivos tienen la característica favorable de requerir poco memoria de almacenamiento, es difícil encontrar un archivo de este tipo que requiera más de un megabyte. Para los fines armónicos, no nos interesa la información de dinámicas y articulación, sino simplemente los mensajes que contengan la información de *Inicio de nota* y *Fin de nota*. de acuerdo con lo anterior, podemos definir una pieza musical como una secuencia ordenada de eventos $(tipo, nota, tiempo)$

$$M = \{(T,n,t)_i\}_{i=0}^{N}$$

donde N es el número total de eventos; T toma el valor 1 para indicar inicio y 0 para el fin de una nota; n toma un valor entre 0 y 127 de acuerdo con las numeración de las notas MIDI, donde 60 es Do central; y t toma valor temporal que puede estar en *milisegundos* para temporización absoluta,

o en *ticks* para temporización relativa[4]. Esta secuencia de notas es sobre la cual se aplicarán las clases definidas en el *Universo Tonal*.

2.3. Identificación de conjuntos de notas

Si tenemos una pieza musical M, queremos organizar los eventos en grupos pequeños de bloques de notas que representen información tonal. Este es un proceso complejo, ya que requiere habilidades como identificación patrones armónicos, es decir, conjuntos de notas que juntas formen un acorde con una cierta función tonal. Para este trabajo, nos limitaremos a representar conjuntos de notas seleccionados por el usuario. Es decir, se deja para investigaciones futuras el subdividir el conjunto M en patrones que tengan significado armónico. El objetivo de este trabajo es proporcionar una herramienta para el usuario que le permita organizar las listas de notas que él seleccione una a una, con esta información se creará un mapa visual de las regiones y funciones tonales que conforman una pieza.

2.4. Grafo Universo Tonal

La representación gráfica de las relaciones tonales es lo más importante en este trabajo, ya que el usuario organizará de forma visual las relaciones que la mayor parte del tiempo son sólo auditivas, e incluso pueden pasar desapercibidas para un oído que no tenga el entrenamiento armónico necesario para escucharlas. Al definir una *métrica*[5] sobre los elementos del Universo Tonal, se define una relación de distancias entre cada una de las tonalidades, esto permite representar las relaciones entre ellas de forma espacial. En la figura (2.1) se muestra un ejemplo de lo anterior. Se presenta un *grafo* donde cada vértice es una tonalidad y las líneas entre ellos denotan una relación

[4]Un tick es la mínima división de tiempo que tiene un archivo MIDI y su duración está determinada por el tempo de la pieza.

[5]Métrica en el sentido de distancia matemática, no en el sentido temporal que suele utilizarse en música. Para más información ver el apéndice de antecedentes matemáticos

de distancia cercana, a su vez, la tonalidad en rojo es la que corresponde a la sección actual de la pieza musical que se estudia, mientras que la azul son algunas de las tonalidades que se visitaron en secciones pasadas. Esto se implementa de modo que el usuario pueda interactuar con los elementos y encontrar la información armónica necesaria para la creación de un mapa, en este caso un grafo, que muestre las regiones tonales visitadas en la pieza.

Si se tiene una secuencia de *Acordes* $\{A_n\}$, cada una de ellos tendrá un significado distinto en las distintas tonalidades. El trabajo del *Universo y Grafo Tonal* es distribuir la secuencia en cada una de ellas. Como se puede ver en la figura (2.1), en las tonalidades *Fa mayor* y *Do menor* se incluyen los acordes adecuados a cada una de ellas de acuerdo al análisis realizado.

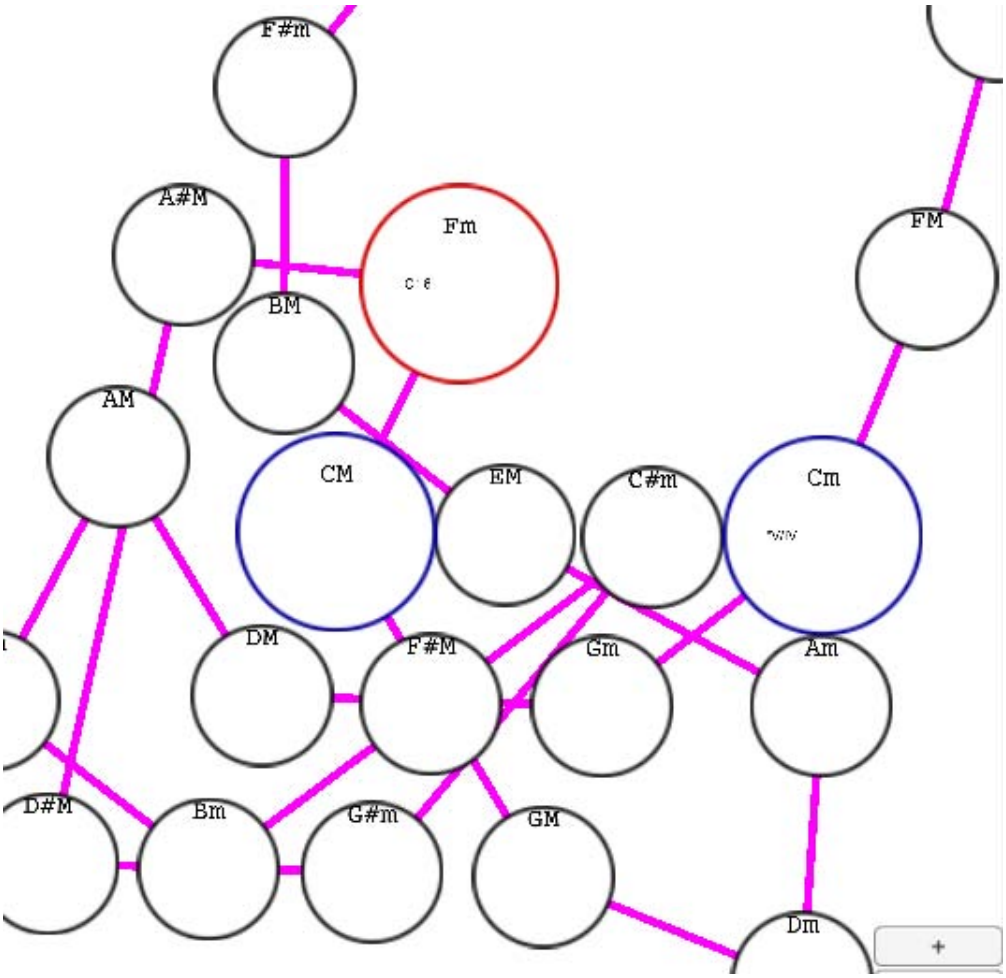


Figura 2.1: Grafo Universo Tonal

No basta con que las tonalidades se relacionen entre sí de manera gráfica, es necesario encontrar una representación que sea regular y sistemática, de modo que el usuario pueda navegarla y entenderla de manera fácil. En la figura (2.1) se puede ver que las tonalidades están dispersas, por lo que sería útil filtrala para crear un nuevo gráfico que sea más representativo. Imaginemos de nuevo que cada tonalidad es un vértice y los uniremos con un arista solamente si tienen una distancia menor o igual a uno[6]. El grafo que obtenemos es sorprendentemente un círculo de quintas, de igual manera que se obtiene en el trabajo de WALTON (2010) y se muestra en la figura (2.2).

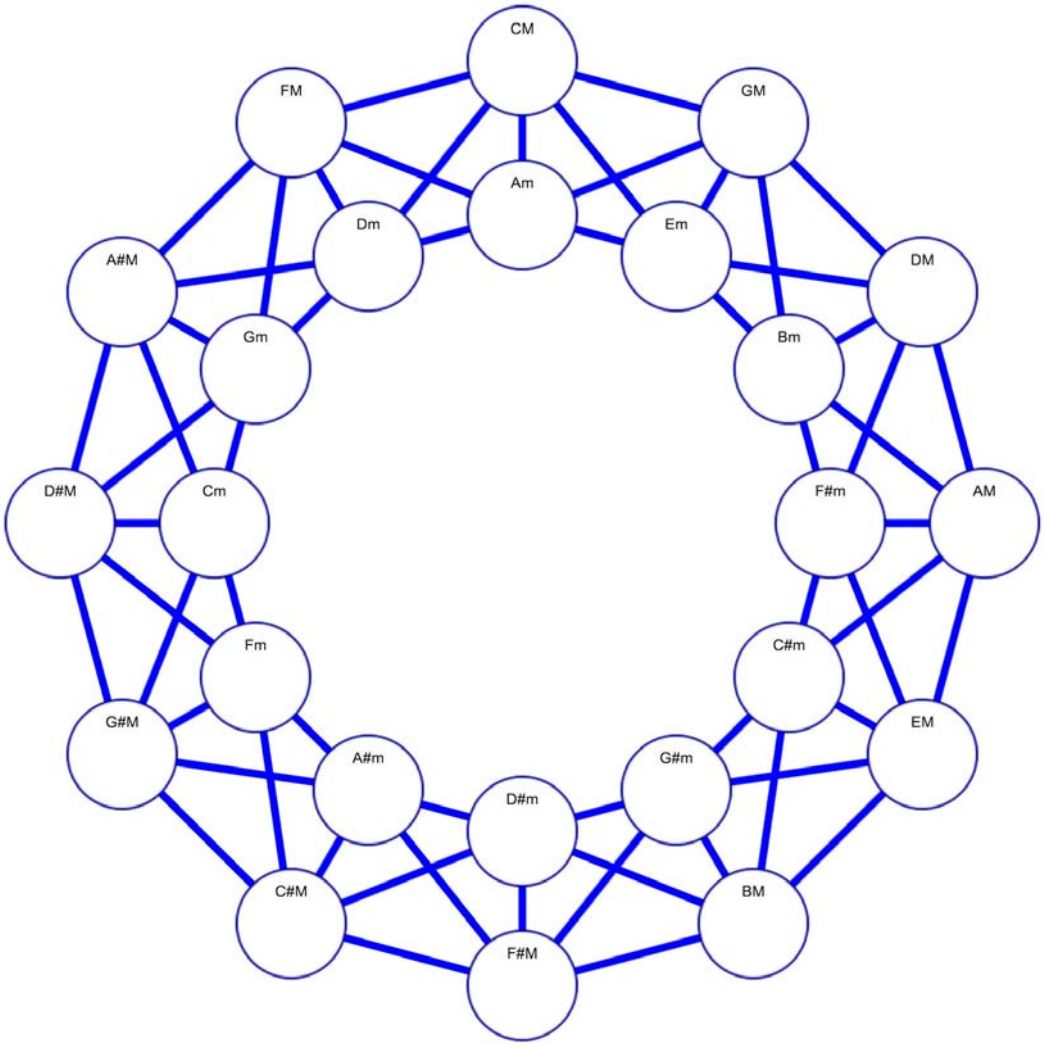


Figura 2.2: Grafo círculo de quintas.

[6]La distancia 0 de una tonalidad indica que se busca la distancia con ella misma, es decir $d(A,A) = 0$, o bien con su tonalidad relativa menor(mayor), con la que comparte la armadura.

El beneficio de esta representación es que permite ver a grandes rasgos y de manera ordenada, la estructura tonal de una obra, así como las zonas donde permanece más tiempo. Se obtiene una representación gráfica del *Plan tonal* de la pieza, la cual se implementará para llevar el registro armónico. Esta nueva gráfica es una extensión de la que muestra Walton, con la diferencia que incluye la serie de acordes que se presenta en la obra como un subgrafo en cada tonalidad y es de gran utilidad para la visualización de la estructura armónica.

Diccionarios de acordes

Existe una combinación enorme de clases de acordes posibles, sin embargo, los que se utilizan en un contexto de piezas tonales no llegan a ser más de cien combinaciones. En la tabla (2.6) se muestra una selección de clases de acordes exhaustiva, el resto surgen de transponer cada uno de ellos. En la música tonal del siglo XIX las clases de acordes más usadas es aun más pequeña y se muestran en la tabla (2.7).

Tipo	**Notas**	**Tipo**	**Notas**	**Tipo**	**Notas**	**Tipo**	**Notas**
M	(0, 4, 7)	M9	(0,2,4,7,11)	m7add13	(0,3,7,9,10)	4P	(0,5)
m	(0, 3, 7)	mM9	(0,2,3,7,11)	mM7add13	(0,3,7,9,11)	6+	(0,6)
o	(0, 3, 6)	11	(0,2,4,5,7,10)	7b5	(0,4,6,10)	5P	(0,7)
+	(0, 4, 8)	m11	(0,2,3,5,7,10)	7#5	(0,4,8,10)	6m	(0,8)
M7	(0, 4, 7, 11)	M11	(0,2,4,5,7,11)	7b9	(0,1,4,7,10)	6M	(0,9)
D7	(0, 4, 7, 10)	mM11	(0,2,3,5,7,11)	7#9	(0,3,4,7,10)	7m	(0,10)
m7	(0, 3, 7, 10)	13	(0,2,4,7,9,10)	7#5b9	(0,1,4,8,10)	7M	(0,11)
o/7	(0, 3, 6, 10)	m13	(0,2,3,7,9,10)	m7#5	(0,3,8,10)		
o7	(0, 3, 6, 9)	M13	(0,2,4,7,9,11)	m7b9	(0,1,3,7,10)		
mM7	(0, 3, 7, 11)	mM13	(0,2,3,7,9,11)	9#11	(0,2,4,6,7,11)		
D7*5	(0, 4, 10)	add9	(0,2,4,7)	9b13	(0,2,4,7,8,11)		
DM9*5	(0, 2, 4, 10)	Madd9	(0,2,3,7)	6sus4	(0,5,7,9)		
Dm9*5	(0, 1, 4, 10)	6add9	(0,2,4,7,9)	7sus4	(0,5,7,10)		
5	(0,7)	m6add9	(0,2,3,7,9)	M7sus4	(0,5,7,11)		
Sus4	(0,5,7)	D7add11	(0,4,5,7,10)	9sus4	(0,2,5,7,10)		
Sus2	(0,2,7)	M7add11	(0,4,5,7,11)	M9sus4	(0,2,5,7,11)		
6	(0,4,7,9)	m7add11	(0,3,5,7,10)	2m	(0,1)		
m6	(0,3,7,9)	mM7add11	(0,3,5,7,11)	2M	(0,2)		
9	(0,2,4,7,10)	D7add13	(0,4,7,9,11)	3m	(0,3)		
m9	(0,2,3,7,10)	M7add13	(0,4,7,9,11)	3M	(0,4)		

Tabla 2.6: Diccionario de clases de acordes, al transponer estos a distintas notas fundamentales obtenemos versiones para cada nota base, (BARBANCHO y cols., 2013)

Podría parecer que el definir un diccionario de acordes es una limitante para las combinaciones

a analizar, sin embargo es una práctica común para el análisis armónico algorítmico, como se mostró en el primer capítulo. Este diccionario es muy útil, ya que permite clasificar las miles de posibilidades armónicas en un grupo pequeño de familias.

Por su parte, los acordes que aparecen en las tonalidades mayor y menor se muestran en la tabla (2.7). Esta lista es la base de la música tonal, las combinaciones y variaciones de la misma nos llevan a construir miles de obras.

