Composición musical algorítmica a partir de secuencias de acordes

Grado en Ingeniería en Sonido e Imagen en Telecomunicación

Trabajo Fin de Grado

Autor:
Lucas A. Montiel Juan

Tutor:
Pedro José Ponce de León Amador

Junio 2017

Universitat d'Alacant
Universidad de Alicante

Agradecimientos

Podría dar las gracias a muchas personas.

Podría dar las gracias a mi tutor, Pedro, por su paciencia y ayuda a la hora de encarrilar el desarrollo de este proyecto. A profesores que, como Juanma, no sólo se limitan a enseñar. Y también un poco a los que sí. A mi familia. A mis padres por confiar en mi. A veces. Y por no hacerlo otras. Por su sinceridad, y comprender que mis errores son míos. A mi hermana y mi otra hermana, por su apoyo incondicional y haber regalado al mundo esa máquina de provocar sonrisas que es mi sobrino. Y a él, por ser. Ya tendrá tiempo de más. Podría dar las gracias a mis amigos, a los de aquí y a los de allí, por estar aunque no estén. Y al dueño del McLaren's por las veces que nos ha puesto aceitunas con la cerveza. Podría, incluso, dar las gracias a aquellas personas que se quedaron por el camino. A algunas de ellas.

Pero no lo voy a hacer.

Esto no es para ellos. Esto es para ti. Para recordarte que, a pesar de todo, puedes conseguir lo que te propongas. Porque llegaste hasta aquí. Y hasta donde quiera que estés ahora. Que seguro es más lejos de lo que puedo imaginar.

Gracias por haber terminado lo que empezaste. Y por seguir adelante. Una y otra vez.

Gracias por haber sido lo que soy.

Gracias por ser lo que seré.

Índice general

Índice de figuras

I

Introducción

Cuenta la leyenda que Pitágoras (s. V a.C), filósofo griego estudioso de las matemáticas y la música entre otros innumerables ámbitos, al pasar cerca de una forja, no pudo evitar centrar su atención en la "musicalidad" que producían varios herreros al golpear el metal. En concreto, un hecho particular llamó su atención: de los cuatro martillos que usaban en diferentes combinaciones, tres de ellos producían un sonido agradable o "armonioso"; y, sin embargo, la inclusión del cuarto en cualquiera de estas combinaciones, volvía al sonido producido algo estridente y desagradable. Pitágoras, por supuesto, no pudo evitar tratar de encontrar un motivo para aquello, de forma que tras estudiar los cuatro aparentemente similares martillos, concluyó que entre los tres primeros había una relación matemática simple: sus masas eran ratios simples o fracciones de cada uno de ellos; y no así el último, que no guardaba ningún tipo de relación aritmética aparente con el resto (Raasted, 1979).

La mencionada leyenda, demostrablemente falsa[1], no deja de ser una curiosidad que embellece el descubrimiento de un hecho que se ha considerado el principio de la armonía musical. Un hecho que pone de manifiesto la naturaleza cuantizable de la adecuada relación entre sonidos, o armonía, así como la existencia de reglas aritméticas concretas que rigen el desarrollo musical.

Lo que se entiende como música, el concepto en sí, es algo tan vivo como la sociedad con la que convive, y que evoluciona en consonancia a sus necesidades, gustos, y a las

[1] Se ha demostrado que las proporciones que se mencionan en la leyenda son de hecho relevantes en cuanto a longitudes de cuerdas, pero no así en lo respectivo a pesos de objetos metálicos y los tonos producidos por ellos.

herramientas que ésta pone a su disposición. Sin embargo, lejos de percepciones subjetivas o intergeneracionales sobre lo que puede ser considerado música por unos y algo más cercano al ruido por otros, las bases de la teoría musical moderna y la aritmética en ellas se han ido asentando a lo largo de los siglos sobre descubrimientos y experiencias de estudiosos, casualidades o curiosidad.

Desde el mismo momento en que estas reglas comenzaron a conocerse y entenderse, se han llevado a cabo innumerables intentos de crear música o secuencias musicales a partir de ellas, y no sólo como apoyo a la creatividad del compositor de forma que funcionen como guía, sino como una suerte de sustituto.

Se hace referencia aquí, por tanto, a lo que se conoce como composición algorítmica, algo que podría entender como un traductor universal para convertir en música cualquier elemento cuantizable de todo aquello que nos rodea. Esta definición, sin embargo, quizás sea una concepción más personal que objetiva, fruto de la curiosidad y el siempre escaso conocimiento de este ámbito de la composición al que se ha llegado tras su estudio.

Ya desde los tiempos de Guido d'Arezzo (s. X), considerado inventor de la notación musical moderna, se documentan intentos de la creación de secuencias musicales a partir de sistemas ajenos. A él mismo, se le atribuye también el primer sistema de composición algorítmica: un sistema basado en la correspondencia de las vocales de un texto con las diferentes alturas de una nota. Y algo más conocido es el denominado *Musikalisches Würfespiel*[2], atribuido a Mozart en el siglo XVIII, que se basaba en la aleatoriedad introducida por un par de dados a la hora de crear secuencias musicales en forma de *minueto*[3], eligiendo cada compás de una matriz de 16 columnas (16 compases) y 11 filas (valores de los dados de 2 a 12). Las diferentes secuencias que podrían generarse con cada uno de ellos rondan las 3^{15} posibilidades en el caso del primero y 10^{29} en el segundo (Diaz-Jerez, 2000).

A partir de aquí, teniendo en cuenta la increíblemente enorme cantidad de diferentes posibilidades que se pueden obtener con procesos tan relativamente simples, es normal plantearse hasta dónde se podría llegar si se introdujera en la ecuación la capacidad computacional de los sistemas informáticos actuales. De hecho, es algo que se lleva haciendo desde la propia creación de los ordenadores, y que tuvo un auge importante alrededor de los años 50. La informática ayudó al procesado de tan grandes como se requieran bases de datos de secuencias musicales de todo tipo para establecer estadísticas y probabilidades con las

[2] En alemán: Juego de Dados Musical.

[3] Forma musical de origen francés escrita normalmente en compases de tres tiempos y basada en danzas cortesanas antiguas.

que crear nuevas secuencias similares de forma automática, algo que se ha convertido en parte importante de prácticamente todo lo relacionado con la composición algorítmica asistida por ordenador.

Existen ya infinidad de sistemas de composición basados en la algoritmia, ya busquen un enfoque completamente automático, minimizando lo máximo posible la intervención humana, u otro más a modo de utilidad, que entrarían dentro de lo que se puede considerar composición asistida. La complejidad de estos sistemas sigue aumentando desde que se empezaron a implementar, y, probablemente, lo seguirá haciendo de forma ilimitada. Hasta que llegue, si llega, el momento en el que gracias a la inteligencia artificial, puedan ser considerados independientes.

I. 1 Motivación y objetivos

El presente proyecto nace de la unión de la pasión por la música junto con las inquietudes técnicas propias de un (futuro) ingeniero, en un intento por comprender un lenguaje de dominio específico enfocado al audio como es Csound, y las posibilidades que éste puede ofrecer en lo relacionado a composición algorítmica.

Apenas se han comenzado a vislumbrar las infinitas posibilidades que tanto el propio ámbito de la composición algorítmica como lo relacionado con el uso de ordenador para ello ofrecen. Por tanto, se pretende crear una aplicación que, más allá de considerarse un ente "completo", pueda mantenerse en constante crecimiento con la inclusión de nuevos algoritmos independientes o la modificación de los ya existentes.

Tras un estudio preliminar de los distintos métodos de composición algorítmica, se vio la posibilidad de no centrar el desarrollo de la aplicación en exhaustivos análisis estadísticos, ni tampoco dejar por completo el resultado a la aleatoriedad propia de este ámbito , sino buscar una suerte de equilibrio en el que se parta de unas probabilidades lógicas que sean variables en cierta medida. Estas variaciones responderían a una serie de reglas musicales implementadas mediante algoritmos específicos que no impondrían sus definiciones, sino que favorecerían las probabilidades de que así sucedieran. De esta forma se pretende alcanzar una solución más interesante desde el punto de vista musical, tratando de que el resultado posea o parezca poseer un comportamiento más "humano" sin dejar de ser una composición lo más automática posible, así como incentivar el atractivo a nivel usuario

que pueda tener introduciendo la posibilidad de que estos algoritmos funcionen de diferentes formas en función de lo que éste requiera.

En definitiva, lo que se pretende es la implementación de una sencilla aplicación de composición algorítmica híbrida basada en reglas y probabilidades, en la que como añadido para el usuario y también a modo de compartir las experiencias estudiadas durante el desarrollo de la misma, éste pueda decidir en qué medida actúan estas reglas o no, o los distintos algoritmos implementados, de forma que pueda comparar resultados o tener la posibilidad de formar parte del proceso de composición, convirtiendo a la aplicación en una posible ayuda a una composición creativa externa.

Para lograr estos objetivos se ha seguido un proceso concreto detallado a continuación:

- Estudio de las posibilidades en cuanto a sistemas de composición algorítmica y la plataforma a utilizar: Csound.
- Diseño preliminar de la aplicación a desarrollar en función de los resultados que se pretenden obtener.
- Implementación de la aplicación siguiendo el diseño y revisión de los aspectos problemáticos del mismo si los hubiera.
- Comprobación del funcionamiento y obtención de resultados.