Mayor			**Menor**	
I	base + 0, M		i	base + 0, m
ii	base + 2, m		ii	base + 2, o
iii	base + 4, m		III	base + 3, M
IV	base + 5, M		iv	base + 5, m
V	base + 7, M		v	base + 7, m
vi	base + 9, m		VI	base + 8, M
vii	base + 11, o		VII	base + 10, M
I7	base + 0, M7		i7	base + 0, m7
ii7	base + 2, m7		ii7	base + 2, o/7
iii7	base + 4, m7		III7	base + 3, M7
IV7	base + 5, M7		iv7	base + 5, m7
V7	base + 7, D7		v7	base + 7, m7
vi7	base + 9, m7		VI7	base + 8, M7
vii7	base + 11, o/7		VII7	base + 10, D7
ii arm	base + 2, o		iii arm	base + 3, +
vi arm	base + 9, +		V arm	base + 7, M
vii arm	base + 11, o7		vii arm	base + 11, o
iv arm	base + 5, m		i7 arm	base + 0, I+
N	base + 1, M		III+ arm	base + 3, III+
V9	base + 7, 9		V7 arm	base + 7, D7
VM9-5	base + 7, DM9*5		vii7 arm	base + 11, o7
Vm9-5	base + 7, Dm9*5		N	base + 1, M
V 7b9	base + 7, 7b9		V9	base + 7, DM9*5
V 7b5	base + 7, 7b5		VM9-5	base + 7, Dm9*5
V 7#5	base + 7, 7#5		V7-5	base + 7, D7*5
V 7-5	base + 7, D7*5		V7b9	base + 7, 7b9

Tabla 2.7: Acordes utilizados en las tonalidades mayor y menor

El uso de los acordes y armonías depende del contexto y estilo histórico en el que una pieza fue creada. Los tipos de acordes y colecciones de notas utilizadas por Mozart, son muy distintos a los de Chopin, Stravinsky o Hindemith. Por esta razón, el análisis armónico es particularmente difícil, ya que no hay reglas que se cumplan rigurosamente en todos los casos. Sin embargo, si tomamos obras tonales del periodo clásico o romántico, encontraremos que existen patrones muy

firmes y tendencias a preferir ciertas clases de acordes sobre otras, por ejemplo, los acordes mayor y menor sobre los acordes aumentados. En este trabajo analizaremos obras que se estructuran con las dinámicas de las funciones Tónica, Subdominante y Dominante para guiar el flujo armónico, en particular analizaremos algunas piezas del periodo Clásico y Romántico que sigan esta estructura de funciones tonales claras.

Por todo lo anterior es necesario y útil contar con una lista de acordes que sirva como base de datos y referencia para clasificar las colecciones de notas obtenidas en la sección anterior.

Limitaciones del modelo matemático

Como cualquier otro modelo matemático, la abstracción de un proceso implica hacer una simplificación del problema para poder abordarlo inicialmente, para pasar después a generalizaciones que permitan encontrar una solución mejor aproximada a la realidad[7].

En el contexto de esta investigación, las simplificaciones que se decidieron hacer van de la mano con la delimitación del problema. Por una parte, la elección de la música tonal como repertorio de obras a estudiar requiere una selección más precisa, hay tantas formas de escribir la música tonal y varían entre épocas, estilos y compositores, que sería ingenuo tratar de abarcarlas a todas con un sólo modelo. Por esta razón, se eligieron como prueba para el sistema compositores y obras que sigan un plan tonal que se apegue a las funciones de *Tónica, Subdominante y Dominante* como guía del flujo armónico. A su vez, el sistema funciona para piezas donde los acordes y armonías que se utilizan son relativamente fáciles de identificar y agrupar, es decir, piezas donde

[7]Como ejemplo de esto tenemos el modelo matemático de la caída libre de un objeto. Para modelar esto en física primero se supone que el objeto cae en el vacío y se toma en cuenta sólo la masa y la velocidad inicial. Este modelo nos da la posición del objeto que cae con respecto al tiempo de una manera muy precisa, sin embargo si se realizan mediciones y experimentos, se podrá ver que hay un error, principalmente debido a que en la formula se ignora la fricción del aire. Para hacer que el modelo sea más preciso se deben incluir términos que tomen en cuenta los demás factores que afectan al objeto. De igual manera en este trabajo, comenzamos con una aproximación de cómo se comporta la música tonal, y en trabajos posteriores se incluirán cada vez más condiciones que nos permitan acercarnos más a la realidad.

los acordes se puedan visualizar por su disposición de notas simultáneas en el tiempo, o bien, en forma de arpegios que sean sencillos de identificar. Más que una limitante, al ser este un trabajo en proceso, la elección de este tipo de piezas y compositores funciona como un primer paso para abordar temas armónicos más complejos. Por otra parte, el estudio de la música atonal puede hacer uso de las técnicas desarrolladas en este trabajo de tesis si se hacen las modificaciones adecuadas. Las representaciones visuales del *Círculo Cromático*, *Banda de Moebius*, *Tonnetz* pueden ofrecer caracterización de grupos de notas por medio de figuras geométricas que aparezcan en cada uno y de esta forma ayudar a identificarlas en una partitura. El grafo tonal no aplica directamente al estudio de la música atonal, ya que por definición ya no se buscan las relaciones y funciones tonales que se utilizan en la música tonal. Sin embargo, se puede adaptar la definición de *Tonalidad* de este trabajo para que se ajuste a sistemas arbitrarios de conjuntos de notas y acordes, y no sólo a las tonalidades mayores y menores.

En cuanto a la obtención de datos que se requiere, se decidió utilizar los formatos *MIDI* y *MusicXML*, esto puede ser controversial por varias razones. En general la selección de una partitura para analizar requiere el conocimiento y la certeza que corresponden a una edición confiable y fiel a la versión original del autor. Debido a que la tecnología necesaria para la digitalización de partituras es muy reciente, (MAKEMUSIC, 2015; MIDI, 2015) [8], es natural que aun no tengan una distribución generalizada en el mundo de la música. En especial es difícil encontrar ediciones profesionales de estos archivos de obras de compositores anteriores al siglo *XX*, es mucho más fácil encontrar la digitalización de una partitura en formato de imagen, sin embargo esto no es muy práctico para fines de análisis de contenido musical[9]. Esto es debido a que la industria de las empresas dedicadas a la publicación de partituras digitales apenas están incursionando en producir partituras digitales de alta calidad. En contrapeso, podemos suponer que los compositores actuales utilizan cada vez más editores digitales de partitura como *Sibelius, Finale o MuseScore*, de modo

[8]La tecnología *MIDI* es de los años ochenta y el formato *MusicXML* de principios del siglo XXI.

[9]Existen programas de reconocimiento de imágen OCR (*Optical Character Recognition*) para extraer de una imagen el contenido musical, sin embargo el formato resultante termina siendo uno compatible con *MusicXML*.

que todas sus composiciones nacen directamente en estos formatos. Por lo anterior, defiendo la idea de que hay una tendencia por preferir cada vez más, partituras digitales en archivos *MusicXML* tanto para la creación de obras inéditas, como la adaptación de obras de compositores pasados, por esta razón la investigación realizada se enfoca en el análisis de estos archivos. Actualmente si el usuario no encuentra la versión digital de la pieza que busca, deberá él crear su propia versión. Esto podría ser una limitante de momento, sin embargo me parece que la validación y creación de partituras digitales sale del enfoque de este trabajo, nuestro objetivo es crear una herramienta que ayude a su análisis por medio de etiquetado de acordes y clasificación de acuerdo con sus regiones tonales.

Con respecto al etiquetado de acordes, el modelo actual es un trabajo en progreso, de modo que existen cuestiones más específicas y detalladas que se implementarán en versiones posteriores. Entre ellas tenemos la identificación de posibles acordes de tipo *napólitanos*, de sexta *inglésa*, *francesa* y *alemana*. El programa de momento reconoce los acordes diatónicos de las tonalidades en cada uno de sus siete grados, con séptima o novena, completos e incompletas. A mi conocimiento, es el primer programa que permite al usuario hacer este tipo de etiquetado en archivos *MIDI* y *MusicXML* y que despliegue la información de funciones tonales, junto con las visualizaciones mencionadas, todo en un sólo paquete portable entre plataformas, y creo que puede llegar a ser la base para una herramienta más versátil y poderosa.

Por último en la figura (2.3) se muestra un esquema de cómo se relacionan las *Clases* de sistema desarrollado entre sí, y de qué manera se comunican con los modelos de visualización geométricos *Círculo Cromático*, *Tonnetz*, *Banda de Moebius* y *Grafo Tonal*.

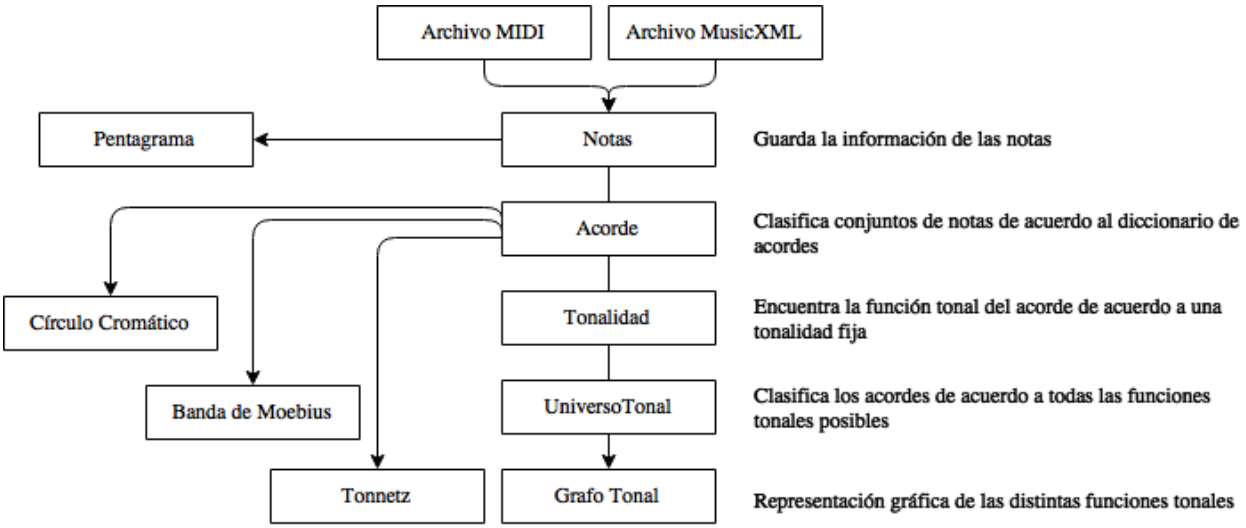


Figura 2.3: Diagrama de relaciones de los elementos en el modelo Universo Tonal.

Capítulo 3

Implementación del programa de cómputo

Para la creación del programa de cómputo se deben tomar en cuenta varios factores, entre ellos la plataforma de desarrollo, la fuente de los archivos de obras musicales y las plataformas en las que el usuario final tendrá acceso a la herramienta. Para este trabajo se creó una biblioteca que implementa las definiciones desarrolladas para el modelo *Universo Tonal*. Esta biblioteca consiste únicamente en cálculos numéricos de clasificación de acordes de acuerdo con sus conjuntos de notas, de modo que el lenguaje de desarrollo puede ser básicamente cualquier lenguaje de programación orientado a objetos[1]. Actualmente, la biblioteca de código se encuentra implementada en *Java*, *SuperCollider* y *C#*. Sin embargo, para que la implementación de la herramienta lea archivos tanto MIDI como MusicXML y proporcione una interfaz de usuario amigable, se requieren bibliotecas de código especializadas. Por esta razón se eligió trabajar en el ambiente de desarrollo UNITY (2015), ya que a pesar de estar diseñado principalmente para el desarrollo de videojuegos, es lo suficientemente versátil como para ser utilizado en proyectos multimedia como este. Por otra parte, tiene la ventaja de que se puede importar la biblioteca de código en *C#* y la aplicación resultante se puede exportar directamente a las plataformas Windows XP+, Mac OS X 10.7+, Ubuntu 10.10+,

[1]El requerimiento de que sea orientado a objetos es debido a que en este tipo de lenguajes podemos crear *clases* de programación. Cada una de ellas incluirá las propiedades y métodos de los conceptos definidos en el capítulo 2: *Nota, Universo Tonal, Función Tonal, Tonalidad*. Esto simplifica mucho la codificación ya que permite segmentar el programa en módulos independientes con tareas específicas a resolver.

SteamOS+, iOS 6+, Android 2.3.1+, entre otras; lo cual lo hace adecuado para una distribución mayor del programa de cómputo final.

Desarrollo de biblioteca de código *Universo Tonal*

Como se expuso en el capitulo anterior, la biblioteca de código *Universo Tonal* consiste en las clases: *Nota, Acorde, Tonalidad, FuncionTonal y UniversoTonal*, cada una de ella tiene en su código los datos y funciones necesarias para calcular las regiones armónicas necesarias[2] . La biblioteca debe ser lo más general posible, de modo que pueda implementarse en distintos sistemas. La entrada de datos debe ser una serie de números que representan notas, y la salida es una lista de información que es necesaria para graficar las visualizaciones independientemente de la plataforma.

```
arm = new Armonia(new float[] { 62, 65, 69, 71 });
UniversoTonal univ = new UniversoTonal();

univ.gradoAcordeRomSecEscAct(arm);
univ.vectorPorcentajes(arm);
```

Tabla 3.1: Ejemplo de uso de la librería en UNITY

En el ejemplo de código que se presenta a continuación se muestra un fragmento de cómo se representa en código *C♯* la clase que se encarga de presentar al usuario la etiqueta del acorde que seleccionó. La clase se llama *EtiquetaNota*, ya que es la etiqueta que se pondrá en el acorde con la información armónica. En el método de inicialización `start()`, se accesa a la variable UniversoTonal, que está definida en su propia clase, también se incluyen comandos de la forma en que se va a presentar el texto con la variable `TextMesh`. La lista `cuadritosId` es la encargada de llevar el registro de cuáles notas de la partitura y en que orden las eligió el usuario; `Armonia` es quién traduce la selección de notas y asigna la etiqueta del tipo de acorde. La función `update()` es la más importante en esta clase, ya que es la que se encarga de imprimir la función tonal del

[2]Las *Clases* de programación se escriben en una sola palabra y no llevan acentos ya que provoca un error en el sistema.

acorde seleccionado, esta función se actualiza varias veces por segundo para revisar si el usuario ha cambiado la tonalidad actual de ese fragmento y de este modo actualizarse con la etiqueta y región tonal adecuada.

```
public class EtiquetaNotas : MonoBehaviour
{
   // Aqui se lleva el registro de las notas seleccionadas por el usuario
    public Armonia armonia;
    public List<float> cuadritosId;
    UniversoTonal univ;
    TextMesh texto;
    public string tonalidadEtiqueta = "CM";

    void Start()
    {
      // Por default se elige la tonalidad Do Mayor, a menos que el usuario
      // la cambie durante la ejecuci'on del programa
        tonalidadEtiqueta = "CM";
        GameObject obj = GameObject.Find("DATOS_tonalidad");
        MainDatos mainDatos = obj.GetComponent<MainDatos>();
        univ = mainDatos.univ;
        texto = GetComponent<TextMesh>();
        texto.color = Color.black;
    }

    void Update()
    {
      // Se actualiza constantemente la revisi'on de etiqueta para
      // revisar si el usuario ha cambiado la selecci'on de acordes
      // o la tonalidad actual.
        texto.text = univ.gradoAcordeRomSec(armonia,
            univ.stringToEscala(tonalidadEtiqueta)) + "\n" +
            armonia.getStringSencillo();
    }
}
```

En la figura (3.1) se muestra el ambiente de desarrollo *Unity 5.0*, donde se desarrolló este trabajo de tesis. Del lado izquierdo se muestra la interfaz gráfica que se presenta al usuario, mientras que del lado derecho se encuentran las propiedades de cada uno de los objetos que integran al programa de cómputo, así como los accesos a los archivos de programación en *C♯*.

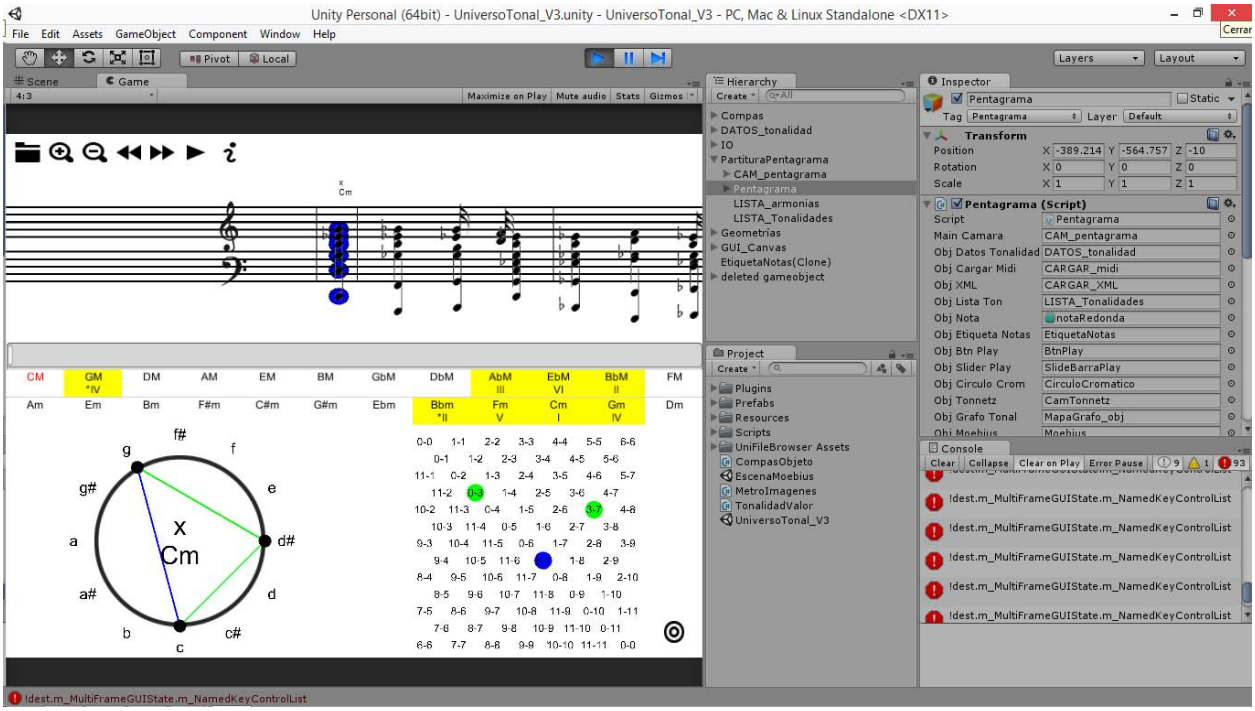


Figura 3.1: Ambiente de desarrollo y programación *Unity 5.0*

3.1. Extracción de información musical de un archivo digital

En cuestión de la fuente de los archivos, existen básicamente tres opciones: la partitura, un archivo de audio, o un archivo de representación numérica como MIDI o MusicXML. Si usamos la partitura, debemos traducir todas y cada una de las notas a sus unidades numéricas de altura y tiempo, lo cual requiere mucho trabajo, por lo tanto la partitura no es una buena fuente. En el caso del archivo de audio como *mp3* y *wav*, el problema es que es muy difícil decodificar de la señal auditiva cada una de las notas que suenan simultáneamente. El archivo MIDI tiene la característica de contener todos los eventos musicales de una pieza, tanto de altura como temporales. A su vez, existen un gran número de archivos en este formato disponibles para obtener en la red. Existe controversia acerca de la calidad de estos archivos, ya que no son regulados como las partituras o las grabaciones; sin embargo, el proceso de validación de ellos está fuera del alcance de este trabajo. Por su parte, el formato MusicXML es el más completo para representar digitalmente el contenido de una partitura, no obstante, no es muy común encontrar una gran cantidad archivos en este formato en línea. En este trabajo implementamos la decodificación de ambos formatos,

cada uno con sus ventajas y desventajas; de MIDI, la facilidad de tener un catálogo muy grande de piezas a elegir; MusicXML, tener la información completa de la partitura en formato digital, es decir: ritmos, tempos, armaduras, ligaduras, dinámicas, dirección de plicas, entre muchos otros. En el caso de que más adelante surja una aplicación que pueda extraer eficientemente información musical de la imagen digital de una partitura o de un archivo de audio, es muy probable que el resultado de esa información sea vaciado a un archivo MusicXML o MIDI, para lo cual este programa estará habilitado. Es importante tener en cuenta que estos archivos no guardan sonido de ningún tipo, lo que contienen son instrucciones para ser interpretadas por un sintetizador. Existe la confusión de que estos archivos son de mala calidad de audio, sin embargo este juicio es falso, ya que la calidad de sonido viene del sintetizador que lo ejecuta y no del archivo MIDI. A pesar de trabajar de momento sobre MIDI, los métodos desarrollados para el *Universo Tonal* podrían aplicarse a implementaciones futuras sobre archivos de audio o partituras.

Quiero hacer una reflexión sobre qué tipos de partituras se pueden encontrar en ambos archivos. Por un lado existen páginas como *Kunst Der Fugue*, (ONCLASSICAL, 2015), donde podemos encontrar la mayoría de las obras emblemáticas de los compositores de los periodos Barroco, Clásico y Romántico. Tienen en su catálogo todas las sonatas y sinfonías de Beethoven y de Mozart, la gran parte de la obra de J.S. Bach, así como de Chopin, y una lista muy grande de compositores la cual se muestra sólo como referencia: Albeniz, Albinoni, Albrechtsberger, Alkan, Anglebert, Bach, J.C., Balbastre, Bartók, Becker, Berlioz, Bizet, Boëly, Brade, Brahms, Bruckner, Bühler, Busoni, Buxtehude, Byrd, Clementi, Couperin, Dandrieu, Danzi, Debussy, Dowland, Dupré, Dvořák, Eccard, Fauré, Frescobaldi, Franck, Froberger, Führer, Gabrieli, A., Gabrieli, G., Galilei, Gershwin, Gesualdo da Venosa, Glazunov, Glinka, Gomółka, Gorczycki, Gottschalk, Grieg, Guerrero, Haydn, F.J., Hammerschmidt, Handel, Hassler, Haydn, F., Henry VIII,Hindemith, Holborne, Hummel, Janáček, Johnson, J.P., Joplin, Karg Elert, Lamb, Lasso (Ordandodi), Liszt, Machault, Mahler, Maier, Marcello, B., Marini, Medtner, Mendelssohn-Bartholdy, Merula, Messiaen, Monteverdi, Morales, Morley, Morton, Mussorgsky, Pacchioni, Pachelbel, J., Palestrina, Peeters, Poulenc, Praetorius, Pres (Despres, Prokofiev, Rachmaninov, Ræhs, Rameau, Ravel, Ravenscroft, Reger,

Rimskij-Korsakov, Roman, Rosenmüller, Rossini, Rubinstein, Saint-Saëns, Satie, Scarlatti, D., Schein, Schönberg, Schubert, Schütz, Schumann, Scott, Scriabin, Shostakovich, Sibelius, Soler, Sor, Sousa,Strauss, J. jr., Stravinsky, Tallis, Tchaikovsky, Telemann, Tudor (Henry VIII), Vecchi, Verdi, Vierne, Visée, Vivaldi, Wagner, Walther, Widor, Zipoli.

Cada una de las obras en este sitio son preparadas por distintos interpretes, de modo que se pueden encontrar distintas grabaciones MIDI de la misma pieza y así comparar cuál es la más adecuada para elegir. Esta lista puede parecer exhaustiva y quizá innecesaria de mostrar en este trabajo, sin embargo, decidí hacerlo para dejar un poco más claro que el MIDI se ha vuelto un formato de archivo utilizado no sólo para la música popular, sino que hay una comunidad de músicos que lo utilizan para distintos géneros.

Por otra parte, los archivos MusicXML cuentan con un catálogo mucho menor en cuanto a variedad de piezas, esto debido en parte a que es mucho más reciente, la Versión 1.0 fue publicada en enero de 2004, mientras que la actualización más reciente (3.0) se publicó en agosto de 2011. A pesar de su corto tiempo de vida, a comparación del MIDI que data de agosto de 1983, el formato MusicXML ha encontrado un lugar en la composición y digitalización de la música ya que es un formato el cual trabajan sin problemas los editores de música profesionales como Finale, Sibelius y MuseScore. Por otra parte existen proyectos que buscan digitalizar de forma profesional cada vez más música en este formato y publicarla con licencias libres, como es el caso de *The open well-tempered clavier*, el cual consiste en digitalizar el libro primero del Clave Bien Temperado, de J.S. Bach. Por lo anterior, creo que el futuro de la digitalización de partituras hará cada vez más uso de este formato. De momento uno de sus catálogos más grandes se encuentra en *MuseScore*, (MUSESCORE, 1996), no muestra un catálogo con el nombre de los autores, pero se pueden encontrar diversas obras de compositores emblemáticos como J.S. Bach, Mozart, Beethoven y Chopin, entre otros. Sin embargo, aunque la lista es pequeña, el formato ofrece la posibilidad de que el músico prepare sus propios archivos y versiones en su editor musical favorito.

MIDI

El paso siguiente es encontrar formas de extraer el contenido armónico de una pieza de los miles de números en el archivo. Un archivo MIDI es una lista de mensajes para un sintetizador ordenados en el tiempo, lo que nos interesa es extraer dos tipos específicos de mensajes que se denominan *Note On* y *Note Off*, los cuales sirven para indicar el inicio de una nota y su fin, respectivamente. Como parámetros contienen: el valor numérico de la nota, la intensidad, y el tiempo en el cual se debe ejecutar la acción. Resumiendo, los mensajes que nos interesan contienen la información:

$$(tipo, nota, intensidad, tiempo),$$

con ellos hacemos una lista que contenga toda las notas de la pieza, y sobre ella realizaremos los cálculos para la clasificación armónica y tonal, esto corresponde a la definición de *Pieza musical* como una serie de eventos, que se presentó en el capítulo anterior. Las unidades de tiempo en MIDI se miden en *ticks* y son relativas al tempo de la pieza. Cabe notar que en este formato de archivo, no se incluye ninguna referencia a información armónica, ni de cómo deben agruparse los conjuntos de notas en el tiempo para poder clasificarse en funciones tonales, por esta razón la partitura que generemos con archivos MIDI será una mezcla entre la notación *pianola* y el pentagrama tradicional, ya que sólo indicaremos cuando una nota empieza con un símbolo de duración redonda. Esto puede ser un poco contra intuitivo para el músico, sin embargo realizar una cuantización automática como la hacen los editores de partitura sale del enfoque de este trabajo, por lo que optamos por realizar un pentagrama simplificado para esta primera versión del programa.

Note On: 69.0	vel: 0	tick: 1024
Note Off: 71.0	vel: 96	tick: 2024
Note On: 64.0	vel: 0	tick: 3024
Note Off: 64.0	vel: 96	tick: 4024
Note On: 60.0	vel: 0	tick: 5024
Note Off: 59.0	vel: 96	tick: 6024
Note On: 57.0	vel: 0	tick: 7024
Note Off: 56.0	vel: 96	tick: 8024
Note On: 57.0	vel: 0	tick: 9024
Note Off: 56.0	vel: 96	tick: 10024

Tabla 3.2: Eventos Midi

MusicXML

En el caso de los archivos MusicXML, podemos encontrarlos con dos extensiones: ".xml" y ".mxl", ambas contienen la misma información, solo que la segunda es una versión comprimida de la primera, para este trabajo utilizaremos exclusivamente ".xml", de modo que si se quiere abrir una tipo ".mxl", esta se debe extraer con un programa de descompresión[3]. A diferencia del formato MIDI, el MusicXML consiste en una serie de etiquetas que indican el tipo de información musical a expresar. Por ejemplo para indicar que la octava en la que se encuentra una nota es 5, se escribe`<octave>5</octave>`, donde la etiqueta se abre con la notación `<nombre>` y se cierra con `</nombre>` y en el medio se coloca el valor de esa etiqueta. De igual manera hay muchas etiquetas, cada una para cierta información musical diferente, entre ellas tenemos: el compás `<measure>`; la nota que agrupa toda la información acerca de un evento musical `<note>`, el tipo de nota `<step>`, es decir a que familia de notas pertenece, *do, re, mi,*; modificadores de la nota como sostenidos y bemoles `<accidental>`; duración de la nota `<duration>`; dirección de la plica `<stem>`; armadura `<key>`; división de metro `<beat>` y `<beat-type>`, entre muchas otras. A nosotros nos interesa encontrar todos los mensajes de tipo *note*, de ellos necesitamos extraer el valor de *step, octave, accidental* y *duration*, con ellos generaremos una lista equivalente a la que utilizamos cuando extrajimos la información del archivo MIDI. Con el MusicXML podemos utilizar toda la información que nos da para realizar una partitura completa incluyendo las duraciones, los compases y la dirección de las plicas, esto es muy útil para un músico ya que si se le presenta la notación musical a la que está acostumbrado, tendrá una curva de aprendizaje menos pesada.

[3] A la fecha de la escritura de este texto se recomienda el software "7-zip".

```
<measure number="1" width="339.82">
  < attributes >
    <divisions>2</divisions>
    <key>
      < fifths >0</fifths>
      </key>
    <time>
      <beats>4</beats>
      <beat-type>4</beat-type>
      </time>
    <staves>2</staves>
    <clef number="1">
      <sign>G</sign>
      <line>2</line>
      </clef>
    <clef number="2">
      <sign>F</sign>
      <line>4</line>
      </clef>
    </ attributes >
  <note default-x="79.00" default-y="-50.00">
    <pitch>
      <step>C</step>
      <octave>4</octave>
      </pitch>
    <duration>2</duration>
    <voice>1</voice>
    <type>quarter</type>
    <stem>up</stem>
    <staff>1</staff>
    </note>
  <barline location="right">
    <bar-style>light-heavy</bar-style>
    </barline>
  </measure>
```

Tabla 3.3: Ejemplo de archivo en formato MusicXML

Selección de grupos de notas

Una vez que contamos con la lista de eventos musicales completos, sin importar si se obtuvieron de MIDI o MusicXML, debemos mostrarlos al usuario para que pueda elegir los conjuntos de notas que se quieren analizar. Para lograr la selección se le presentará al usuario una interfaz gráfica estilo partitura digital en la cual por medio de *clics* y movimiento de *mouse* podrá seleccionar los conjuntos de notas que desee analizar, esto servirá para que tenga la posibilidad de agrupar notas que no solo aparezcan de forma simultánea, sino poder agrupar fragmentos melódicos o de notas arpegiadas y de esta manera buscar si delinean una función armónica en específico. Cada

vez que el usuario seleccione un grupo de notas, el programa le asignará su etiqueta armónica correspondiente de acuerdo al tipo de acorde, sus funciones y regiones tonales a las que pertenezca. Asimismo, se mostrará el acorde en las distintas representaciones geométricas que se trabajaron en las secciones pasadas.

3.2. Diseño de interfaz gráfica

Para que el programa de cómputo sea útil, es de suma importancia tener una interfaz de usuario que sea eficiente y amigable para alguien no especializado en temas computacionales ni matemáticos. Debido a que la aplicación resultante debe ser útil tanto en dispositivos móviles, como en computadoras de escritorio, se buscó un diseño que sea amigable para ambas disposiciones[4]. En las figuras (3.2) y (3.3) se muestran capturas de pantalla del programa desarrollado. La distribución de los elementos es básicamente en cuatro partes. En la parte superior, se muestra el pentagrama con las notas extraídas del archivo digital MIDI o MusicXML. En el primer caso, debido a que en estos archivos no existe información acerca de compases y la cuantización de las duraciones no es siempre exacta, se optó por identificar a las notas sin notación rítmica, sólo se introduce un símbolo de nota redonda cada vez que aparece una nueva. Para el segundo caso se incluye la notación rítmica y de compases. En el lado derecho se encuentra la representación matemática del contenido armónico, se puede elegir cuál visualización se desea por medio del botón de cambio de geometría. Las opciones son *Círculo Cromático*, el *Tonnetz*, la *Banda de Moebius* y el *Grafo Tonal*. En la parte del medio se encuentra una tabla con las veinticuatro tonalidades mayores y menores, las cuales se activarán por parte del usuario a lo largo de la partitura. Por último, en la parte superior izquierda se encuentra el menú y las opciones del programa. De izquierda a derecha tenemos botones para: abrir archivo, alejar partitura, acercar partitura, disminuir velocidad de reproducción,

[4]Para la creación de este programa de cómputo se siguieron los lineamientos y sugerencias del diseño de interfaces de usuario, en general con tendencia a mostrar en la pantalla sólo la información más importante, mostrar la información principal del lado superior izquierdo, guiar al usuario acerca de cómo debe navegar la información, entre otras cosas. Para más información de los lineamientos utilizados, ver (HOOBER y BERKMAN, 2011).

aumentar velocidad, iniciar reproducción del archivo, y por último, información.