I. 2 Estructura del proyecto

A modo de facilitar tanto el desarrollo del proyecto como su comprensión por posibles lectores, se ha trabajado siguiendo una estructura sencilla dividida en capítulos que abarque de forma concisa todos los aspectos necesarios para ello:

- **Capítulo 1: Introducción.** Presentación del proyecto.
- **Capítulo 2: Conocimientos previos.** Breve introducción a los conocimientos previos que sean considerados fuera del ámbito en el que se incluye el presente proyecto necesarios para el desarrollo y comprensión del mismo.
- **Capítulo 3: Análisis.** Análisis del estado de la tecnología y problema que se pretende solucionar. Solución que se propone y diseño previo del sistema.
- **Capítulo 4: Diseño.** Diseño definitivo del sistema, explicación de cada una de las partes y breves detalles de su implementación.

- **Capítulo 5: Resultados.** Uso del sistema y discusión de los resultados obtenidos.
- **Capítulo 6: Conclusiones.** Análisis final de objetivos cumplidos y posibilidades de mejora.

II

Conocimientos previos

Para el desarrollo de este proyecto se ha estudiado y puesto en práctica el conocimiento de ciertos aspectos técnicos concretos en ámbitos que podrían ser desconocidos para el lector. Por ello, y además de a modo de recapitulación en cuanto al proceso de investigación previo que se ha llevado a cabo, en el presente capítulo se procede a exponer brevemente unas nociones básicas de aspectos que facilitarán la comprensión del proyecto en general y detalles concretos del mismo que se explicarán en profundidad en capítulos siguientes.

II. 1 Teoría musical

Dada la naturaleza del proyecto, la componente de teoría musical es un aspecto de elevada importancia para la comprensión del mismo. Sin embargo, la literatura sobre este campo del conocimiento es extremadamente abundante y extensa, y su estudio detallado no forma parte de las competencias de este proyecto. Se procede, por tanto, a explicar de forma concisa los términos que se han considerado necesarios por formar parte explícita del diseño y/o implementación de la aplicación.

En primer lugar, cabe definir algunos conceptos que se usarán de forma continua a lo largo del presente y posteriores capítulos:

II. 1. 1 Conceptos básicos musicales

Características del sonido:

Desde el punto de vista musical, las dos características básicas del sonido o nota a tener en cuenta son:

- Altura o tono: Es la cualidad que determinará si un sonido es agudo o grave. Viene definida por la frecuencia de la vibración sonora. Cada nota y sus variaciones representan una altura.

- Duración: Se corresponde con el tiempo que se mantiene la vibración sonora, o en el caso presente, la nota o las notas en cuestión. Esta duración vendrá determinada por el ritmo o "*tempo*" definido para el conjunto.

Conceptos básicos y representación

Se añaden definiciones de elementos básicos de representación que serán empleados de ahora en adelante.

- Pentagrama: Símbolo gráfico sobre el que se expresa la grafía musical. Consta de cinco líneas horizontales equidistantes y numeradas desde abajo, y los cuatro espacios que se generan entre ellas.

- Clave: Símbolo que se usa como referencia en el pentagrama para determinar la ubicación de una determinada nota a partir de la cual construir el resto. En este proyecto se utilizará generalmente la clave de Sol pudiendo ser complementada ocasionalmente con una partitura en clave de Fa.

    - Clave de Sol: Indica que la nota Sol se encuentra en la segunda línea del pentagrama:
    - Clave de Fa en 4ª: Indica que la nota Fa se encuentra en la cuarta línea del pentagrama:

En la figura II. 1 se representan cada una de las claves con su respectiva nota referencia:

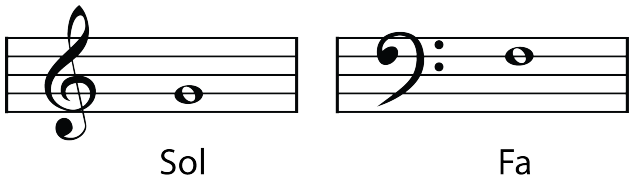

Figura **II. 1:** Representación de clave de Sol (izquierda) y clave de Fa (derecha) con sus respectivas notas de referencia.

- Compás: Unidad de tiempo en que se divide el pentagrama. Cada compás, a su vez, se subdivide en tiempos. La cantidad y duración de éstos vendrá definida por la fórmula de compás de la forma que se ilustra en la figura II.2. El numerador indica el número de tiempos en cada compás, y el denominador la figura que valdrá un tiempo. En el caso ejemplificado serán cuatro tiempos (numerador) de una negra cada uno (denominador).

  En el presente proyecto, por ser el compás más común y por simplificar el desarrollo de la aplicación, se empleará generalmente un compás de 4/4.

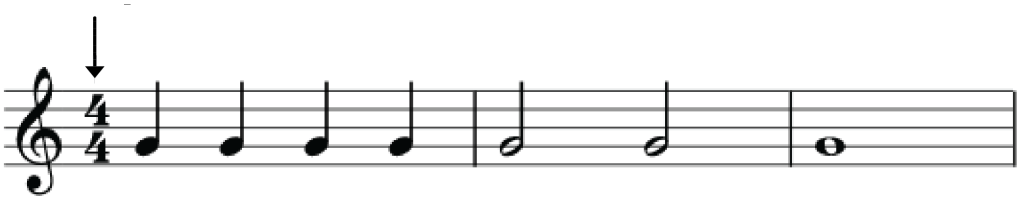

Figura II. 2: Representación de la fórmula de compás en una secuencia 4/4.

- Figura: Representación de la duración de la nota.

- Alteración: Modificación aplicable a las notas que aumentan o disminuyen la altura de todas aquellas de mismo nombre y octava dentro de un compás.

- Armadura de clave: Conjunto de alteraciones indicadas junto a la clave que hacen referencia a todas las notas de la partitura.

- Tempo: Velocidad de la pieza musical. Se expresa en pulsaciones por minuto (*ppm*, o más comúnmente en inglés: *bpm*). Generalmente una pulsación equivale a una negra.

Conceptos relacionados con la tonalidad

Antes de introducir conceptos más concretos en los que se requerirá cierta profundidad, cabe definir algunos más amplios (que quizá por ello suelen resultar confusos), y no dejan de ser necesarios para la comprensión no sólo de estos conceptos musicales empleados en el presente proyecto, sino también la toma de decisiones que se ha llevado a cabo en algunos puntos como los relacionados con la composición musical en sí.

- Tonalidad: Específicamente, es una determinada organización jerárquica de las relaciones entre distintas alturas en función de la consonancia sonora con respecto al centro tonal o tónica, que es una nota, su acorde y su escala diatónica (que se explicará en el siguiente apartado).

  En un sentido más amplio, hace referencia a la tónica, junto con todos los acordes y escalas asociadas, en torno a la cual giran las frases y progresiones musicales de una pieza.

- Grado: Posición de cada nota dentro de una escala musical en el sistema tonal. También se puede hablar de grado armónico para referirse a los acordes construidos dentro de una tonalidad. A cada nota de la escala diatónica le corresponde un grado al que se hace referencia mediante números romanos correlativos, además de una nomenclatura característica como se puede observar en la figura II.3.

II. 1. 2 Intervalos

El concepto de intervalo es muy importante en general en teoría musical y en particular para el presente proyecto, ya que es la característica que hace que una melodía o un acorde sea reconocible independientemente de su altura absoluta.

Se define como intervalo a la distancia en altura entre dos notas musicales. Se pueden distinguir en dos grupos: Intervalos armónicos (distancia entre dos sonidos simultáneos) e intervalos melódicos (distancia entre dos sonidos consecutivos).

En la música occidental, la mínima distancia que separará a dos notas será un semitono, por lo que para la medida de cada intervalo se hará uso de éstos y los tonos (suma de dos semitonos). De la misma forma, para expresarlos se utiliza el número de grados entre las notas en cuestión, incluyendo ambas, ya sea ascendente o descendente. Por ejemplo, un intervalo del grado I al III será un intervalo de tercera ascendente; mientras que uno del grado V al I será un intervalo de quinta descendente.

Grado	Nombre
I	Tónica
II	Supertónica
III	Mediante
IV	Subdominante
V	Dominante
VI	Superdominante
VII	Sensible

Figura **II. 3:** Representación y nomenclatura de los grados de una tonalidad.

Dependiendo de los tonos y semitonos que lo componen, los intervalos pueden ser justos o perfectos, mayores, menores, aumentados o disminuidos.

Además, existen otras clasificaciones en función de la naturaleza propia del intervalo:

- Intervalo conjunto: El intervalo se produce entre dos grados inmediatos (intervalo de segunda).
- Intervalo simple: Los sonidos que forman el intervalo no exceden la octava.
- Intervalo compuesto: Los sonidos que lo forman exceden la octava.
- Intervalo melódico: Las notas se tocan consecutivamente.
- Intervalo armónico: Las notas se tocan conjuntamente.

II. 1. 3 Escalas

Una escala se define como el conjunto de sonidos ordenados dentro de un entorno sonoro particular. Se clasifican según el número de notas que contengan pudiendo ser pentatónicas (cinco notas), hexatónicas (seis), heptatónicas o diatónicas (siete) y dodecafónicas o cromáticas (doce). A cada una de estas notas le corresponde un grado de la escala, como se ha explicado anteriormente. Dentro de cada uno de estos grupos existen distintas escalas que responden a la situación de los intervalos que las forman.

Este campo es extenso y complejo, por lo que en la presente explicación se incluirán sólo las escalas utilizadas en la implementación del proyecto.

Escala diatónica

En la música occidental el modelo de escala diatónica es el más común. Está formada por siete notas separadas por intervalos consecutivos de segunda mayor o menor, en concreto cinco intervalos de segunda mayor y dos de segunda menor. Dependiendo de la situación de estos intervalos en relación a los grados de la escala se distinguirán distintos tipos de escala.