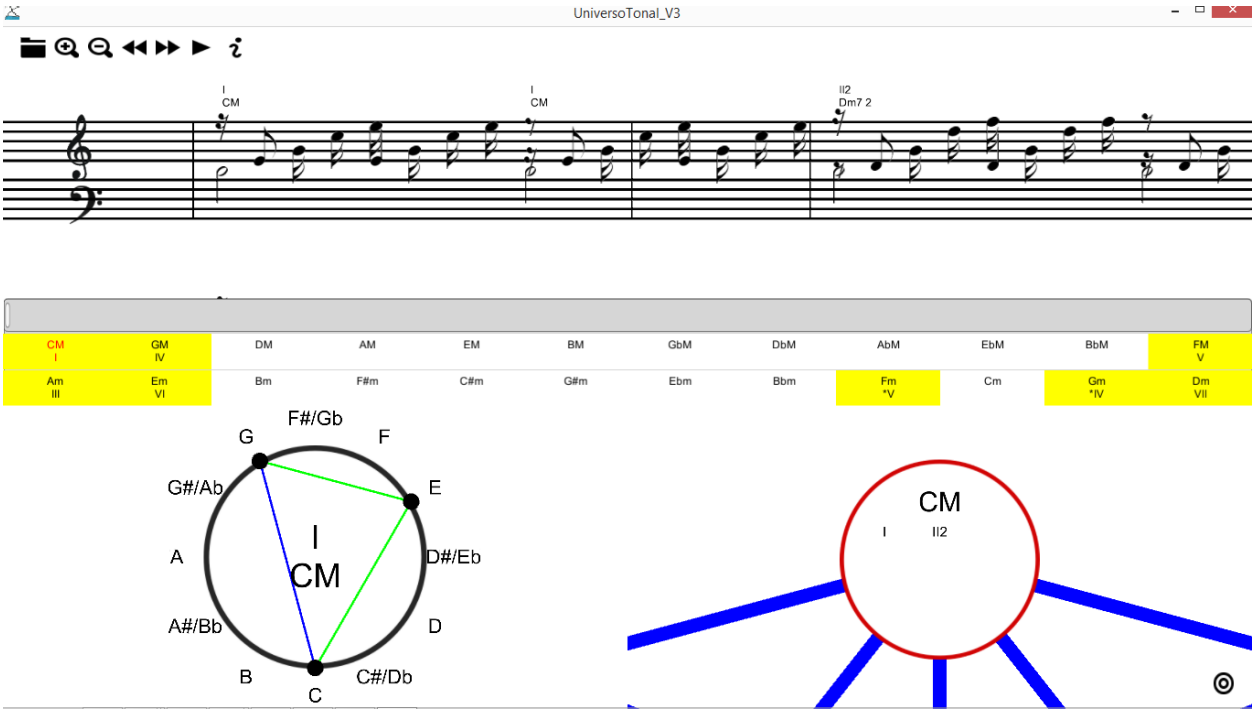


Figura 3.2: Interfaz de usuario, representación: Círculo Cromático y Grafo Tonal.

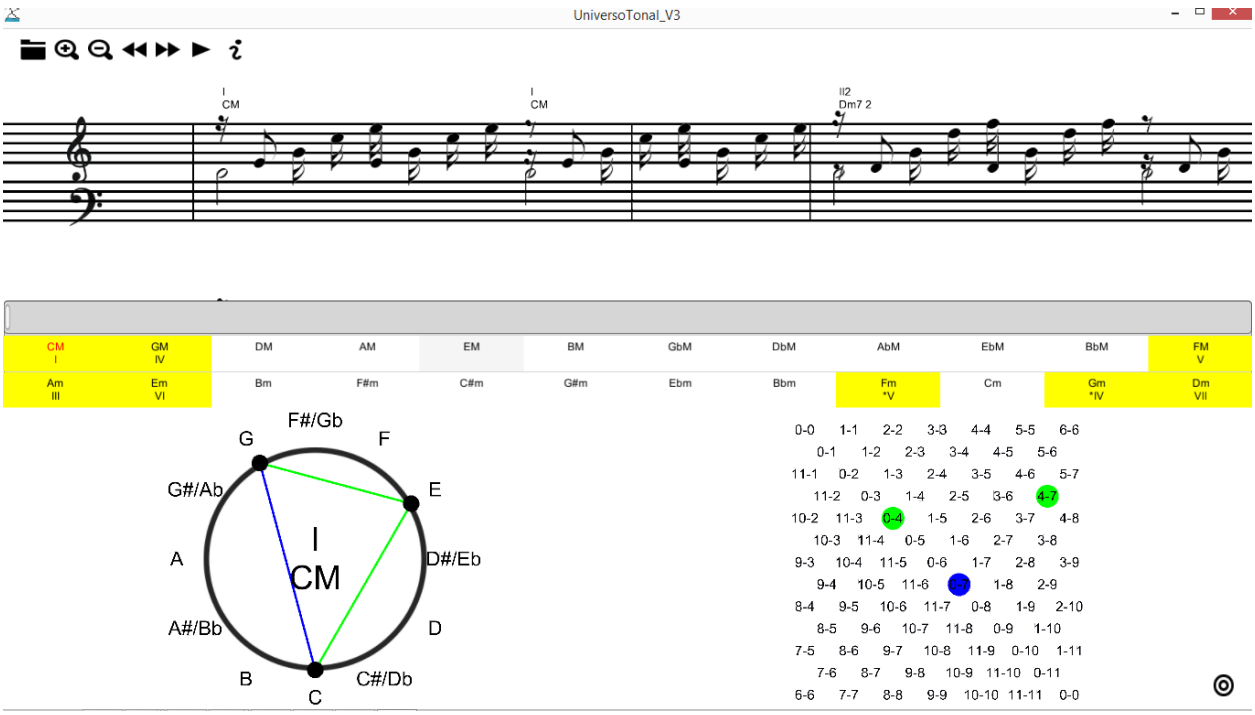


Figura 3.3: Interfaz de usuario, representación: Círculo Cromático y Banda de Moebius.

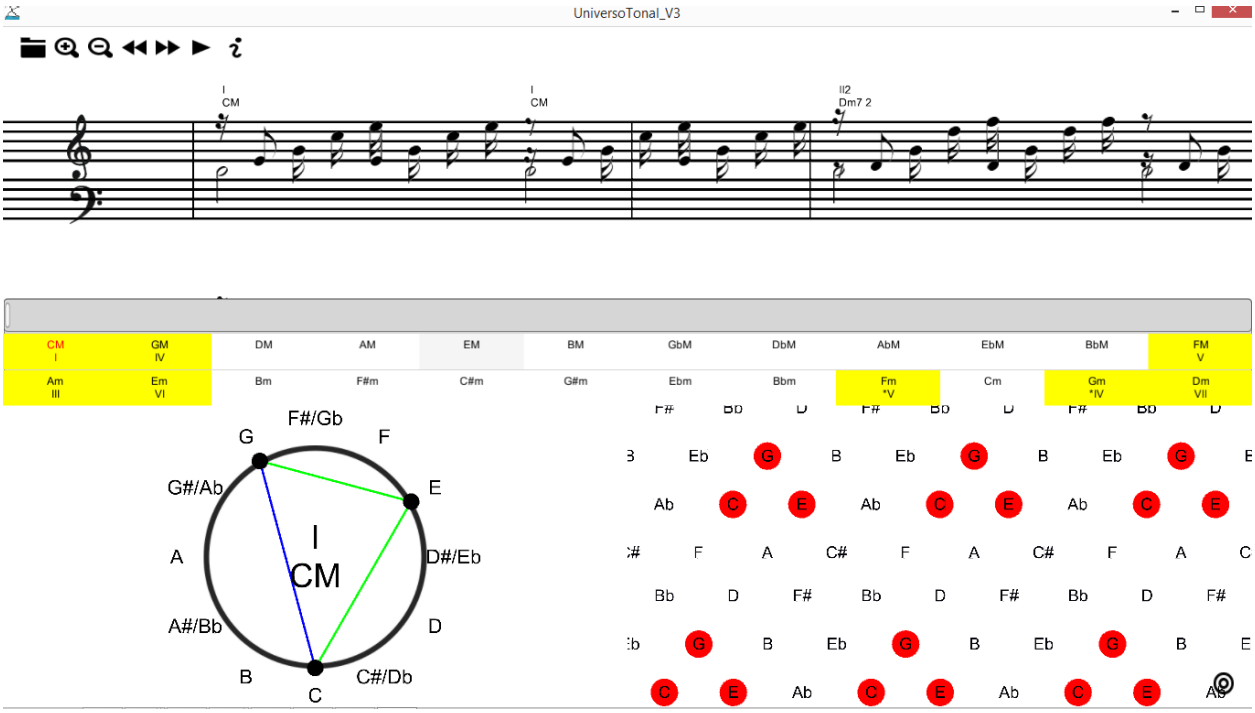


Figura 3.4: Interfaz de usuario, representación: Tonnetz.

3.3. Instrucciones de uso

La principal función del programa es ayudar al usuario a explorar las regiones armónicas de una pieza, por esta razón el trabajo del músico es escoger distintos grupos de acordes y por medio de las representaciones geométricas, encontrar las funciones tonales de las mismas. A continuación se muestra un flujo de uso del programa:

- Se abre el programa.
- Se abre un archivo MIDI o MusicXML proporcionado por el usuario, y elige por default la tonalidad *Do Mayor* para iniciar el análisis, el usuario puede cambiarla más adelante.
- El usuario selecciona uno o varios grupos de notas presionando y arrastrando sin soltar el mouse hasta finalizar la selección.
- En la tabla de tonalidades en la parte del medio se muestran los posibles significados de dichos grupos en cada una de las tonalidades.
- El usuario selecciona la tonalidad de ese fragmento y la activa en la posición adecuada de la partitura.

- Al repetir este proceso, el *Grafo Tonal*, *Círculo Cromático*, *Tonnetz* y la *Banda de Moebius*, se actualizan con los datos correspondientes que permiten visualizar al usuario las regiones tonales por donde transita. Para navegar la partitura, se mueve con las flechas del teclado.
- Cuando el usuario termina de seleccionar todas las colecciones de notas/acordes que desee, puede dar *play* en el icono de triángulo para que la pieza se reproduzca mostrando las animaciones correspondientes en cada geometría.
- Cerrar el programa y finalizar.

Controles

- Los botones + y - en el programa hacen Zoom In/Out en el diagrama de plan tonal, así como en el espaciamiento de la partitura.
- El botón con círculos concentricos en la parte inferior derecha cambia la representación visual de los acordes.
- Las flechas izquierda y derecha navegan la partitura.
- *Drag* con *clic* izquierdo selecciona las notas del acorde.
- Para borrar una etiqueta de acorde seleccionado dar *clic* derecho en ella.
- Para seleccionar una tonalidad, dar *clic* en ella en la parte superior del programa, después dar otro *clic* en la parte de la partitura donde quiere que se cambie tonalidad.
- Para borrar una tonalidad elegida, dar *clic* derecho en ella.

En el próximo capítulo se mostrarán ejemplos de uso en piezas de distintos compositores, para evaluar la eficiencia y utilidad del programa.

Distribución y licencia

El resultado de este trabajo, junto con su código fuente, se encuentra disponible en el anexo digital de la tesis. Posteriormente que se haga el registro en las instancias correspondientes de propiedad intelectual en México, se hará disponible a la comunidad en general bajo una licencia de software libre para que el proyecto siga creciendo de forma orgánica, esto se podrá acceder en la página `http://www.holomorfo.com`.

Capítulo 4

Pruebas de análisis

El programa desarrollado debe servir para analizar fácilmente la identificación de armonías y regiones tonales de una pieza. A continuación mostramos algunos ejemplos de la aplicación del mismo en fragmentos de obras de distintos compositores.

4.1. Fuentes de archivos MIDI y MusicXML

Como se mencionó en los capítulos anteriores, para este trabajo se eligió el formato MIDI ya que actualmente en la Internet existen muchos archivos con este formato. Entre las más importanes que se pueden encontrar está IMSLP, Kunst der Fugue, Midi World, Classical Archives, Piano Midi, entre muchas otras, (PETRUCCI, 2015; ONCLASSICAL, 2015; MIDIWORLD, 2015; ARCHIVES, 1994; KRUEGER, 1996). Debido a que la mayoría no son sitios académicos, la autenticidad de los archivos con respecto a la partitura no está verificada, sin embargo el usuario puede verificar mediante una escucha con partitura si el archivo corresponde aceptablemente a la pieza original y decidir si utilizar o no ese archivo.

Es mejor trabajar con archivos tipo MusicXML, ya que con ellos se tiene toda la información musical de la partitura, sin embargo, a diferencia de MIDI, hay muy pocos archivos en línea. Como este trabajo pretende hacer uso de los datos que un usuario promedio tiene al alcance por medio del internet, sacrificamos temporalmente la fidelidad a la partitura, por variedad de repertorio posi-

ble. Los archivos digitales utilizados a continuación fueron obtenidos del catálogo de MuseScore, (MUSESCORE, 1996). Para comenzar a probar el sistema se eligieron obras que principalmente consisten en acordes dispuestos en forma vertical que cambian con cada pulso, o que aparezcan de forma arpegiada.

4.2. Preludio No.1 en Do Mayor, BWV 846, Libro 1, J.S. Bach.

El Preludio 1, en *Do mayor*, BWV 846, *El clave bien temperado*, Libro 1 de Johann Sebastian Bach, (MUGELLINI, 1964), funciona como un ejemplo de análisis inicial para el programa de cómputo, ya que la progresión armónica es definida claramente como una serie de arpegios y en general se puede identificar de manera directa donde termina una armonía. La pieza tiene una modulación a *Sol Mayor* antes de regresar a la tónica. Al realizar un análisis armónico de forma manual en la partitura vemos que la armonía cambia cada compás, los primeros cuatro nos dan una progresión:

$$I \rightarrow ii_2 \rightarrow V_5^6 \rightarrow I$$

En el programa se obtiene un resultado directo equivalente, como se puede ver en la figura (4.2). Debido a cuestiones de espacio, no se puede mostrar todo el análisis del programa, sin embargo al utilizarlo, es muy directo encontrar estas armonías. En otras piezas, se verán ejemplos de como se puede analizar obras con un contenido melódico más complejo y notas no armónicas.

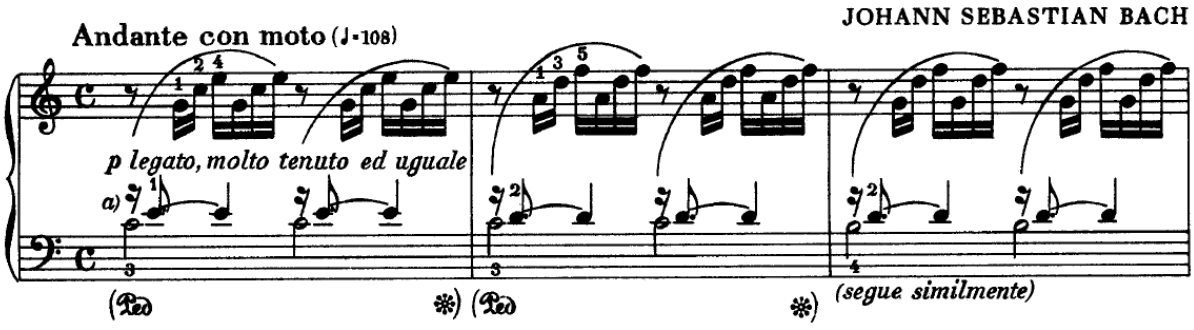


Figura 4.1: Preludio No.1 en Do Mayor, BWV 846, Libro 1, J.S. Bach. Compases 1-3, (MUGELLINI, 1964).