Las escalas diatónicas más comunes son las contenidas dentro del conjunto llamado modos griegos, entre los que se encuentran la escala mayor natural y la escala menor natural. Estas serán en las que los intervalos de segunda menor se encuentran entre los grados III - IV y VII - VIII (escala mayor natural); y entre los grados II - III y V - VI (escala menor natural). Son conocidas como escalas naturales aquellas que no tienen alteraciones de ningún tipo.

Además de estas dos escalas, hay otras cinco dentro del conjunto de modos griegos que se corresponden con las distintas posiciones en las que se encuentren los intervalos antes mencionados. Por tanto, se pueden considerar las distintas escalas, o modos, en función de las distancias que haya entre sus notas de la forma que se expresa a continuación (T = tono, S = semitono figura), todas ellas se crean, como se ha comentado, a partir de la escala natural de Do empezando por cada una del resto de notas (figura II.4). Las escalas diatónicas que no son naturales en una tonalidad se crearán mediante el uso de alteraciones.

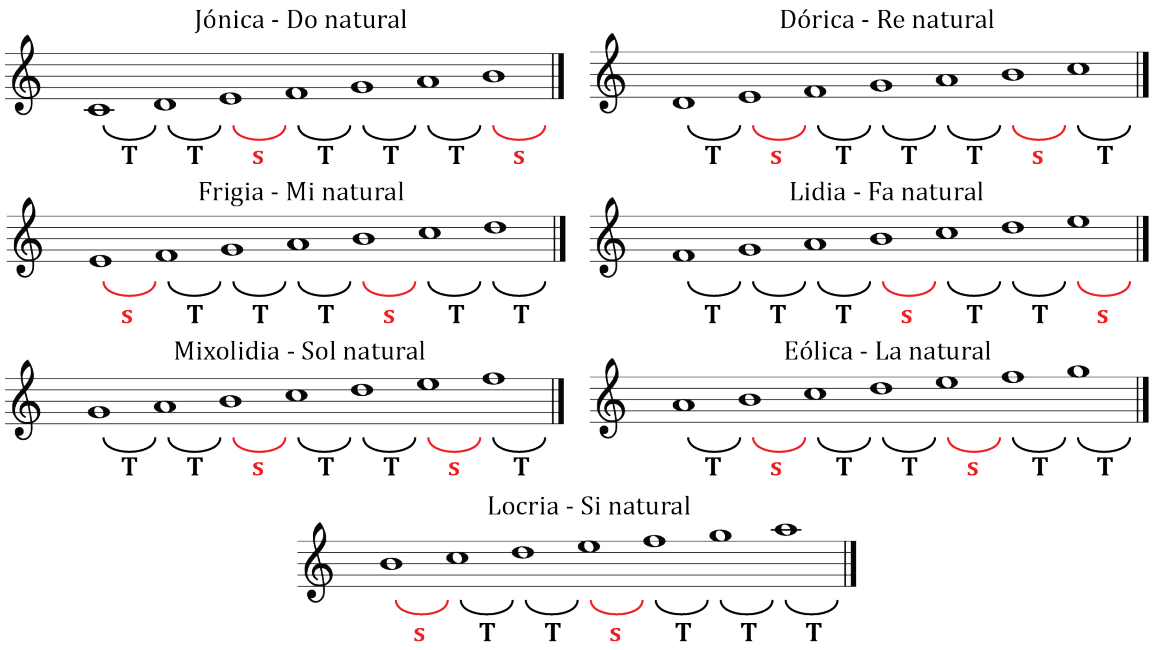

Figura **II. 4:** Los siete modos griegos en sus formas naturales con la posición de los intervalos.

Existen variaciones de estas escalas menos comunes en ámbitos generales pero que son usadas en diferentes estilos musicales. Para el desarrollo del presente proyecto, tan sólo se ha tenido en cuenta la escala Lidia b7, por ser una escala usada comúnmente en el jazz. Ésta se basa en la Lidia pero disminuyendo su séptima, por lo que tendrá una estructura diferente al resto de modos de la forma: T-T-T-S-T-S-T.

También existen otras dos escalas construidas a partir de la escala menor natural junto con las cuales forman el grupo conocido como escalas menores. Éstas son la escala menor armónica, escala menor natural con grado VII aumentado, y la escala menor melódica, con grados V y VII aumentados.

Escala pentatónica

Las escalas pentatónicas están construidas sobre cinco notas separadas entre sí por dos semitonos salvo entre los grados I – II, y los grados IV – V, que están separados por tres semitonos, para el caso de la pentatónica menor. Para el caso de la pentatónica mayor, la secuencia es la misma pero empezando por el segundo grado, es decir, las distancias de tres semitonos estarán entre los grados III – IV y los grados V-I. Esto, junto con las notas de blues que se comentarán en el apartado siguiente, se muestra en la figura II.5.

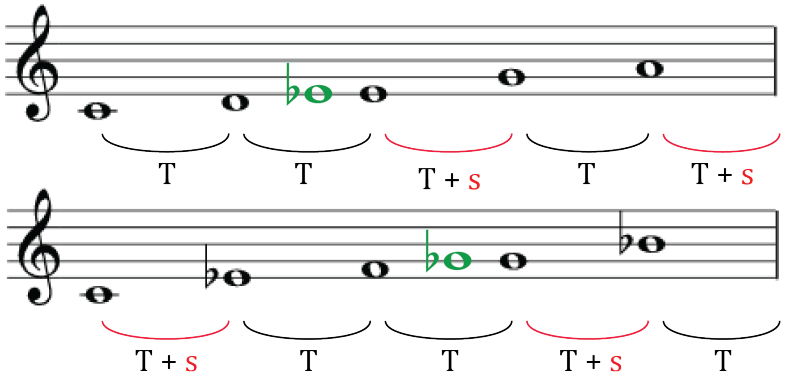

Figura **II. 5**: Escalas pentatónicas de Do mayor (arriba) y Do menor (abajo) con nota blues añadida.

Escala de blues

Las escalas de blues son escalas de seis notas que siguen el mismo patrón que la pentatónica pero añadiendo una quinta menor a la escala pentatónica menor, en el

caso de la escala de blues menor; y una tercera menor a la escala pentatónica mayor en el caso de la escala de blues mayor.

II. 1. 4 Acordes

Se conoce como acorde a un conjunto de tres o más notas que tradicionalmente se construyen por terceras sonando simultáneamente. Los acordes se forman sobre una nota base que será conocida como tónica, raíz o fundamental.

Acordes perfectos o tríadas

Los acordes de tres notas se conocen como tríadas. Éstos están formados por intervalos de tercera ascendente a partir de la tónica, con lo que formarían un conjunto de I, III y V para el caso de los acordes mayores. Las distintas variaciones de la III y la V darán lugar a los distintos tipos de acorde. Un ejemplo de la variación entre un acorde mayor y su respectivo menor se muestra en la figura II.6.:

- Mayor: Sin variaciones.
- Menor: Tercera menor (bemol).
- Disminuido: Quinta disminuida (bemol).
- Aumentado: Quinta aumentada (sostenido).

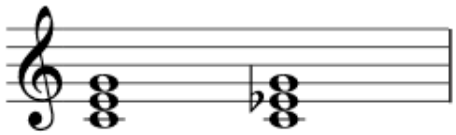

Figura **II. 6**: Acordes de Do mayor (I – III - V) y Do menor (I – IIIb - V)

Antes de entrar en el siguiente apartado, cabe definir un término más que será de vital importancia en la elección de los acordes de la composición. Éste será la llamada **función tonal**. Las funciones tonales o armónicas son clasificaciones jerárquicas que se les da a acordes, familias de acordes, progresiones melódicas o incluso frases musicales completas, que presentan características únicas en cierto sentido contextual (Terefenko, 2014); y, sobre todo, que interactúan de una forma concreta y sistemática entre ellas. Las funciones tonales principales se corresponden con los grados I, IV y V y son tónica, subdominante y dominante respectivamente.

A cada función tonal se le atribuyen unas características que puede corresponderse con percepciones subjetivas del oyente en términos tales como sensación de finalidad o inestabilidad. La relación entre las distintas funciones tonales está regida por la **tónica**, a la que se le atribuye los conceptos de estabilidad, descanso y cese de la secuencia armónica. La **subdominante**, por su parte, es la que propicia el movimiento armónico, siendo normalmente un paso intermedio entre la tónica y la dominante, o en menor, medida, en el paso contrario. La **dominante**, por tanto, es la que genera la inestabilidad y tensión armónica, que no descansará hasta volver a encontrar una tónica. Con estas definiciones, se llega a lo que puede ser considerada una de las sucesiones armónicas básicas sobre la que se construyen infinidad de secuencias musicales, la secuencia I – IV – V o tónica – subdominante – dominante. En la figura II.7 se muestran las relaciones de acordes de cuatríadas en distintos grados con sus respectivas posibles funciones tonales.

Acordes de cuatríadas

En este apartado se extenderá en cierta medida la explicación, ya que este tipo de acordes serán los empleados como base de la composición musical competencia del presente proyecto.

Los acordes de cuatríadas se forman al añadir un nuevo intervalo de tercera a un acorde de tríada. El nuevo acorde tendrá, por tanto, cuatro notas que se corresponderán con los grados I, III, V y VII de la tónica o raíz, y son conocidos comúnmente como acordes de séptima en cualquiera de sus variantes, o de sexta, si ésta sustituye a la séptima. De nuevo, en función de las variaciones de III, V y VII se obtendrán las distintas variaciones del acorde, cada una de las cuales será favorable a realizar una función tonal concreta, entre las que podremos encontrar las siguientes:

Categoría mayor

En este grupo se incluyen aquellos que mantienen la tónica y la tercera mayor. Las funciones tonales que suelen realizar son de tónica y subdominante. Entre ellos se encuentran los siguientes:

- Acorde de sexta (Do6, Sol6…): Tendrán, además de tónica y tercera mayor, la quinta justa y la sexta (I, III, V, VI).
- Acordes mayores de séptima (ReM7, SiM7…): Añadirán al conjunto de acorde mayor una séptima mayor (I, III, V, VII).