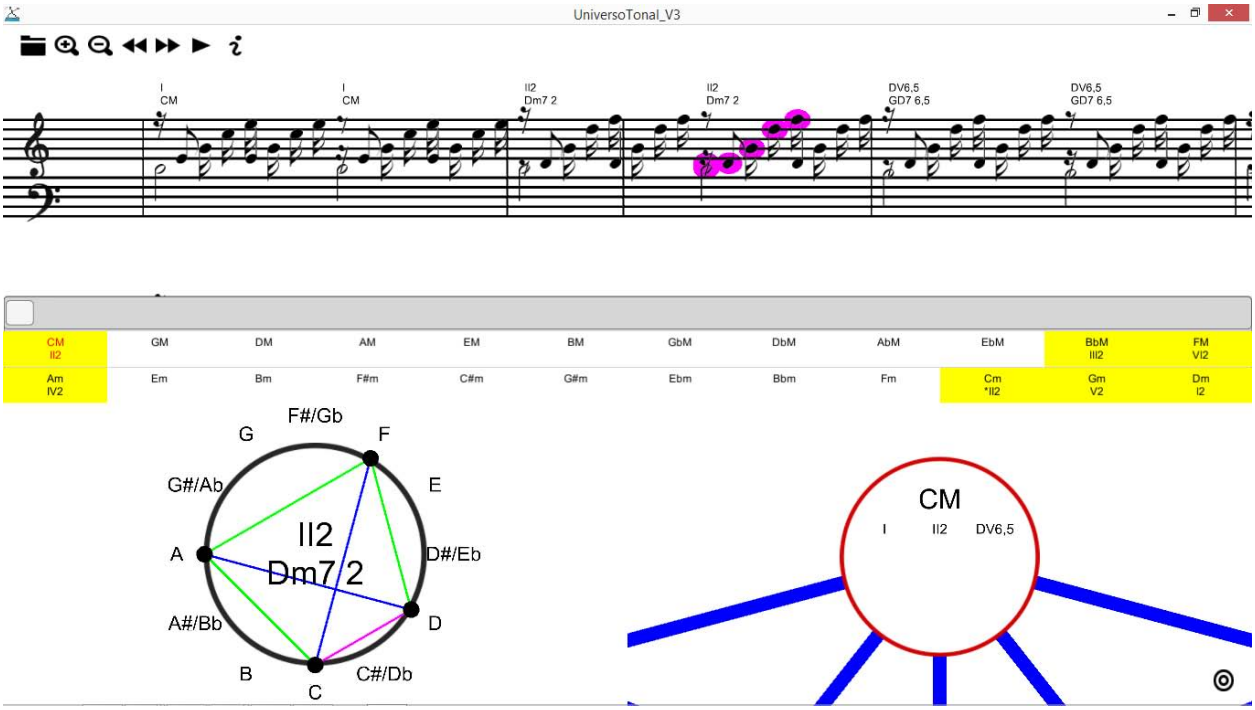


Figura 4.2: Preludio No.1 en Do Mayor, BWV 846, Libro 1, J.S. Bach. Compases 1-4. Análisis de regiones armónicas en el *Grafo Tonal* y el *Círculo Cromático*. En este caso todos los acordes pertenecen a la tonalidad *Do* mayor.

4.3. Sonata No. 16 K.545 W.A. Mozart

En el caso de la sonata de Mozart, (SIGMUND LEBERT, 1893), tenemos el ejemplo de una pieza que, de manera similar a la anterior tiene arpegiada la armonía, sin embargo, dado que cuenta con una melodía clara y definida, sirve como ejemplo de complejidad progresiva para el programa. Esta pieza se presta muy bien para la selección de armonías en el bajo, el problema aquí del usuario es aprender a distinguir cuales son las notas no estructurales de la pieza, de modo que pueda identificar una selección adecuada de progresiones. En la figura (4.4) se muestra el inicio de la pieza, el cual se puede comparar con la partitura de la figura (4.3). En este caso el programa empieza automáticamente en la tonalidad de *Do Mayor* así que para la primer parte no se requiere cambio. Es hasta más adelante, cuando comienzan a aparecer acordes que no son naturales en *Do Mayor*, pero si en *Sol Mayor*, su dominante. Esto se puede apreciar en la parte superior donde se indica la función tonal que cumple el acorde actual, en cada una de las veinticuatro tonalidades mayores y menores.

La progresión de acordes en esta pieza es similar a la anterior, bien definida principalmente por el arpegio en el bajo:

$$I \rightarrow V_7 \rightarrow I \rightarrow IV_4^6 \rightarrow I$$

Mientras se van agregando más acordes a la lista, obtenemos el siguiente diagrama de plan tonal de la figura (4.4) en el cual se ve claramente una modulación entre *Do Mayor* y *Sol Mayor*, lo cual es correcto en la exposición de esta forma sonata.

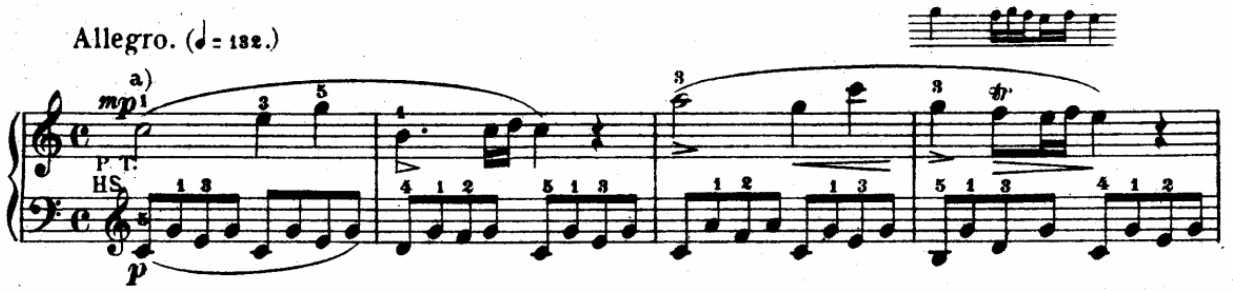


Figura 4.3: Sonata No. 16 K.545 W.A. Mozart. Compases 1-4, (SIGMUND LEBERT, 1893).

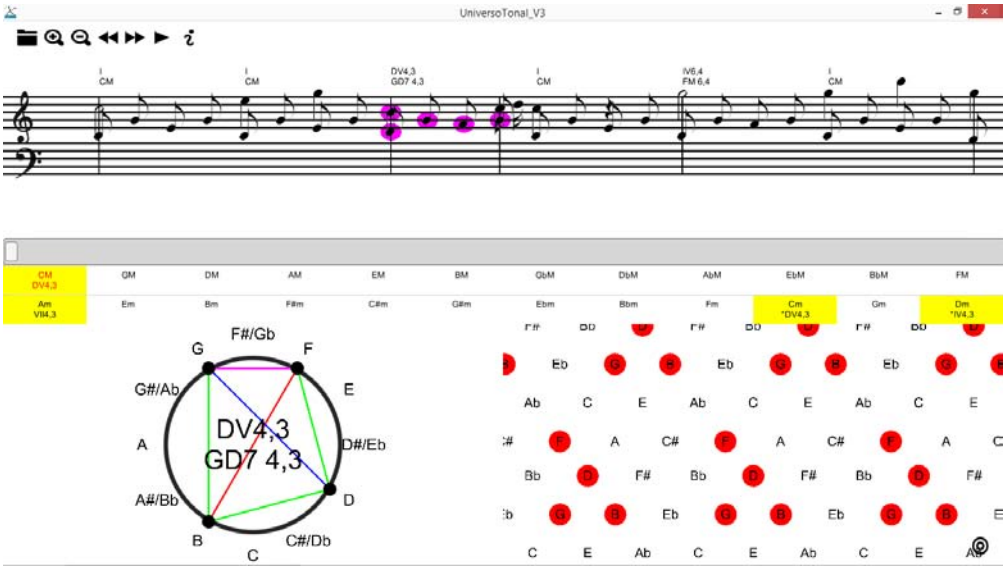


Figura 4.4: Sonata No. 16 K.545 W.A. Mozart. Compases 1-3, muestra de la visualización de las regiones tonales en el *Tonnetz* y el *Círculo Cromático*.

4.4. Preludio Op. 28, No. 20 F. Chopin

En el preludio No. 20, Op. 28 de F. Chopin, (KULLAK, 1882), tenemos una pieza con movimiento armónico que fluye principalmente en bloques verticales. A diferencia de los dos anteriores, la tonalidad del preludio no es la que se carga por default (*Do Mayor*), así que el usuario debe seleccionar la adecuada. En este caso el proceso consiste en seleccionar los primeros acordes de la pieza y analizar en que tonalidades comunes pertenecen, para así poder elegir la adecuada. De acuerdo al análisis realizado por Gilbert, (GILBERT, 2015), se obtiene:

$$I \rightarrow IV_7 \rightarrow V \rightarrow I \rightarrow IV \rightarrow N_\flat \rightarrow DV_7/VI \rightarrow VI$$

Al seleccionar los acordes en el programa de cómputo es claro que los primeros acordes pertenecen a la tonalidad *Do menor*, sin embargo, el usuario debe experimentar con varias combinaciones de notas para evitar los retardos y las nota no armónicas, como en el tercer acorde. Esto se puede ver en la figura (4.6). El círculo cromático y la tabla de pertenencias cumplen una función importante, si el usuario hubiera seleccionado el tercer acorde de forma vertical, tendría un acorde que no es identificado en ninguna tonalidad, de modo que el usuario puede corregir su selección

e inferir que las dos notas superiores consisten en un retardo. Una vez establecida, se pueden ir seleccionando grupos de notas que formen los acordes para después pasar a la visualización del diagrama de tonalidades y su plan tonal, figura (4.8).

Es curioso notar que en el sexto acorde de la figura (4.6), aparece un *Do# mayor* el cual no se identifica como perteneciente a *Do menor*, sin embargo en este caso podría cumplir la función de un acorde Napolitano. En esta versión del programa no se implementó el reconocimiento de acordes napolitanos, sextas alemanas, italianas ni francesas. Esto debido a que se buscaba como un primer paso, la identificación de las funciones tonales más claras y directas, sin embargo se planea adaptar para versiones posteriores.

En la figura (4.8) se muestra la visualización de los acordes seleccionados en el *Grafo Tonal*. Cada nuevo acorde se agrega a los círculos donde tiene una función tonal reconocida. De esta manera se hace una distribución de las regiones tonales más visitadas y se da al usuario una sugerencia de las tonalidades más fuertes que se utilizan. El círculo de la tonalidad se indica con el color rojo cuando más del cincuenta por ciento de los acordes utilizados tienen una función tonal clara en él. En este caso, podemos ver que hay una fuerte tendencia a *Do menor*, con unas desviaciones a la subdominante *Fa menor*.

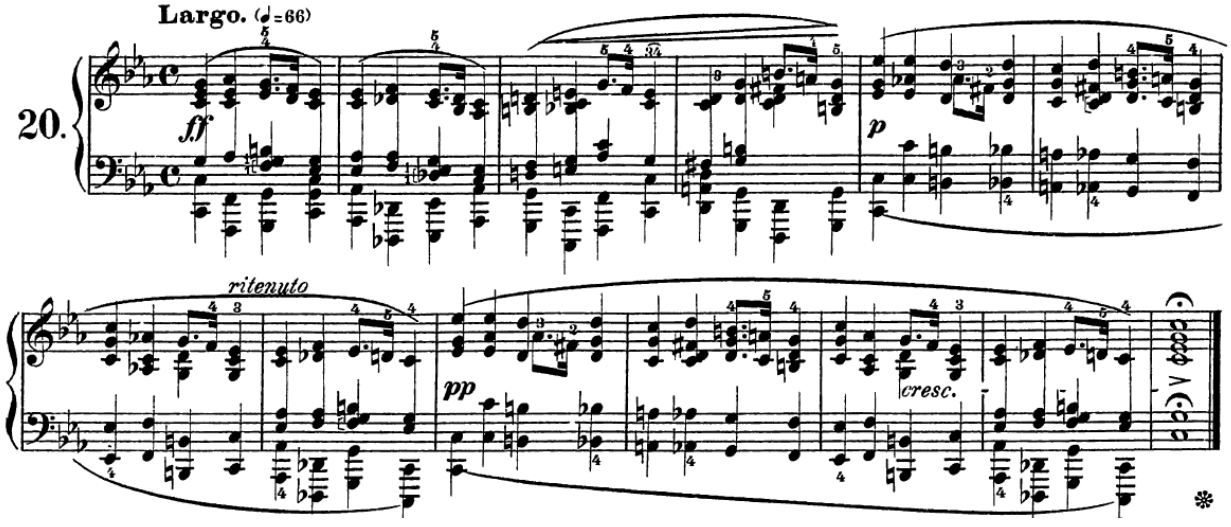


Figura 4.5: Preludio Op.28, No. 20 F. Chopin, (KULLAK, 1882).

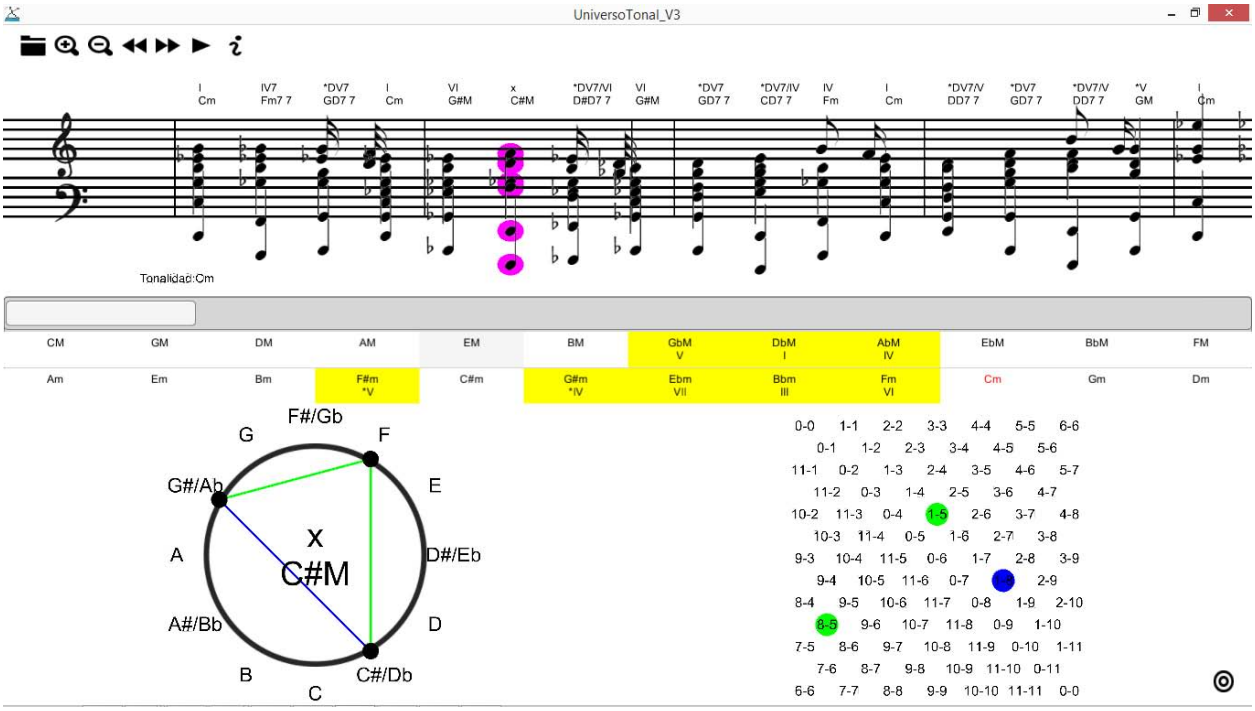


Figura 4.6: Preludio Op.28, No. 20 F. Chopin. Compases 1-2. Ejemplo de visualización de acordes en el círculo cromático. Se muestra la problemática que surge con el acorde napolitano.

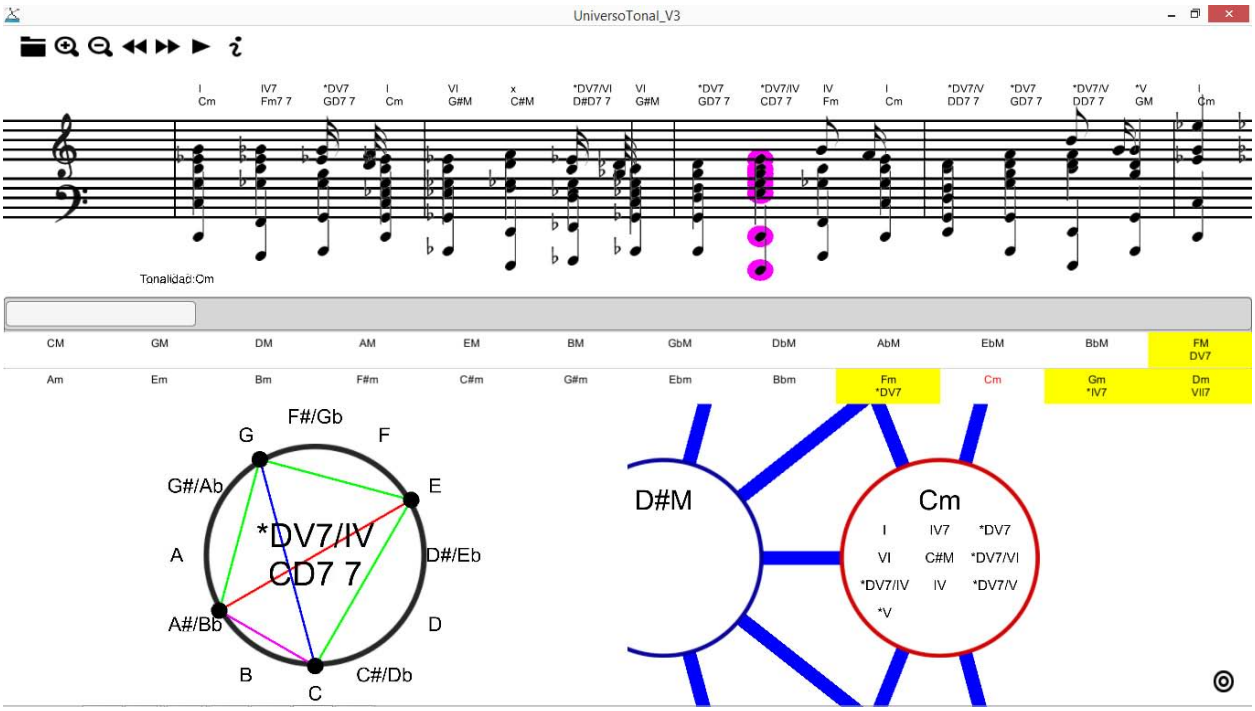


Figura 4.7: Preludio Op.28, No. 20 F. Chopin. Compases 1-2. Ejemplo de visualización de acordes en el círculo cromático. Compases 8 y 9. Visualización de regiones tonales en la tonalidad de *Do* menor.

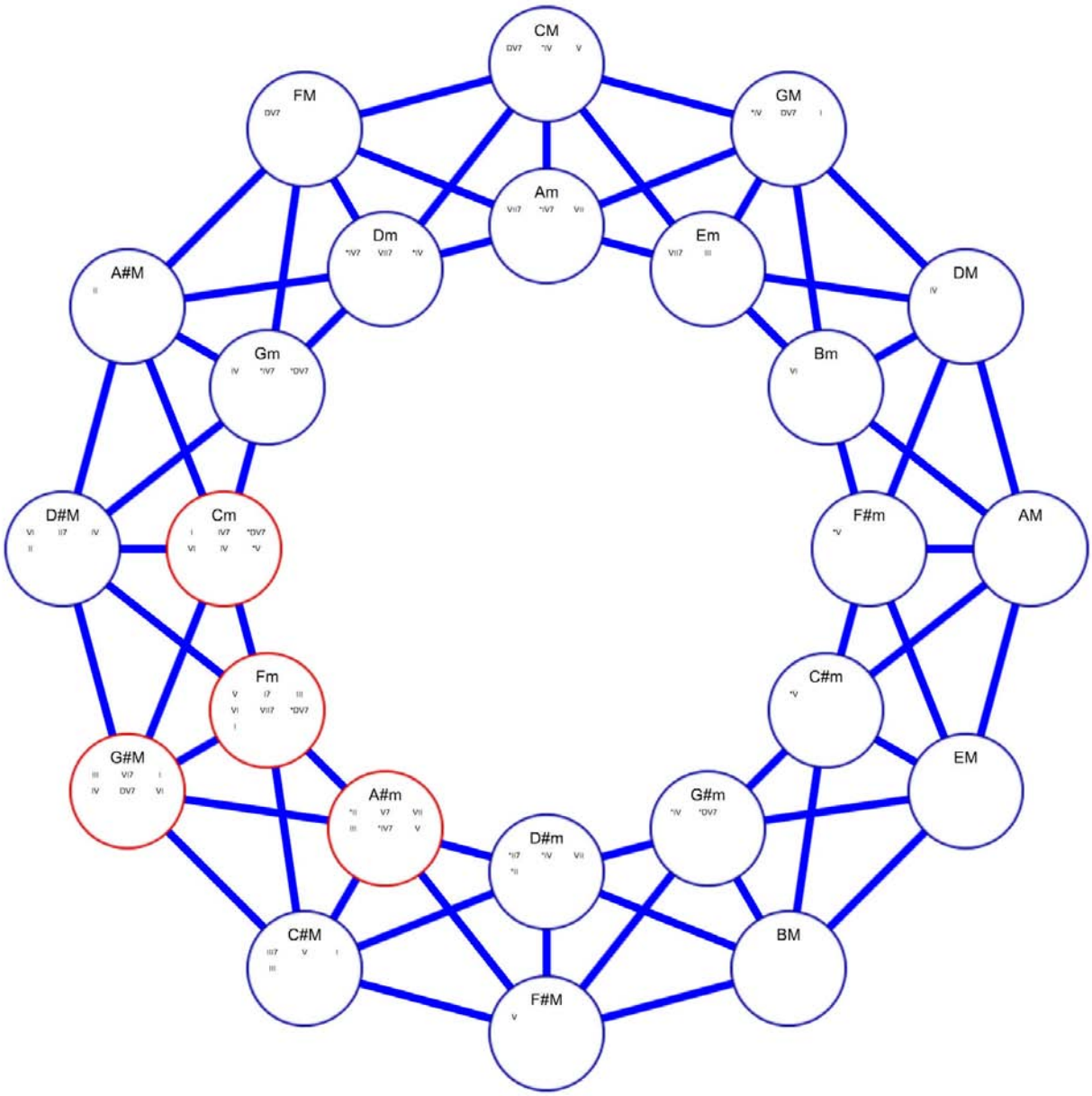


Figura 4.8: Preludio Op.28, No. 20 F. Chopin. Compases 1-2. Regiones tonales más probables de la pieza visualizadas en el Grafo Tonal.

Conclusiones

Esta investigación implicó un trabajo interdisciplinario que generó casi tantas preguntas como las que respondió. Lo que nació de una inquietud, quizá ingenua, de saber si es posible automatizar el etiquetado de acordes y clasificarlos de acuerdo con sus regiones tonales, se convirtió en un problema de definiciones musicales. En primera instancia, fue necesario crear un modelo computacional de la lista de todos los acordes y tonalidades, para poder tener métodos y funciones que facilitaran calcular las representaciones geométricas y relaciones tonales que existen entre ellos, al que llamamos *Universo Tonal*. El segundo paso fue implementar una forma de obtener la información musical de las partituras en formato digital. Se exploró con los formatos *MIDI* y *MusicXML* y se encontró que del primero existe un gran catálogo de obras que se pueden descargar de la Internet; sin embargo, este formato no ofrece información confiable de duraciones rítmicas, lo cual dificulta la lectura de no convertirlo a una partitura digital con editores tipo *Sibelius, Finale* y *MuseScore*. El segundo es un archivo ideal para los fines de este trabajo, ya que incluye toda la información musical que una partitura contiene; la desventaja es que es un formato muy reciente y el catálogo es más pequeño. Sin embargo, cualquier persona que utilice los editores mencionados puede exportar sus archivos a este formato y analizarlos. Ya que se obtuvo el modelo computacional de representaciones de acordes y tonalidades, junto con los métodos de extracción musical de archivos digitales, fue necesario crear la interfaz gráfica del programa con la que el usuario interactúa con el sistema. Esta se realizó en *Unity 5.0* debido a su portabilidad entre sistemas operativos.

Más allá del requerimiento técnico para el modelado y la creación del programa de cómputo, este trabajo encontró más dificultades y cuestiones por resolver en el ámbito de la delimitación

conceptual y términos musicales que utiliza. Uno de los problemas más grandes fue definir para qué tipos de obras musicales sería apropiada esta aplicación. El concepto de música tonal es demasiado amplio y no tiene una definición clara, la solución para delimitar la funcionalidad del programa fue sugerir ejemplos específicos de compositores con los que el software funciona, y dejar abierta la posibilidad para validar su uso con otras obras y estilos. Las obras elegidas fueron de los compositores: J.S. Bach, Mozart y Chopin debido a la claridad armónica que presentan en el manejo de los acordes y armonías en el contexto tonal. La intención inicial era crear una herramienta que tuviera ciertas habilidades y que resolviera un problema específico, en este caso el etiquetado de acordes y su clasificación en funciones tonales; el perfil de usuario, el ámbito de aplicación y distribución del programa son cuestiones que no forman parte de los objetivos de este trabajo de investigación. Se buscó presentar en pantalla la información tanto musical como de representaciones geométricas de la manera más clara y precisa, utilizando las notaciones adecuadas de cada área.

En retrospectiva, a pesar de que se llegó al objetivo planteado: ayudar al músico a automatizar el etiquetado de acordes y la clasificación por regiones tonales, es claro que este es un primer paso para algo mayor. Es necesario utilizar este conocimiento para analizar las obras de una manera más general y no tan microscópica. Una vez logrado esto, la herramienta será verdaderamente útil para entender mejor las dinámicas de tensión y relajación que aparecen en una pieza musical. Uno de los aportes más relevantes de este trabajo consiste en brindar una visión equilibrada de las cuestiones tanto matemáticas, computacionales y musicales, y por medio de ella saber identificar los problemas a resolver en la música y abordarlos con las herramientas conceptuales y pertinentes de la matemática y computación. Al trabajar de forma interdisciplinaria y buscar problemas a resolver, estos muchas veces pasan desapercibidos por los músicos por la simple inexistencia de una técnica para abordarlos y solucionarlos. Una cuestión que surge de este enfoque es que muchas veces pareciera que no hay problema a resolver, y sólo hasta que existe la solución y aparece como algo obvio es cuando se valora su utilidad. En cierta forma se necesita ir contra corriente y defender la visión propia; una vez que esté creada la herramienta, se podrá validar su uso como algo eficiente

o no, y construir sobre ella. En el caso del análisis musical armónico, hay mucho trabajo por desarrollar y esta tesis sirve como antecedente de la posibilidad de llevar el conocimiento teórico de la Teoría Matemática de la Música, al músico en la vida cotidiana por medio de herramientas de cómputo.

Al concluir este trabajo se encuentran nuevos problemas por abordar. Uno de los más importantes es utilizar los etiquetados y clasificación de acordes para analizar elementos de la forma de una obra. Sería necesario implementar algoritmos de reconocimiento de patrones que utilicen el resultado de este trabajo para encontrar de forma automática las diferentes secciones en las que se divide una obra, y así pasar del micro al macroanálisis. Este tipo de problemas se abordarán en el trabajo de doctorado, en el cual se aplicarán técnicas de inteligencia artificial y minería de datos para entrenar a un sistema en el uso de la información contenida en las partituras digitales. Los resultados del *Universo Tonal* servirán para encontrar información global de la pieza, buscar regiones de tensión y relajación. Esta información podría ser muy útil para el estudio de la música, ya que el análisis armónico busca, más allá de un etiquetado, entender la forma en que se maneja la tensión, la repetición y variación. Es de interés también saber si estas representaciones geométricas de los acordes pueden utilizarse como herramienta didáctica con estudiantes que recién comienzan su formación musical. La representación matemática de la música puede aportar información útil e interesante a músicos de distintos niveles. La única cuestión es presentar estos sistemas de modo que el usuario no tenga que realizar cálculos numéricos. Por otra parte, se tiene planeado investigar las formas en que estos procesos de representar matemática y computacionalmente los acordes, tonalidades y funciones tonales, se pueden usar para la composición de música tonal. Esto puede hacerse por medio de la creación de aplicaciones que en lugar de etiquetar los acordes en una partitura, utilicen directamente las geometrías para la creación de acordes. Para lograr esto se podría utilizar la tecnología de las pantallas táctiles, de modo que el usuario pueda seleccionar puntos en los espacios geométricos para activar ciertos acordes y secuencias armónicas.

Con el fin de integrar esta investigación al flujo de trabajo de un músico que utiliza herramientas digitales, se tiene planeado como proyecto futuro adaptar el programa de cómputo a fin de que sea

un *plugin* de software de edición de partituras como *MuseScore, Finale o Sibelius*, para que el músico pueda aprovechar el resultado de esta investigación sin tener que enfrentarse a la curva de aprendizaje de una herramienta de cómputo nueva.

En conclusión, el resultado de este trabajo es una aplicación de cómputo que permite analizar archivos musicales en formato MIDI y MusicXML utilizando técnicas de la Teoría Matemática de la Música. Se creó un sistema en el cual usuario puede navegar de forma interactiva en las regiones armónicas de una pieza por medio de distintas representaciones geométricas: el *Tonnetz*, *Círculo Cromático* y la *Banda de Moebius*. Como propuesta de este trabajo se desarrollaron las representaciones de *Grafo Tonal* y *Universo Tonal*. Este programa es el primero en su tipo que permite utilizar cada una de las representaciones mencionadas para el análisis de archivos musicales. En cuanto a las pruebas de análisis, el sistema funciona muy bien para obras tonales; se analizaron piezas de Bach, Mozart y Chopin. Los resultados de etiquetado de acordes y clasificación de acuerdo con sus regiones y funciones tonales fueron similares a los encontrados en la bibliografía de análisis tradicionales. La aplicación de cómputo se puede utilizar en las plataformas Windows XP+, Mac OS X 10.7+, Ubuntu 10.10+, y queda como trabajo futuro, realizar una versión para dispositivos móviles como tabletas y celulares. Por otra parte, la biblioteca de código creada está disponible en los lenguajes de programación *C#*, Java y SuperCollider, de modo que se puede usar como base para nuevos proyectos.

En este trabajo se sentaron las bases para un análisis algorítmico de información musical dirigido principalmente a un usuario con conocimientos científicos que desee adentrarse en su conocimiento matemático, así como un músico con afición científica e inquietud de conocer nuevas representaciones de la información musical que maneja a diario. Hay aún mucho por hacer para que este tipo de herramientas encuentren su lugar en el trabajo cotidiano del músico; sin embargo, mientras en el diseño y modelado se tenga en mente al usuario final y la facilidad de uso, poco a poco la práctica y el estudio analítico de la música, se beneficiarán de los procesos científicos que pueden simplificar su trabajo. Con esto se podrá utilizar menos tiempo en tareas repetitivas y mecánicas, para poder así enfocarse en cuestiones de creación estética y artística.

Apéndices

Apéndice A

Antecedentes matemáticos

A.1. Teoría de conjuntos

En esta sección se muestran los conceptos básicos de teoría de conjuntos necesarios para esta tesis, las definiciones toman como referencia los trabajos de DUMMIT y FOOTE (2003) y BENSON (2007). El primer concepto que necesitamos para organizar la información musical de manera abstracta es el de un **conjunto**, que es una colección de elementos. Se puede representar como una lista en corchetes, por ejemplo el conjunto de números pares del 1 al 10 sería $\{2,4,6,8,10\}$. En forma general un conjunto se puede describir sólo con las propiedades que cumple sin mostrar explícitamente todos los elementos:

$$A = \{a \in A \mid \ldots(condiciones\ de\ a)\ldots\}$$

Esta notación significa que el elemento a pertenece ($\in$) al conjunto A solamente si cumple las condiciones indicadas del lado derecho. Por ejemplo

$$\{a \in A | a\ es\ una\ nota\ de\ tecla\ blanca\} = \{do, re, mi, fa, sol, la, si\}$$

No importa el orden en que se presenten los elementos en el conjunto, sólo quiénes están en

él. Para mostrar algunas operaciones que se pueden realizar con ellos tomemos por ejemplo los siguientes tres: $A = \{1,2,3\}$, $B = \{3,4,5\}$, $C = \{1,7,8\}$:

- La unión $\cup$ consiste en crear un nuevo conjunto con los elementos de A y de B, sin repetir los que estén en ambos por ejemplo $A \cup B = \{1,2,3,4,5\}$, $A \cup C = \{1,2,3,7,8\}$. Ver figura (A.1)
- La intersección $\cap$ consiste en crear un nuevo conjunto con los elementos que simultaneamente se encuentran en A y en B, por ejemplo $A \cap B = \{3\}$, $A \cap C = \{1\}$. Ver figura (A.2)

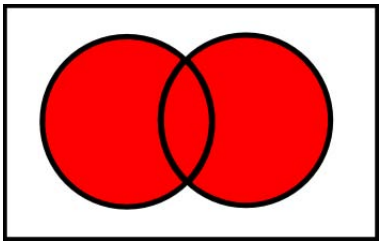


Figura A.1: Unión de conjuntos

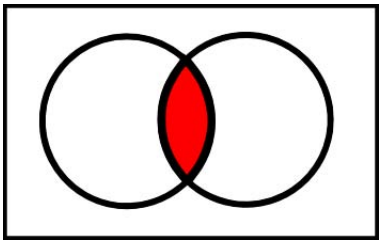


Figura A.2: Intersección de conjuntos

Se dice que A es un subconjunto de D, escrito como $A \subset D$ si todos los elementos del primero se encuentran también en el segundo. Por ejemplo, si $A = \{1,2,3\}$ y $D = \{0,1,2,3,7,4\}$, entonces se cumple $A \subset D$.