- Acordes de séptima mayor con quinta bemol (LaM7b5…): Serán como los anteriores pero con la quinta disminuida (I, III, bV, VII).
- Acordes de séptima mayor con quinta aumentada (SiM7#5…): Se construyen de forma análoga al caso anterior sólo que aumentando la quinta (I, III, #V, VII).

Categoría menor

En esta categoría se encuentran aquellos con tónica, quinta y tercera menor. Suelen realizar las funciones de tónica y subdominante. Son los siguientes:

- Acordes menores de sexta (Dom6, Rem6…): Formados por tónica, tercera menor, quinta y sexta (I, bIII, V, VI).
- Acordes menores de séptima (Fam7, Lam7…): Formados por tónica, tercera menor, quinta y séptima menor (I, bIII, V, bVII).
- Acordes menores de séptima mayor (Remin#7[1]…): Los forman tónica, tercera menor, quinta y séptima mayor (I, bIII, V, VII).

Categoría de séptima dominante

En este grupo se incluyen aquellos con tónica y séptima menor. Como su nombre indica, suelen realizar función de dominante.

- Acordes de séptima dominante (Sol7, La7…): Se forman con tónica, tercera mayor, quinta y séptima menor (I, III, V, bVII).
- Acordes suspendidos de séptima (Dosus7, Resus7): Éstos incluyen una cuarta en lugar de la tercera, por lo que se formarían mediante tónica, cuarta, quinta y séptima menor (I, IV, V, bVII).
- Acordes de séptima dominante con quinta bemol (Do7b5…): Se forman con tónica, tercera mayor, quinta disminuida y séptima menor (I, III, bV, bVII).
- Acordes de séptima dominante con quinta aumentada (Re7#5…): Se forman con tónica, tercera mayor, quinta aumentada y séptima menor (I, III, #V, bVII).

[1] El # aquí se usa como notación para el acorde pero carece del significado habitual, en este caso se refiere a que la séptima es mayor y no a que se le aumenta el semitono correspondiente

Categoría intermedia

Son aquellos que comparten tónica, tercera menor y quinta disminuida. Según el contexto, pueden ejercer cualquiera de las tres funciones tonales principales.

- Acordes de séptima menor con quinta bemol (Fam7b5…): También conocidos como semidisminuidos (Faø), se forman con tónica,tercerca menor, quinta disminuida y séptima menor (I, bIII, bV, bVII).
- Acordes de séptima disminuida(Sol°…): Se forman con tónica, tercera menor, quinta disminuida y séptima doble disminuida[2] (I, bIII, bV, bbVII).
- Acorde disminuido de séptima mayor(Do°#7[3]): Se forman con tónica, tercera menor, quinta disminuida y séptima mayor (I, bIII, bV, VII).

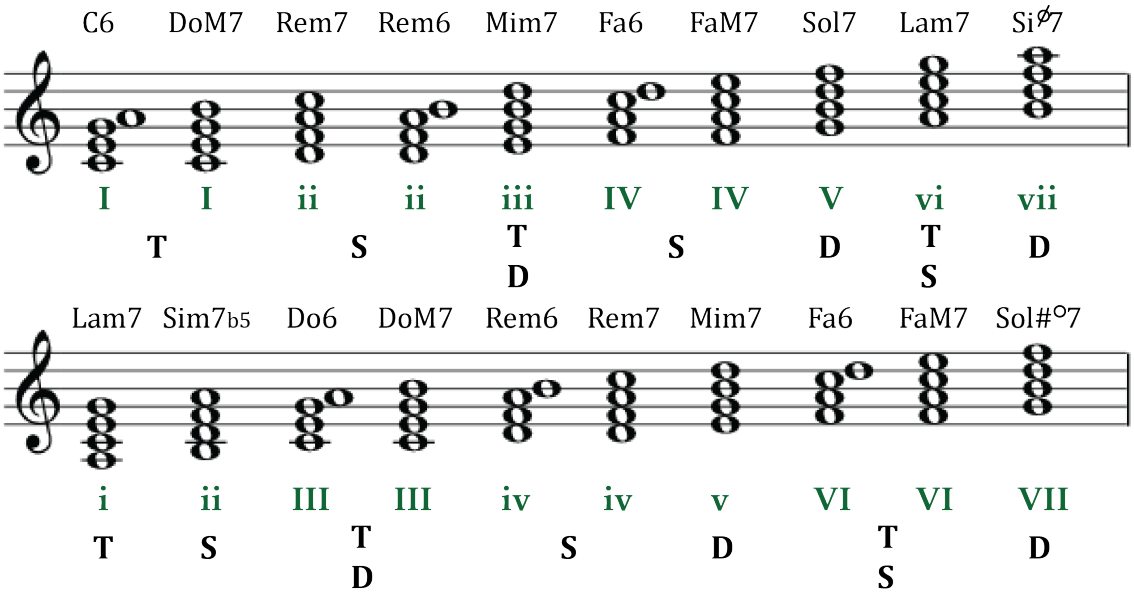

Figura **II. 7**: Relaciones de grados de una tonalidad con sus respectivos acordes de cada función tonal para tonalidades mayores (arriba, DoM natural) y tonalidades menores naturales (abajo, Lam Natural).

La función tonal que realice el acorde viene definido en primera instancia por el tipo de acorde, pero es de igual importancia para llegar a esta definición localizar el grado en el que se sitúa dicho acorde respecto a la tonalidad de la secuencia musical, ya que este

[2] Se mantiene la nomenclatura como séptima, y no como sexta, a pesar de ser equivalentes, dado que las funciones que realiza se asemejan más a las de una séptima que a las de una sexta.

[3] De nuevo el # se usa en esta notación para hacer referencia a la séptima mayor, y no como aumento de un semitono.

parámetro será el que defina la altura relativa del acorde, es decir, la distancia respecto a la tónica de la tonalidad. La relación entre grado, tipo de acorde y tonalidad será la base sobre la que se crearán las secuencias de acordes en el presente proyecto, y se seguirán los patrones teóricos mostrados en la figura II.7.

Al tratarse de acordes dentro de una tonalidad, se hace referencia a ellos mediante la nomenclatura típica de grados en números romanos, siendo éstos en mayúscula para referirse a acordes mayores y en minúscula para referirse a acordes menores.

Inversiones

Las inversiones de un acorde hacen referencia a la posición relativa de sus notas en el pentagrama. En su estado fundamental, la nota de menor altura del acorde será su tónica o fundamental, en cualquier otro caso, se considerará una inversión pudiendo haber tantas como notas del acorde que no sean la tónica. En el caso de estudio, los acordes de cuatro notas, se podrán considerar tres inversiones por acorde:

- Primera inversión: Lota de menor altura será el grado III.
- Segunda inversión: La nota de menor altura será el grado V.
- Tercera inversión: La nota de menor altura será el grado VII.

Las inversiones conservan su nomenclatura en términos ascendentes o descendentes. Sin embargo, para la simplificación de distintos aspectos de la implementación y por tanto la explicación del siguiente proyecto, se tomará la licencia de usar términos de inversiones negativas tomando como referencia el estado fundamental de un acorde. Un ejemplo de esto se puede ver en la figura II. 8.

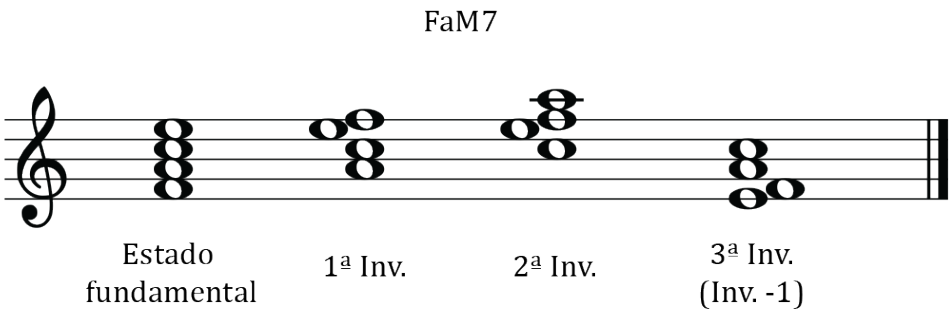

Figura **II. 8**. Distintas inversiones del acorde de FaM7 (I, III, V, VII).

II. 2 Composición algorítmica

La composición algorítmica, a la que también se le puede hacer referencia como composición automática, se define básicamente como "el proceso de usar procedimientos formales para crear música con la mínima intervención humana" (Alpern, 1995). Se puede considerar que esos procedimientos formales se han usado desde tiempos antiguos, sin embargo, el título concreto que hace referencia a estos procesos es relativamente nuevo. "Algoritmo", es un término que ha sido adoptado de los campos de la informática desde alrededor de mitad del s. XX. Los ordenadores han dado infinidad de nuevas herramientas y facilidades a la hora de automatizar procesos de composición que han sido estudiados y desarrollados a lo largo de las décadas.

Se definirá algoritmo, por tanto, como un conjunto de reglas o secuencia de operaciones diseñados para cumplir una tarea o resolver un problema en un determinado número de pasos (Maurer, 1999). En este caso, el problema a resolver será el componer una secuencia musical. La estructura y manera de procesar los datos de los considerados algoritmos de composición, serán lo que categoricen el tipo de composición algorítmica. Principalmente se podrían distinguir los siguientes (Papadopouluos & Wiggins, 1999):

- Basados en modelos matemáticos.
- Basados en el conocimiento (estadístico).
- Métodos evolucionarios o basados en algoritmos genéticos.
- Gramática.
- Sistemas de autoaprendizaje.