Definimos al conjunto vacío como al conjunto que no contiene ningún elemento, por definición éste es un subconjunto de cualquier otro conjunto y se suele representar por $\emptyset$.

El producto cartesiano de dos conjuntos $A \times B$ es el conjunto de todos los pares ordenados (a,b) donde $a \in A$ mientras que $b \in B$. En un par ordenado sí importa el orden de los elementos. Tomando los conjuntos de ejemplo que mencionamos arriba, el conjunto

$$A \times B = \{(1,3),(1,4),(1,5),(2,3),(2,4),(2,5),(3,3),(3,4),(3,5)\}$$

$$A^2 = A \times A = \{(1,1),(1,2),(1,3),(2,1),(2,2),(2,3),(3,1),(3,2),(3,3)\}$$

Se pueden multiplicar más de una vez los conjuntos de modo que queden triadas ordenadas, por ejemplo A^3 como conjunto representa todas las combinaciones de (a_1,a_2,a_3), mientras que A^4 son todos los (a_1,a_2,a_3,a_4).

A continuación necesitamos definir condiciones para saber cuándo los elementos están relacionados entre sí. Una relación binaria de un conjunto A es un subconjunto R de $A \times A$, es decir, es un conjunto de pares ordenados, y escribiremos $a \sim b$ si $(a,b) \in R$. Nos interesan en especial las **Relaciones de Equivalencia**, que son las que cumplen lo siguiente:

- Es *reflexiva*, es decir $a \sim a$ para todo $a \in A$.
- Es *simétrica*, donde $a \sim b$ implica que $b \sim a$, para todos los $a,b \in A$.
- Es *transitiva*, si tenemos $a \sim b$ y $b \sim c$ implica que $a \sim c$ para todos los $a,b,c \in A$.

Para ver un ejemplo de lo anterior definamos que dos notas están relacionadas entre sí, si pertenecen a la misma familia de notas, es decir si la distancia melódica entre ellas es un múltiplo de 12. De esta forma *Do6* estaría relacionado con *Do5*, y todos los demás *Do*. El número indica la octava de la nota y definiremos 4 como la octava central. Por su parte *Sol4* estaría relacionado con todos los demás *Sol*. Verifiquemos con algunos casos particulares que esta forma de asociar notas es una relación de equivalencia. En este caso el conjunto A será el conjunto de todas las notas. Mientras que la condición $\sim$ estará dada por condición de que pertenezcan a la misma familia de notas (*pitch class*). Veamos por ejemplo algunos pares de notas relacionadas: $Do6 \sim Do4$, $La3 \sim La8$, etc. Verifiquemos que cumplan las condiciones de clase de equivalencia. Es claro que $Do6 \sim Do6$ ya que ambas de manera obvia están en la misma familia, por lo tanto la relación es reflexiva. Por otra parte $Do4 \sim Do5$ implica que ambas están en la familia de *Do*, por lo tanto si las invertimos $Do5 \sim Do4$, ambas siguen estando en la misma familia, de modo que se cumple la simetría. Por otra parte si tenemos que $Do4 \sim Do6$ y $Do6 \sim Do8$, significa que tanto *Do4, Do6 y Do8* están en la familia de *Do*, por lo tanto de manera directa se cumple que $Do4 \sim Do8$, cumpliendo de esta manera la transitividad. Si revisamos esto para todos los elementos del conjunto

llegaremos a la conclusión de que es una relación de equivalencia.

Las relaciones de equivalencia toman importancia para nosotros al clasificar las formas en que los elementos pueden relacionarse. Si tomamos un elemento a del conjunto A, definimos su **clase de equivalencia** $[a]$ como el conjunto de todos quienes están relacionados con él, es decir $[a] = \{x \in A \mid x \sim a\}$. En el ejemplo de la relación de equivalencia pasado, en realidad sólo existen doce clases de equivalencia, una para cada familia de notas, o *pitch classes*.

Los conjuntos que utilizamos pueden ser muy grandes y es conveniente tener formas de relacionar los elementos entre sí, y encontrar cómo se comportan algunos de sus subconjuntos. En este sentido, una **partición** de un conjunto A es una colección de sus subconjuntos no vacíos que cumplen las siguientes dos condiciones:

- La unión de todos esos subconjuntos da como resultado al conjunto completo A.
- La intersección de cualquier par de esos subconjuntos es vacía, es decir, ningún par de subconjuntos tienen elementos en común.

Como ejemplo de una partición tomemos al conjunto A como las teclas de un piano. Si colocamos en el subconjunto A_1 todas las distintas versiones de *do*, sin importar la octava, en el conjunto A_2, todos los *do-sostenidos*, y así hasta llegar al conjunto de *si* A_{12}. De esta manera es claro ver que hemos dividido el conjunto de las teclas del piano en doce familias que no tienen elementos en común, y que si las unimos todas obtenemos una vez más el conjunto original A.

De lo anterior resulta una propiedad interesante. Si $\sim$ define una relación de equivalencia en A, entonces las clases de equivalencia forman una partición del conjunto. Por otra parte, si se tiene una partición del conjunto A, entonces existe una relación de equivalencia cuyas clases son precisamente los subconjuntos que forman la partición. La demostración de esta propiedad se puede ver a detalle en (DUMMIT y FOOTE, 2003). Las clases y relaciones de equivalencia son muy útiles para representar algunos elementos musicales como la equivalencia por octavas, que juega un papel fundamental en el desarrollo de esta tesis.

Además de conocer cómo se relacionan los elementos de un conjunto entre sí, es importante establecer relaciones con otros conjuntos, para esto se utilizan las funciones. Una **función** es una

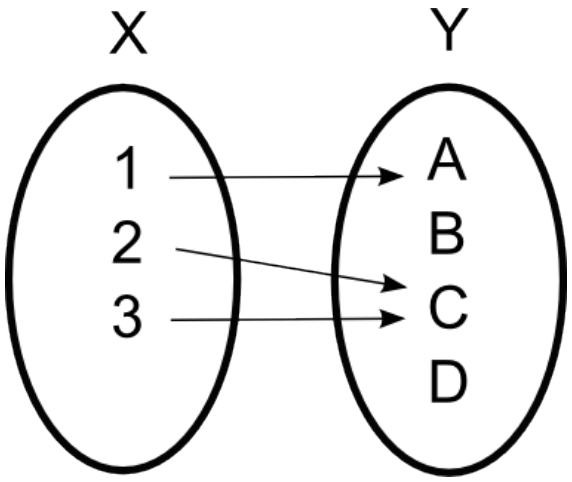


Figura A.3: Ejemplo de una función

regla que asocia a cada elemento de A, que llamaremos *dominio*, uno, y sólo un, elemento de B, nombrado *codominio*. Se denota por $f : A \to B$. En cuestión de elementos se asocian con la notación $f(a) = b$, o bien $f : a \mapsto b$.

Algo importante en la definición de una función es que todos los elementos del dominio deben tener un elemento del contradominio asignado, pero ningún elemento del dominio puede ser asignado a dos elementos del contradominio. Por ejemplo si tomamos la función $f : A \to B$, donde $A = \{1,2,3,4\}$ y $B = \{a,b,c\}$. Podemos definirla elemento por elemento como se muestra en la tabla (A.1). A cada elemento de A se le asigna uno de B, puede ser que dos del primero compartan uno del segundo, como es en el caso de 2 y 3, con b, pero no puede pasar que se defina adicionalmente $f(1) = a$, ya que 1 estaría definido de manera doble.

Para visualizar su lógica tomemos como ejemplo el conjunto de los números naturales $N = \{0,1,2,3,4,\cdots\}$. Definamos que tanto el dominio como el codominio igual a él $A = B = N$. Y definamos la función $f : N \to N$ como $f(a) = 2a$. Si seguimos la regla con los primeros elementos del conjunto veremos que $f(0) = 0$, $f(1) = 2$, $f(3) = 6$, $f(4) = 8$, $f(5) = 10$, etc. De modo que una función es una regla para asociar elementos de un par de conjuntos, que pueden ser el mismo, ver figura (A.3).

$$f(1) = c$$
$$f(2) = b$$
$$f(3) = b$$
$$f(4) = a$$

Tabla A.1: Ejemplo función

A veces es necesario enlistar los elementos de un conjunto de forma secuencial, para ello definimos las sucesiones. Una **sucesión** es cualquier función que tenga como dominio a los números naturales. Funciona como una lista de elementos que tienen como índice la numeración de los naturales, por ejemplo $\{n\}_{n=1}^{7} = \{1,2,3,4,5,6,7\}$. También se pueden definir operaciones de distintos tipos para crear la lista $\{4n\}_{n=3}^{5} = \{12,16,20\}$, incluso se pueden usar sólo como etiquetas $\{A_n\}_{n=0}^{6} = \{A_1,A_2,A_3,A_4,A_5,A_6\}$.

Cuando se requieren sumar estas listas se emplea la notación de series. Una **serie** es una suma escrita de forma condensada utilizando la notación

$$\sum_{i=0}^{4} x_i = x_0 + x_1 + x_2 + x_3 + x_4$$

La letra griega sigma representa que se va a sumar desde el índice que está en la parte inferior, hasta el de la parte superior. Por ejemplo:

$$\sum_{i=3}^{7} 2^i = 2^3 + 2^4 + 2^5 + 2^6 + 2^7 = 248$$

Con esto terminamos el breve repaso de conceptos y notaciones necesarias para manejar el contenido de esta tesis. Para mayor detalle y aclaraciones ver DUMMIT y FOOTE (2003).

A.2. Geometría

Las definiciones anteriores nos sirven para crear nuevos espacios. Como primer ejemplo tomemos las coordenadas en un plano. El conjunto de todas las coordenadas en dos dimensiones es el producto cartesiano $R^2 = R \times R$, por lo que una coordenada en un plano requiere dos parámetros, la

distancia horizontal y la vertical, es decir un par ordenado (x, y), cuyos valores son números reales. En la figura (A.2) se muestra un fragmento de este espacio con algunos ejemplos de coordenadas. A este espacio se le llama **espacio** R^n.

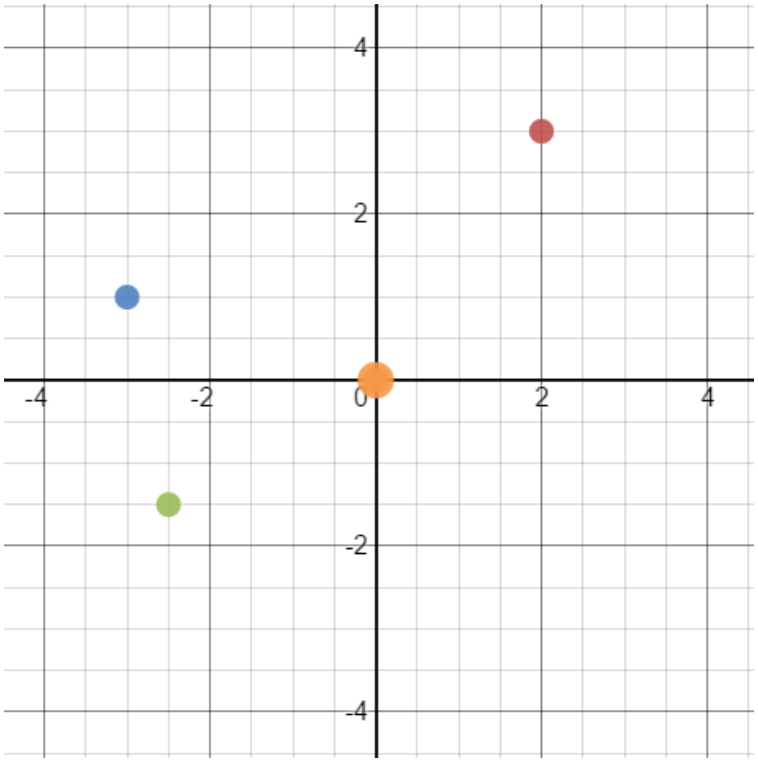


Figura A.4: Coordenadas en el plano de dos dimensiones.

Una vez que se tiene una representación abstracta de puntos en el espacio, surge de manera natural la necesidad de definir funciones y relaciones que nos den información sobre cómo se comporta la estructura. ¿Cómo podríamos medir una distancia entre dos puntos? La intuición nos dice que la distancia más corta entre dos puntos es una línea recta, así que podemos comenzar por trazarla y medirla como se ve en la figura (A.5).

Para encontrar numéricamente la distancia se utiliza la siguiente formula:

$$d(A,B) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

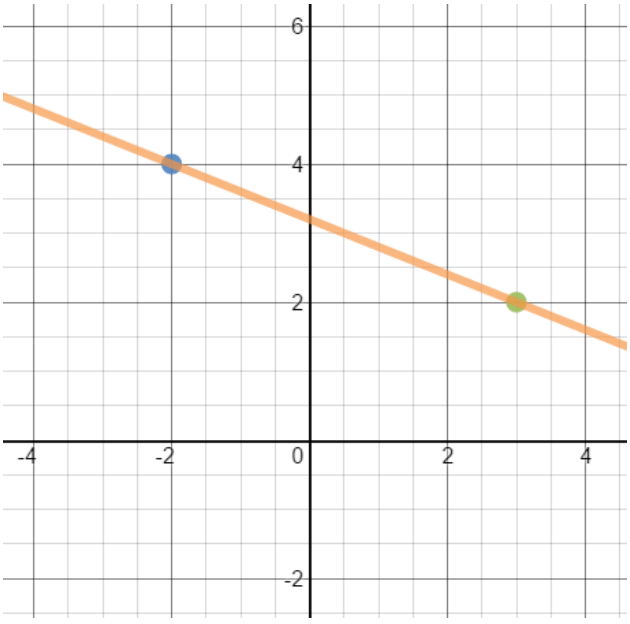


Figura A.5: Distancia euclidea

En el caso de la figura tenemos que $A = (-2, 4)$, mientras que $B = (3, -2)$

$$d(A,B) = \sqrt{(3-(-2))^2 + (-2-4)^2} = 7.81$$

Esta es la forma habitual de medir distancias en el espacio, sin embargo existen muchas otras maneras de hacerlo. El concepto de **métrica**, no en el sentido musical, sino en el sentido matemático de medición, se refiere a una función que define la distancia entre puntos.

En toda su formalidad, una métrica es una función $d : X \times X \to R$, que cumple las siguientes características de: no negatividad, identidad, simetría y desigualdad del triángulo, para todos los elementos $x, y, z \in X$. En términos coloquiales significa que cualquier distancia entre puntos es mayor o igual a cero; un par de puntos tienen distancia cero sí y sólo sí son el mismo punto; la distancia entre A y B es la misma que entre B y A; y que si conocemos la distancia entre A y B, junto con la de B y C, entonces la distancia entre A y C debe ser mayor o igual que la suma de las distancias A y B con la de B y C.