Generalmente, los procesos de composición algorítmica basados en un solo tipo de algoritmia no suelen conseguir objetivos satisfactorios en términos estéticos en cuanto a la composición. Por ello, se definen los sistemas híbridos como aquellos que emplean las bondades de dos o varios de estos métodos para conseguir resultados.

No se definirán los distintos métodos de composición algorítmica salvo aquellos que formen parte del presente proyecto. Por ello, se tendrán en cuenta en la siguiente explicación métodos basados en procesos considerados dentro de los dos primeros grupos antes mencionados, como son los estocásticos y los estadísticos.

II. 2. 1 Métodos basados en procesos estocásticos

Los procesos estocásticos se centran en la modelización de sistemas que evolucionan a lo largo del tiempo, o el espacio, de acuerdo a unas leyes no determinísticas, esto es, de forma total o parcialmente aleatoria. En términos técnicos, un proceso estocástico se define como una colección de variables aleatorias $\{X_t : t \in T\}$ parametrizada por un conjunto T, llamado espacio parametral, en donde las variables toman valores en un conjunto S llamado espacio de estados (Rincón, 2012).

Dentro de los procesos estocásticos podemos encontrar distintas clasificaciones en función de la forma de la variable a calcular, el tiempo en el que hacerlo y la forma de hacerlo. Así, en relación al tiempo, se encuentran:

- Procesos estocásticos en tiempo discreto: El valor de la variable sólo puede cambiar en una serie de momentos determinados en el tiempo.
- Procesos estocásticos en tiempo continuo: El valor de la variable puede cambiar en cualquier momento del tiempo.

Y en relación a la variable:

- Procesos estocásticos de variable discreta: La variable sólo puede tomar determinados valores o estados concretos.
- Procesos estocásticos de variable continua: La variable puede tomar cualquier valor comprendido en un rango.

En el presente proyecto, y en general en la mayoría de procesos de composición algorítmica, el tiempo de cálculo de variables será un parámetro que variará en función de un parámetro proporcional al tempo de la secuencia musical a obtener; al igual que la variable final, que estará definida por los valores frecuenciales de las distintas alturas de las notas posibles. Por tanto, los procesos estocásticos que se llevarán a cabo podrán ser incluidos dentro de los grupos de tiempo y variable discreta. Sin embargo, es común que los procesos de composición algorítmica se basen en secuencias de distintos procesos o se apoyen en variables continuas intermedias para llegar a las variables finales.

Otra clasificación que quizás pueda ser considerada de más importancia por definir los resultados en sí la forma en la que se obtengan los valores del proceso, vendrá dada en función de si las variables se obtienen en cada momento mediante sucesos independientes o mediante sucesos dependientes de algún estado anterior o posterior. A continuación se

procede a introducir brevemente los aspectos de cada una de ellas que sean considerados de relevancia para el desarrollo del proyecto.

Sucesos independientes

Un proceso estocástico basado en sucesos independientes se caracteriza por que los valores obtenidos no dependen de ningún valor o estado previo. Se representa así una sucesión de estados independientes de un mismo experimento aleatorio como en el ejemplo expuesto en la figura II.9.

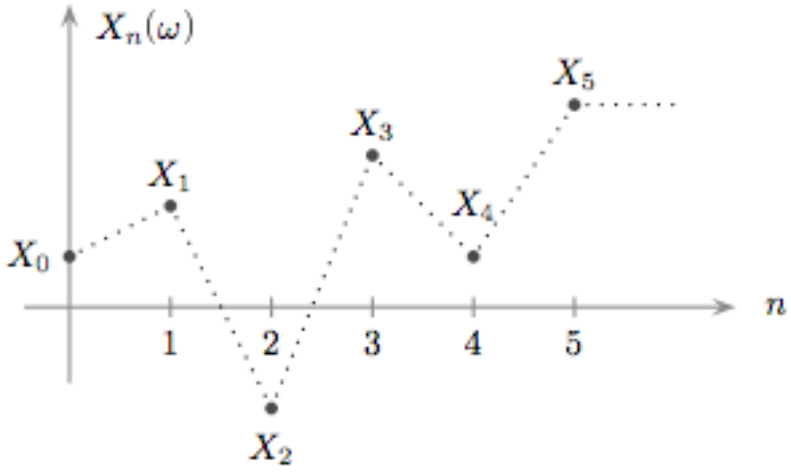

Figura **II. 9**: Representación de un sistema estocástico de resultados X_n en un tiempo *n*.

Si estos valores son discretos, las probabilidades relacionadas también lo serán, al contrario que si éstos son continuos, que se definirán entonces a partir de una función densidad de probabilidad.

La codificación de probabilidades para un suceso se realiza con valores entre 0 y 1, siendo éstos los valores límites que definirán que el suceso no pueda pasar o pase siempre respectivamente. Por tanto, es lógico pensar que la suma de todas las probabilidades relacionadas a un suceso debe ser 1.

En composición algorítmica, como se ha comentado, es común emplear también funciones de densidad para establecer las variables que decidirán el suceso final. Existen distintos tipos de funciones de densidad que pueden resultar de utilidad entre las que las más usadas serán distribución uniforme, distribución lineal, distribución triangular y distribución exponencial.

Sucesos dependientes

En los sucesos dependientes, los estados presentes o futuros pueden haber influido de alguna forma definida en el estado actual. Éstos pueden ser de diversos tipos en función de dicha influencia, en el presente estudio se comentará el caso concreto que se emplea en el desarrollo del proyecto, conocido como cadenas de *Márkov*.

Cadenas de *Márkov*

Este tipo de procesos se definen como aquellos en los que, suponiendo conocido el estado presente del sistema, los estados anteriores no tienen influencia en los estados futuros del sistema. Dada esta propiedad, conocida como propiedad de *Márkov*, se obtienen sucesos en los que la probabilidad del evento futuro depende sólo del estado actual, mientras que la información correspondiente al estado anterior es irrelevante.

Por tanto, se puede definir matemáticamente que un proceso estocástico tomando valores en un espacio concreto se considera una cadena de *Márkov* de orden uno si:

$$\mathbb{P}[X_{n+1} = i_{n+1} \mid X_n = i_n, X_{n-1} = i_{n-1}, \dots, X_1 = i_1, X_0 = i_0] = \mathbb{P}[X_{n+1} = i_{n+1} \mid X_n = i_n]$$

para todo $n \in N_0$, todo $i_0, i_1, \dots, i_n, i_{n+1} \in S$, siempre que las dos probabilidades condicionales estén bien definidas.

Las cadenas de *Márkov* se representan mediante un grafo llamado *grafo de transición* a modo de mapa entre los distintos estados, y una matriz llamada *matriz de transición* en la que se codifican las distintas probabilidades entre estados.

Estas cadenas son de elevada utilidad para diversas disciplinas del conocimiento, y en concreto para el caso que ocupa al presente proyecto. Éstas suelen usarse en composición algorítmica conjuntamente con características de los modelos basados en el conocimiento, es decir, estadísticos, en los que se analizan secuencias musicales de tipos concretos para establecer los valores de probabilidad entre los distintos estados posibles.

	S1	S2	S3
S1	0.2	0.1	0.7
S2	0.5	0.3	0.2
S3	0.0	0.4	0.6

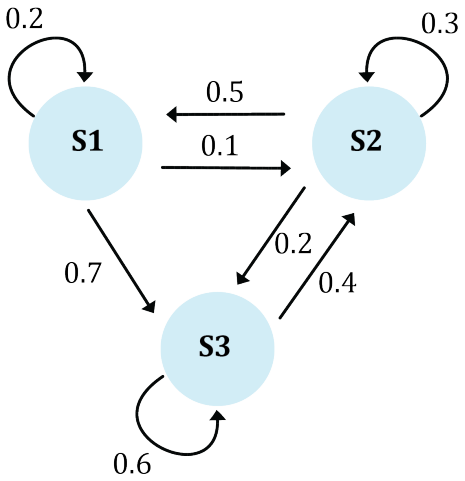

Figura **II. 10**: Representación de la *matriz de transición* y su *grafo de transición* asociado de una secuencia de *Márkov* de tres estados.

Modelo oculto de *Márkov*

Los modelos ocultos de *Márkov* son aquellos en los que se asume que el sistema a modelar es un proceso de *Márkov* de parámetros desconocidos (De Roux, 2015). Esto significa que el estado actual no es visible directamente, al contrario que en la cadena de *Márkov*, en la que los estados son observables y por tanto se conocen tanto el actual como el anterior. Al no ser visibles los estados, no se tiene información sobre ellos, sino tan sólo de los parámetros que estos ofrecen como salida, pudiendo ser a su vez varios y con distintas probabilidades.

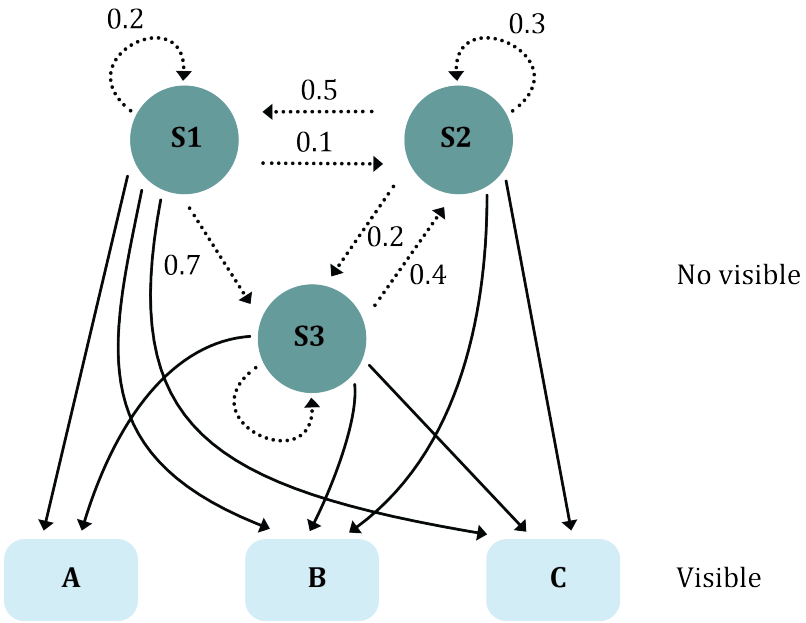

Figura **II. 10**: Representación del grafo de transiciones de la figura II. 9 convertido en un modelo oculto de Márkov.

III

Análisis

En este capítulo de procede a comentar los distintos tipos de análisis que se han llevado a cabo para la realización del presente proyecto. Desde el análisis de los sistemas de composición algorítmica existentes en la actualidad o las plataformas dedicadas a ayudar en la creación, estudio o favorecer el desarrollo de las mismas; hasta el análisis de problema o cuestión que se pretende resolver con la aplicación propuesta y laos primeros detalles de la solución y las bondades de la misma que se ha tomado en consideración. Pasando, por supuesto, por el análisis de las herramientas que se emplearán en el desarrollo de la misma.

En la actualidad, existen infinidad de estudios sobre composición algorítmica y sistemas o aplicaciones dedicadas a ello, cada cual más compleja que la anterior, la mayoría de los cuales suelen centrarse en un ámbito o estilo musical concreto.

III. 1 Sistemas dedicados a la composición algorítmica

Dada las facilidades que ofrecen los sistemas informáticos actuales, y las increíblemente amplias posibilidades del campo del presente estudio, existen infinidad de sistemas y aplicaciones destinadas a la composición algorítmica desarrolladas por técnicos, músicos, estudiosos o simplemente gente curiosa con interés por saber hasta dónde se podría llegar mediante un sistema de estas características.

Los distintos sistemas existentes suelen centrarse en ámbitos o campos musicales, o distintos métodos de obtención de resultados. Existen estudios que se centran, de hecho, en tratar de recopilar y clasificar todos los existentes (Fernández & Francisco, 2013), así que se nombrarán algunos considerados basados en distintos ámbitos u otros que, por sus características, pueden resultar, cuanto menos, curiosos. De esta forma, el lector podrá llegar a hacerse una idea de la infinidad de posibilidades que la composición algorítmica puede ofrecer, siendo algo que, aunque contrario a lo que pueda parecer en un primer momento por definición, puede ser un instrumento más en manos de la creatividad de aquellos que pretendan estudiarla.

Entre aquellos en los cuales los estudios estadísticos se encuentran entre las bases de su obtención de resultados, encontramos sistemas como el *Jazzomat*[1] (Abeßer, Frieler, Pfleiderer, & Zaddach, 2013) , que quizás no sea un sistema de composición algorítmica en sí, pero se centra en el estudio de exhaustivos análisis de solos de Jazz en orden de encontrar cierta correlación (figura III.1) y sentido en los fundamentos de la improvisación de este estilo musical para poder servir de base o otros sistemas mucho más simples junto con los cuales obtener composiciones interesantes estilísticamente.

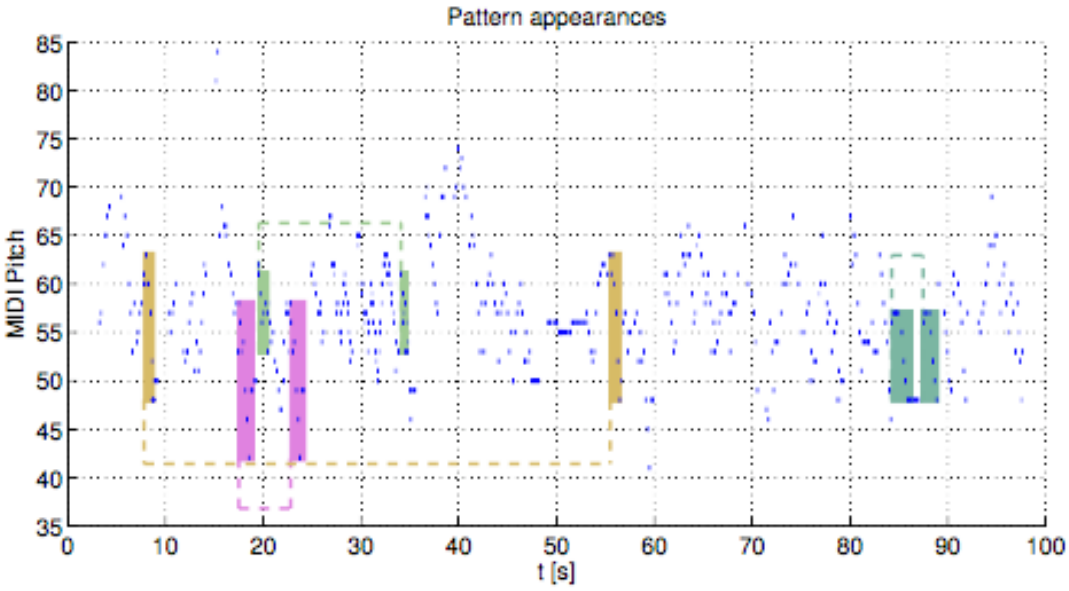

Figura **III.1**: Parejas de patrones de perfiles de altura MIDI iguales en el solo de Clifford Brown en "Joy Spring" obtenidas mediante Jazzomat.

Otros sistemas ya con salidas concretas en forma de secuencias musicales de carácter más general son encontrados fácilmente, algunos de los mejor considerados en cuanto al

[1] http://jazzomat.hfm-weimar.de/

nivel estilístico de la salida final son *Computoser*[2] (Bozhanov, 2014), siendo un modelo híbrido basado en amplias estadísticas y el uso de ciertas reglas musicales; o *SoundHelix*[3] que añade las funcionalidades que el formato MIDI ofrece, como secuenciadores o la posibilidad de extraer partituras en este formato o hacer sonar dispositivos MIDI. Ambos desarrollados en java.

Por otro lado, más allá de estudios técnicos, existen modelos de composición algorítmica orientados a la sociedad de la información en la que nos encontramos, que ofrecen aplicaciones online visualmente atractivas y resultados en función de parámetros concretos que no tienen por qué ser musicales, sino, siendo éstas más enfocadas a gente que desconoce aspectos concretos del ámbito musical, se sirve de estados de ánimo o momento del *clímax*, como *Jukedeck*[4], que es un servicio web de creación musical mediante IA a petición del usuario, o *mubert*[5] (figura III.2), entendido como una suerte de bot basada en una radio en la que el usuario tan sólo tiene que elegir un estilo (dentro del ámbito de la electrónica) y la web compone y reproduce en tiempo real secuencias sin final dentro de ese estilo.

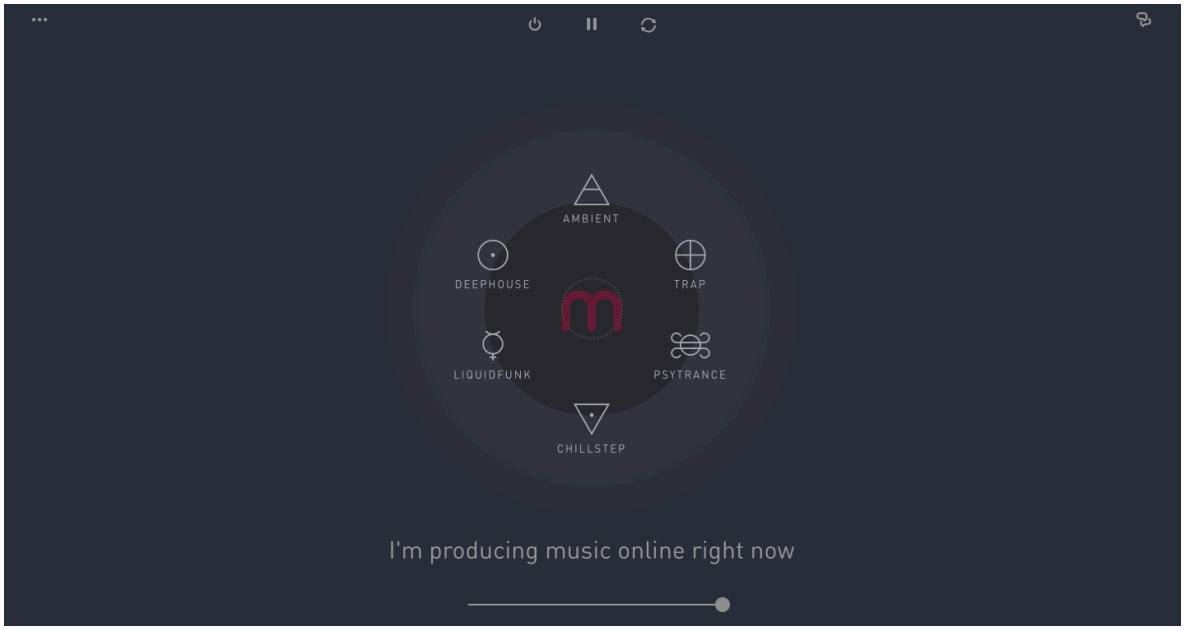

Figura **III. 2**: Interfaz de mubert con las distintas opciones de estilo seleccionables.