Como ejemplo de otros tipos de medir distancias, tomemos la *métrica del taxista*. En ella se toman dos puntos, y la distancia entre ellos será la forma más corta de llegar si sólo se puede

avanzar en unidades verticales y horizontales, es decir, como si se condujera un taxi en las cuadras de una ciudad, figura (A.6). En términos matemáticos si queremos la distancia entre los puntos (p_1, p_2) y (q_1, q_2) debemos calcular $|p_1 - q_1| + |p_2 - q_2|$

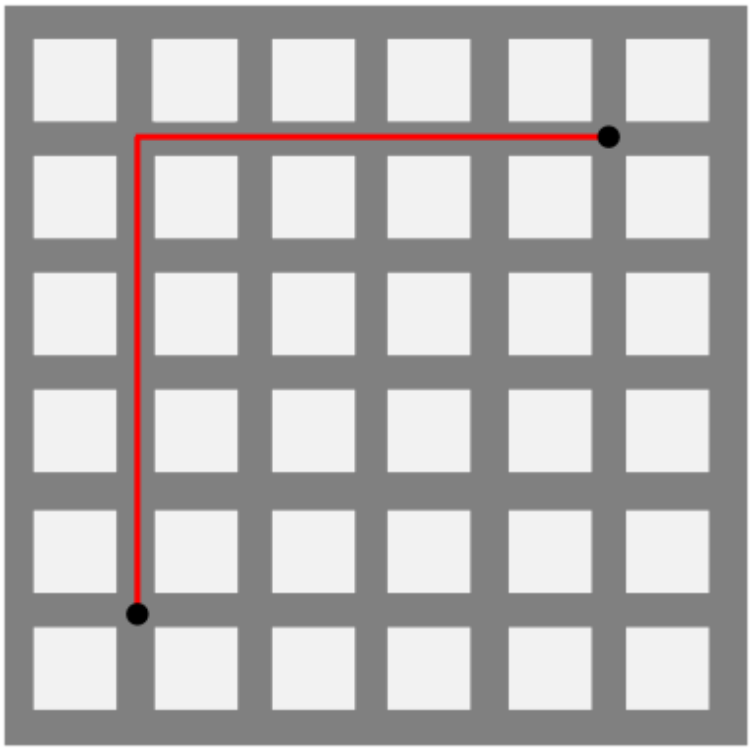


Figura A.6: Métrica del taxista.

Bibliografía

AEBERSOLD, JAMEY (1974). *Volume 1 [-] of A new approach to jazz improvisation*. J. Aebersold.

ALDWELL, EDWARD, C. SCHACHTER y A. CADWALLADER (2010). *Harmony and voice leading*. Cengage Learning.

ANDREATTA, MORENO y C. AGON (2003). *Implementing algebraic methods in openmusic*. En *Proceedings of the International Computer Music Conference, Singaphore*.

ARCHIVES, CLASSICAL (1994). *Classical Archives [Visto 14 de junio de 2015]*. `http://www.classicalarchives.com/midi.html`.

BARBANCHO, ANA M, I. BARBANCHO, L. J. TARDÓN y E. MOLINA (2013). *Database of Piano Chords: An Engineering View of Harmony*. Springer.

BENSON, DAVE (2007). *Music: A Mathematical Offering*. Cambridge University Press, UK and USA.

BIGO, LOUIS, J.-L. GIAVITTO, A. SPICHER y cols. (2013). *Spatial Programming for Musical Transformations and Harmonization*. 2013) colocated with AAMAS (W09), pág. 9.

BOON, JEAN PIERRE y O. DECROLY (1995). *Dynamical systems theory for music dynamics*. Chaos: An Interdisciplinary Journal of Nonlinear Science, 5(3):501–508.

CHEW, ELAINE (2000). *Towards a mathematical model of tonality*. Tesis de Doctorado, Massachusetts Institute of Technology.

CHEW, ELAINE (2008). *Out of the grid and Into the spiral: geometric interpretations of and comparisons with the Spiral Array model*. Tonal Theory for the Digital Age, Computing in Musicology, 15.

CHRISTENSEN, JESPER BØJE y J. B. ROBINSON (2002). *18th century continuo playing: a historical guide to the basics*. Bärenreiter.

COHN, RICHARD (1998). *Introduction to neo-riemannian theory: a survey and a historical perspective*. Journal of Music Theory, págs. 167–180.

COMMONS, WIKIMEDIA (2014a). *Banda de moebius [Visto 14 de junio de 2015]*. `https://en.wikipedia.org/wiki/Fiber_bundle#/media/File:Moebius_Surface_1_Display_Small.png`. Moebius.

COMMONS, WIKIMEDIA (2014b). *Enlarged Detail of automatic piano roll [Visto 14 de junio de 2015]*. `http://commons.wikimedia.org/wiki/File:Enlarged_Detail_of_automatic_piano_roll.gif#globalusage`. Piano Roll.

COOK, NICHOLAS (1987). *A guide to musical analysis*. G. Braziller New York.

COPE, DAVID (1996). *Experiments in musical intelligence*, vol. 12. AR editions Madison, WI.

COUPRIE, PIERRE (2015). *iAnalyze [Software] [Visto 14 de junio de 2015]*. `http://logiciels.pierrecouprie.fr/?page_id=176`.

CRANS, ALISSA S, T. M. FIORE y R. SATYENDRA (2009). *Musical actions of dihedral groups*. American Mathematical Monthly, 116(6):479–495.

CUNNINGHAM, STUART (2004). *Suitability of musicxml as a format for computer music notation and interchange*. En *Proceedings of IADIS Applied Computing 2004 International Conference, Lisbon, Portugal*.

DE HAAS, W Y MAGALHÃES, J y R. WIERING, FRANS Y C VELTKAMP (2013). *Automatic functional harmonic analysis*. Computer Music Journal, 37(4):37–53.

DE HAAS, WB (2012). *Music information retrieval based on tonal harmony*. Utrecht University.

DUMMIT, D.S. y R. FOOTE (2003). *Abstract Algebra*. Wiley.

ENDOLITH (2010). *Tonnetz on a torus. Attribution-NonCommercial-ShareAlike 2.0 Generic (CC BY-NC-SA 2.0) [Visto 14 de junio de 2015]*. `https://www.flickr.com/photos/omegatron/4648680687/`.

FORTE, A. (1973). *The Structure of Atonal Music*. Yale University Press.

FRASER, WILMOT ALFRED (1983). *Jazzology: A study of the tradition in which jazz musicians learn to improvise*. Tesis de Doctorado, Graduate School of Arts and Sciences, University of Pennsylvania.

FUJISHIMA, TAKUYA (1999). *Realtime chord recognition of musical sound: A system using common lisp music*. En *Proc. ICMC*, vol. 1999, págs. 464–467.

GIES, PETER (2009). *The flow of harmony as a dynamical system*. En *Mathematics and Computation in Music*, págs. 117–123. Springer.

GILBERT, JOHN (2015). *Chopin Prelude in C Minor [Visto 14 de junio de 2015]*. http://www.nyu.edu/classes/gilbert/19-20thC/larue.html.

GOOD, MICHAEL y cols. (2001). *MusicXML: An internet-friendly format for sheet music*. En *XML Conference and Expo*, págs. 03–04. Citeseer.

HARTE, CHRISTOPHER y M. SANDLER (2005). *Automatic chord identifcation using a quantised chromagram*. En *Audio Engineering Society Convention 118*. Audio Engineering Society.

HARTE, CHRISTOPHER, M. B. SANDLER, S. A. ABDALLAH y E. GÓMEZ (2005). *Symbolic Representation of Musical Chords: A Proposed Syntax for Text Annotations.*. En *ISMIR*, págs. 66–71.

HEDGES, THOMAS W y A. P. MCPHERSON (2013). *3D GESTURAL INTERACTION WITH HARMONIC PITCH SPACE*. Procedings of the Sound and Music Computation conference.

HOOBER, STEVEN y E. BERKMAN (2011). *Designing mobile interfaces*. .º Reilly Media, Inc.".

IRCAM, MUS3 (2015). *SOAL, Sonic Object Analysis Library OpenMusic [Visto 14 de junio de 2015]*. `http://grfia.dlsi.ua.es/cm/projects/drims/software.php`.

JEKOVEC, MATEVŽ (2015). *Harmonia - Music score analysis, Beta [Software] [Visto 14 de junio de 2015]*. `http://sourceforge.net/projects/harmoniamusic/`.

JOHNSON, TIMOTHY A (2008). *Foundations of diatonic theory: a mathematically based approach to music fundamentals*. Scarecrow Press.

KOSTKA, STEFAN, J. P. CLENDINNING, R. OTTMAN y J. PHILLIPS (1995). *Tonal Harmony with an Introduction to Twentieth-Ce*. McGraw-Hill.

KRÖGER, PEDRO, A. PASSOS, M. SAMPAIO y G. DE CIDRA (2008). *Rameau: A system for automatic harmonic analysis*. En *Proceedings of the 2008 International Computer Music Conference*, págs. 273–281. sn.

KRUEGER, BERND (1996). *Piano Midi [Visto 14 de junio de 2015]*. `http://www.piano-midi.de/`.

KULLAK, THEODOR (1882). *Nineteen Sonatas for the Piano*. Klavierwerke.

LARGE, EDWARD W (2011). *A dynamical systems approach to musical tonality*. En *Nonlinear Dynamics in Human Behavior*, págs. 193–211. Springer.

LERDAHL, FRED y R. JACKENDOFF (1985). *A generative theory of tonal music*. MIT press.

MAKEMUSIC (2015). *MusicXML, Version History [Visto 14 de junio de 2015]*. `http://www.musicxml.com/for-developers/version-history/`.

MANDEREAU, JOHN, D. GHISI, E. AMIOT, M. ANDREATTA y C. AGON (2011). *Z-relation and homometry in musical distributions*. Journal of Mathematics and Music, 5(2):83–98.

MAZZOLA, G. Y GÖLLER, S. (2002). *The Topos of Music: Geometric Logic of Concepts, Theory, and Performance*. Nº v. 1 en *The Topos of Music: Geometric Logic of Concepts, Theory, and Performance*. SPRINGER VERLAG NY.

MDECKS (2015). *Mapping Tonal Harmony Pro [Software] [Visto 14 de junio de 2015]*. `http://mdecks.com/mapharmony.html`.

MICHAEL, CUTHBERT (2015). *Music 21, Python library [Software] [Visto 14 de junio de 2015]*. `https://code.google.com/p/mingus/`.

MIDI (2015). *MIDI Manufacturers Association Incorporated, Tutorial: History of MIDI [Visto 14 de junio de 2015]*. `http://www.midi.org/aboutmidi/tut_history.php`.

MIDIWORLD (2015). *midiworld.com [Visto 14 de junio de 2015]*. `http://www.midiworld.com/classic.htm/`.

MOUTON, RÉMY Y PACHET, FRANÇOIS (1995). *The Symbolic vs. Numeric Controversy in Automatic Analysis of Music*. En *Artificial Intelligence and Music (IJCAI—95 Workshop Program Working Notes)*, págs. 32–40.

MUGELLINI, BRUNO (1964). *El clave bien temperado*. PWM: Warsaw.

MUSESCORE (1996). *Muse Score [Visto 14 de junio de 2015]*. `http://www.piano-midi.de/`.

NIITSUMA, MASAHIRO, M. MATSUBARA, M. OONO y H. SAITO (2011). *Development of a method for automatic basso continuo playing*. Information Processing & Management, 47(3):440–451.

ONCLASSICAL (2015). *Kunst der Fugue [Visto 14 de junio de 2015]*. `http://www.kunstderfuge.com/`.

ORDUÑA F., GODÍNEZ M. (2007). *Tecnologías de la información y la nueva economía - El impacto de las nuevas tecnologías en la educación musical superior de México*. Serie Estudios, Biblioteca de Ciencias Sociales y Humanidades, Universidad Autónoma Metropolitana, Unidad Azcapotzalco.

PARDO, BRYAN y W. P. BIRMINGHAM (2002). *Algorithms for chordal analysis*. Computer Music Journal, 26(2):27–49.

PETRUCCI, PROJECT (2015). *IMSLP-International Music Score Library Project [Visto 14 de junio de 2015]*. `http://www.imslp.org/`.

PISTON, WALTER (1978). *Harmony. Rev. and expanded by Mark DeVoto*. New York: WW Norton.

POPOFF, A. (2014). *An introduction to neo-Riemannian theory [Visto 14 de junio de 2015]*. `https://alpof.wordpress.com/2014/01/26/an-introduction-to-neo-riemannian-theory-9/`.

POPOFF, ALEXANDRE (2013). *Building generalized neo-Riemannian groups of musical transformations as extensions*. Journal of Mathematics and Music, 7(1):55–72.

PÉREZ, CARLOS (2015). *Harmonic analysis library for OpenMusic [Visto 14 de junio de 2015]*. `http://grfia.dlsi.ua.es/cm/projects/drims/software.php`.

RAMEAU, JEAN-PHILIPPE (1722). *(2012) Treatise on harmony*. Courier Corporation.

RE-COMPOSE (2015). *Liquid Notes [Software] [Visto 14 de junio de 2015]*. `http://www.re-compose.com/liquid-notes-music-software.html`.

RHIJNAUWEN, ONDERSTE (2015). *Mingus, Python library [Software] [Visto 14 de junio de 2015]*. `https://code.google.com/p/mingus/`.

RIEMANN, HUGO (1896). *Harmony simplified: Or, the theory of the tonal functions of chords*. Augener.

RUMBAUGH, JAMES, M. BLAHA, W. PREMERLANI, F. EDDY, W. E. LORENSEN y cols. (1991). *Object-oriented modeling and design. Vol. 1*, vol. 199. Prentice-hall Englewood Cliffs.

RUMBAUGH, JAMES, I. JACOBSON y G. BOOCH (1998). *The Unified Modeling Language Reference Guide*. Addison-Wesley.

SALZER, FELIX (1962). *Structural hearing: Tonal coherence in music, Vol. 1*. Houghton Mifflin Harcourt.

SCHOENBERG, ARNOLD (1978). *Theory of harmony*. Univ of California Press.

SCHOTTSTAEDT, BILL (1997). *Common music notation, Beyond MIDI*. MIT Press.

SETHARES, WILLIAM A (2005). *Tuning, timbre, spectrum, scale*. Springer Science & Business Media.

SHEH, ALEXANDER Y ELLIS, DANIEL PW (2003). *Chord segmentation and recognition using EM-trained hidden Markov models*. ISMIR 2003, págs. 185–191.

SIGMUND LEBERT, WILLIAM SCHARFENBERG (1893). *Nineteen Sonatas for the Piano*. G. Schirmer.

SPIER, NORM (2015). *NORM'S MUSIC VISUALIZER. MIDI-FILE BASED MUSIC (QUASI-) ANALYZER and MUSICAL-PERCEPTION-IMPROVEMENT SOFTWARE [Visto 14 de junio de 2015]*. `http://nastechservices.com/MidiAnalysis.html`.

TAUBE, HEINRICH (1999). *Automatic tonal analysis: Toward the implementation of a music theory workbench*. Computer Music Journal, 23(4):18–32.

TOUSSAINT, GODFRIED (2010). *Computational geometric aspects of rhythm, melody, and voice-leading*. Computational geometry: Theory and applications, 43(1):2–22.

TYMOCZKO, D. (2011). *A Geometry of Music: Harmony and Counterpoint in the Extended Common Practice*. Oxford Studies in Music Theory. Oxford University Press, USA.

TYMOCZKO, DMITRI (2009). *Three conceptions of musical distance*. En *Mathematics and computation in music*, págs. 258–272. Springer.

TYMOCZKO, DMITRI (2015). *ChordGeometries 1.1 [Software] [Visto 14 de junio de 2015]*. `http://dmitri.mycpanel.princeton.edu/ChordGeometries.html`.

UNITY, TECHNOLOGIES (2015). *Unity 5 [Software] [Visto 14 de junio de 2015]*. `http://www.unity3d.com/`.

WALTON, ADRIAN (2010). *A graph theoretic approach to tonal modulation*. Journal of Mathematics and Music, 4(1):45–56.

YOSHIOKA, TAKUYA, T. KITAHARA, K. KOMATANI, T. OGATA y H. G. OKUNO (2004). *Automatic Chord Transcription with Concurrent Recognition of Chord Symbols and Boundaries.*. En *ISMIR*.

YUST, JASON (2013). *Tonal prisms: iterated quantization in chromatic tonality and Ravel's 'Ondine'*. Journal of Mathematics and Music, 7(2):145–165.