[2] http://computoser.com/
[3] https://www.soundhelix.com/
[4] https://www.jukedeck.com/
[5] http://mubert.com/es/

III. 2 Entornos de programación orientados

Conforme se fue haciendo más popular el estudio e intento de la composición algorítmica, fueron apareciendo entornos que facilitaban el desarrollo de aplicaciones destinados a ello.

Algunos de los más conocidos y usados son los siguientes:

III. 2. 1 Openmusic

OpenMusic es un entorno visual de programación orientada a objetos dedicado a la composición, el análisis y la investigación musical basado en Common Lisp[6]. Su objetivo principal no es la composición algorítmica en sí, pero sí la composición asistida por ordenador, y por tanto, mediante la creación de distintas herramientas podrá ser usado para este fin o como ayuda en conjunto con otro entornos comentados a continuación.

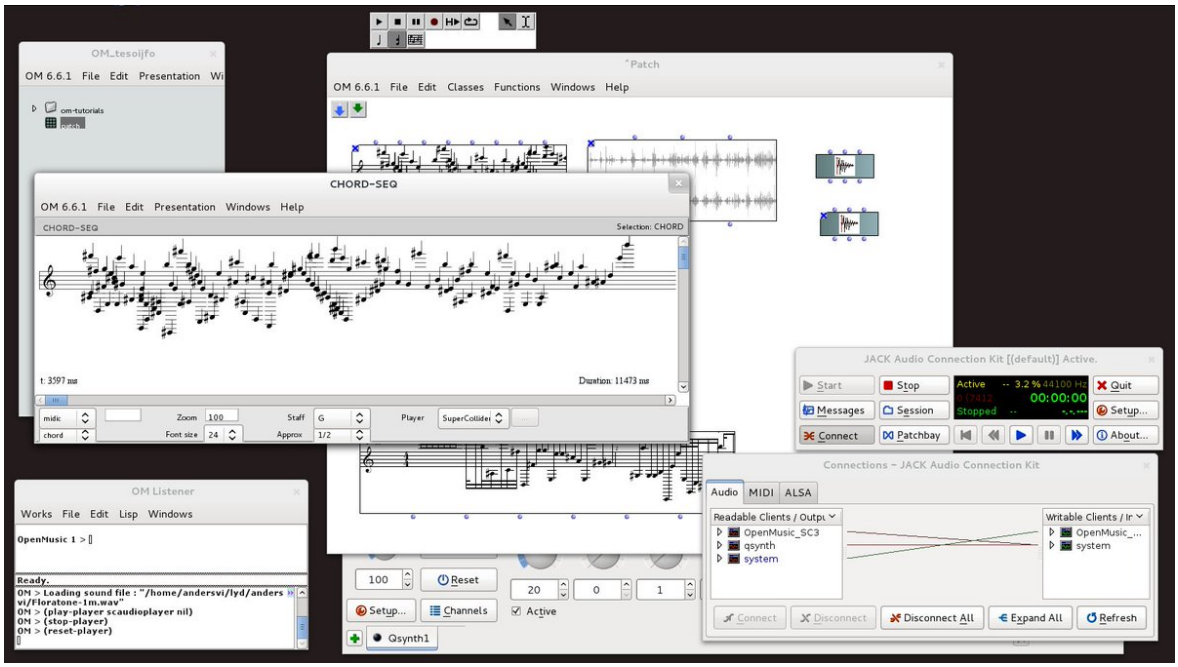

Figura **III. 3:** Ejemplo del entorno de programación OpenMusic.

III. 2. 2 Common Music

Common Music es otro entorno de programación orientada a objetos basado también en Lisp. Éste está orientado a la composición musical y permite la creación y manipulación de

[6] Dialecto o especificación del lenguaje de programación Lisp

información musical, que puede ser transformada en sonido, partituras u otras representaciones. Su principal aplicación a nivel usuario se conoce como Grace[7] y ofrece la posibilidad de emplear distintos métodos estocásticos para general valores musicales así como editores de código para los dialectos de lenguaje Sal y Scheme, conjuntos de instrumentos predefinidos y más funcionalidades orientadas a la composición algorítmica.

III. 2. 3 SuperCollider

SuperCollider es un entorno y lenguaje de programación para síntesis de audio en tiempo real y dedicado específicamente a la composición algorítmica. Éste combina la estructura orientada a objetos de Smalltalk[8] junto con la sintaxis del lenguaje C. Su funcionamiento se divide en dos componentes: un servidor, conocido como *scsynth*; y un cliente, *sclang*. Ambos se comunican mediante OSC[9] y en conjunto ofrecen variedad de tipos de síntesis de sonido, el uso de controladores MIDI y la posibilidad de comunicarse con otros lenguajes o aplicaciones.

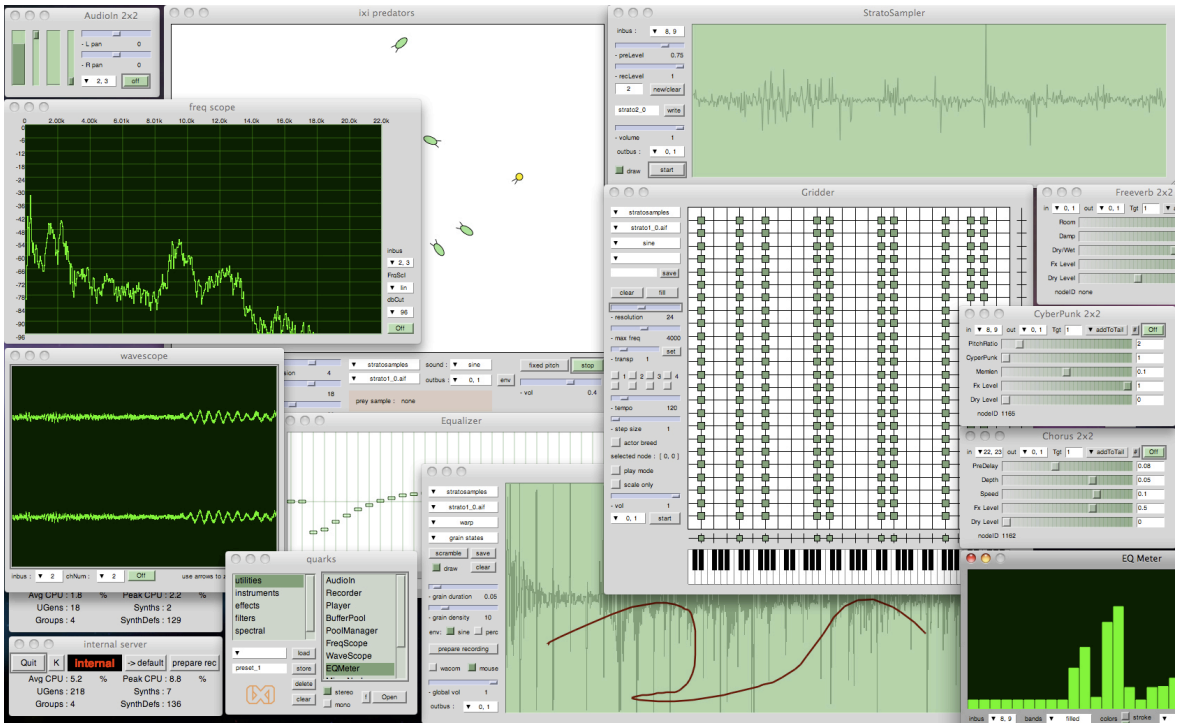

Figura **III. 4**: Ejemplo de las funcionalidades visuales del entorno de programación SuperCollider.

[7] Graphical Realtime Algorithmic Composition Environment (entorno gráfico de composición algorítmica en tiempo real)

[8] Lenguaje reflexivo de programación orientado a objetos con tipado dinámico.

[9] Open Sound Control, protocolo de comunicaciones que permite comunicar instrumentos de música, computadoras y otros instrumentos multimedia.

III. 2. 4 Csound

Csound es un lenguaje de programación, o más concretamente, un lenguaje de dominio específico dedicado al audio, escrito en C. Ofrece infinidad de herramientas de sínstesis de sonido y un flujo de funcionamiento particular que, por ser el entorno que se ha seleccionado para la realización del presente proyecto, se comentará con algo más de profundidad en el siguiente apartado.

III. 3 Herramientas usadas en el desarrollo

Como es objetivo de este proyecto, se ha considerado desde un primer momento utilizar para su desarrollo únicamente el entorno **Csound**. Por tanto, se procederá a comentar muy brevemente los principios básicos de funcionamiento de este entorno para que sea más fácilmente comprensible la explicación de su desarrollo y las distintas tomas de decisiones que se han llevado a cabo en la implementación.

Cabe mencionar que Csound proporciona una API que permite el control de este entorno mediante funciones de otros lenguajes como Java, C++ o Python. Como se ha comentado, se ha pretendido usar tan sólo el entorno Csound, pero, como se comentará en el capítulo VI, para posibles futuras implementaciones o ampliaciones de la aplicación fruto de estudio se tendrán en cuenta estas posibilidades.

Para el desarrollo en Csound se tienen en cuenta principalmente dos clases de objetos que se dividen en los dos grandes apartados de cada aplicación:

- Orquesta (orchestra): Donde se incluirán los instrumentos, que serán los encargados de realizar las operaciones diseñadas.
- Partitura (score): Ésta será una tabla en la que se especifica el orden de actuación de los instrumentos a lo largo del tiempo.

Los instrumentos serán creados en la orquesta, definidos en todos sus aspectos, y se les llamará desde la partitura. Sin embargo, los instrumentos podrán interactuar entre sí mediante el uso de variables globales o los llamados *eventos*, algo que será de mucha utilidad en el ámbito de la composición algorítmica o aleatoria.

Otro gran grupo que cabría destacar es el de los llamados *opcodes,* que son una suerte de funciones (de hecho, más adelante se hará referencia a ellos como tal), en un principio y

versiones anteriores del lenguaje propias del entorno, pero que en últimas versiones se da la opción de ser definidas por el usuario para realizar operaciones o funciones concretas con distintas entradas y salidas pudiendo ser llamadas en cualquier momento desde los instrumentos.

El flujo de ejecución de los códigos en Csound son algo a tener en cuenta. Éste es constante mientras no se indique lo contrario, por lo que los instrumentos que sean llamados o encendidos lo seguirán hasta que sean desactivados. Tras su ejecución, habrá una primera compilación llamada pase de inicialización, en la que las variables establecidas de tipo "i" serán inicializadas a los valores establecidos. Esta acción se producirá una vez en el instante inicial y no se repetirá, a no ser que se requiera mediante funciones específicas para ello, y posteriormente se ejecutarán los ciclos tipo "k" de forma constante y continua un número de veces definido por la frecuencia de muestreo (sr) y el número de muestras que se tomarán (ksmps) introducidos para la ejecución.

Por ejemplo:

$$Para\ un\ sr = 44100\ Hz\ y\ un\ ksmps = 4410$$

El número de ciclos por segundo vendrá definido por:

$$\frac{sr}{ksmps} = \frac{44100}{4410} = 10\ ciclos\ por\ segundo$$

Por tanto, cada el código de cada instrumento activado se ejecutará 10 veces por segundo de forma constante hasta que éste sea apagado.

III. 4 Análisis de requisitos iniciales

Como se ha comentado en los objetivos de la sección I.1, se pretende diseñar un sistema independiente basado en el lenguaje Csound de composición algorítmica híbrida estableciendo una serie probabilidades y reglas musicales lógicas para conseguir una composición musical coherente y estilísticamente atractiva.

Para ello se prevén una serie de requisitos que habrá que cumplir en función de lograr los objetivos impuestos.

- No salir del entorno Csound. Como se ha visto en capítulos anteriores, se propone la opción de implementar completamente la aplicación dentro del entorno Csound como

forma de estudio de las posibilidades que éste ofrece en cuanto a composición algorítmica.

- Lograr un sistema en el que el estudio estadístico no sea el elemento central sino que las probabilidades introducidas como base sean tan sólo una suerte de guías modificables en función de distintas reglas musicales lógicas que se definirán a lo largo de la fase diseño.

- Conseguir un sistema ampliable. Como se ha comentado, la teoría musical es tan amplia que sería imposible tratar de abarcar todas las posibilidades de composición existentes. Tan sólo introducir las declaraciones de acordes y escalas puede ser una tarea costosa dada no sólo la enorme cantidad de éstas, sino por sus distintas formas y tamaños. Por tanto, se propone diseñar una base de algoritmos y sucesiones de operaciones y eventos en los que se pueda trabajar con elementos musicales variables que podrían ser añadidos en futuras ampliaciones o extrapolaciones de la aplicación, así como nuevos algoritmos que puedan complementar a los ya existentes sin entorpecer su funcionamiento.

- Establecer las sucesiones de acordes como la base musical sobre la que establecer la composición y por tanto el resto de elementos que la formarán.

- Conseguir un sistema que pueda funcionar de forma independiente con la mínima intervención humana, pero ofrezca posibilidades de interacción a nivel usuario sencillas y sustanciales en la forma de obtener los resultados o en conceptos musicales simples como podrían ser la tonalidad, el modo de la secuencia musical o el tempo de la misma.

- Lograr una interacción en tiempo real con el sistema de la forma lo más dinámica posible dentro de las limitaciones del entorno.

- Y por último, lograr una salida musical coherente y estilísticamente interesante o agradable para el oyente como requisito principal sobre el que girará el diseño de los algoritmos a emplear para la obtención de resultados.

III. 5 Solución propuesta

Tras el análisis realizado y teniendo en cuenta los objetivos a conseguir, se propone un sistema de composición algorítmica basado en tres elementos:

- Acordes: Parte principal por ser a partir de la cual se generará el resto de elementos. Por ello, se pretende un desarrollo con una base pseudoaleatoria con restricciones bien definidas para lograr en todo momento unos fundamentos coherentes.
- Melodía: Elemento en el que se tratará de definir un desarrollo motívico más variado y más aleatorio dentro de lo estilísticamente agradable.
- Bajo: Éste será un elemento de apoyo para el resto que se incluirá para dar una sensación de completitud a la secuencia final. Por esto, junto con la naturaleza musical propia de las líneas de bajo dentro de lo que podrían ser consideradas "simples", la parte aleatoria del mismo será reducida.

Teniendo en cuenta, también, la enorme cantidad de posibilidades en cuanto a estilos musicales, se decidió partir del *jazz*, por ser éste un estilo lo suficientemente variado como para llegar a permitir numerosas posibilidades a la vez que se fundamenta en reglas concretas extrapolables muchas de ellas a términos aritméticos, y dada la importancia que ámbitos como la improvisación, a la que se puede asociar fácilmente la composición algorítmica, tienen en él.

III. 5. 1 Diseño preliminar

Dado el flujo de trabajo de Csound (apartado III.2), se pensó desde un primer momento definir dos grupos de instrumentos trabajando principalmente cada uno de ellos con los diferentes tipos de variable "i" y "k".

- Instrumentos de variables tipo "k" o *triggers:* Dada la naturaleza de los ciclos "k" de Csound, estas variables podrán ser calculadas varias veces por segundo en función del tiempo definido. Esto se consideró de utilidad para los cálculos de duraciones de notas ya que en sus representaciones más breves se necesitará la emisión de varias notas por segundo. Por tanto, en este grupo de instrumentos se implementarán aquellos que realizarán los cálculos de duraciones de cada una de los elementos antes definidos.
- Instrumentos de variables tipo "i" o instrumentos *calculadores*: Para obtener un programa bien definido y estructurado se consideró que la mejor forma de

implementación era la de separar en otro bloque de instrumentos a aquellos que realizarán los cálculos concretos para la obtención de las notas que finalmente sonarán.

- Instrumentos de salida: Y por último, se implementará un conjunto de instrumentos que serán los encargados de convertir en audio las variables anteriormente obtenidas.

En una primera instancia se consideró que la forma lógica y por tanto la adecuada de combinar las distintas secciones musicales con los distintos tipos de instrumentos era crear un instrumento de cada elemento para cada una de las dos secciones. Cada instrumento tipo *trigger* funcionaría de forma continua y activaría su instrumento respectivo tipo *calculador* en los momentos requeridos; y éstos, a su vez, activarían sus instrumentos de salida para lograr la conversión a audio. Por tanto, se obtendría una estructura de tres niveles de la forma que se muestra en la figura III.5.

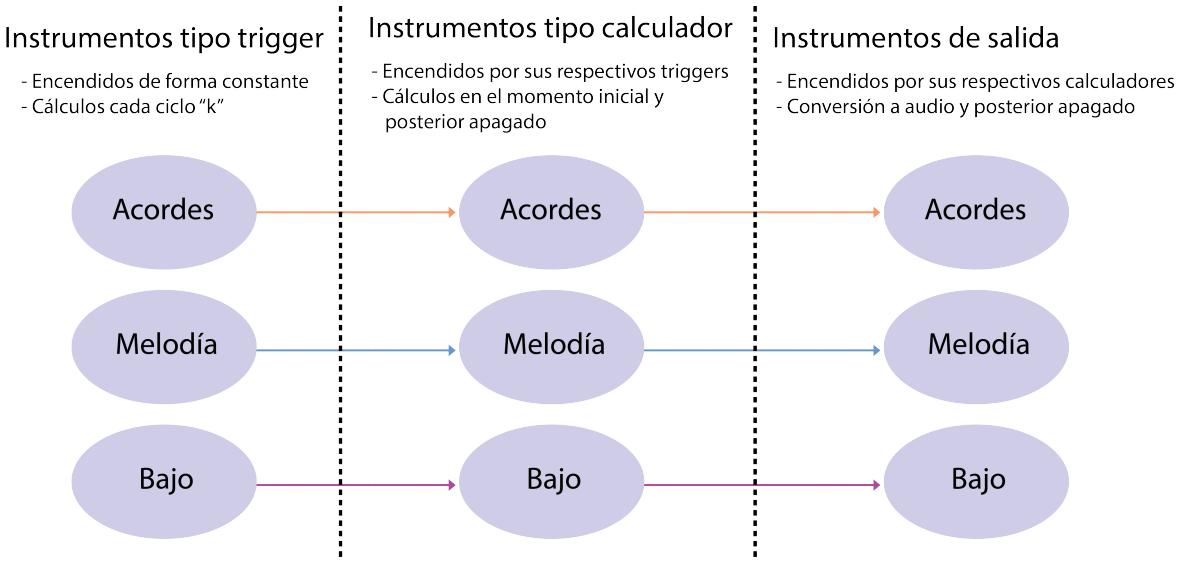

Figura **III. 5:** Diagrama de funcionamiento preliminar.

Este primer diseño llegó a plantearse en implementación demostrando ser insatisfactorio tras las primeras pruebas. La propia naturaleza de la aplicación a diseñar implica una dependencia indispensable entre los cálculos de las distintas secciones, por lo que variables calculadas, por ejemplo, en la sección de acordes, deben ser usadas al mismo tiempo en la sección de melodía. De un instrumento a otro se pueden compartir variables siempre y cuando éstas hayan sido declaradas globales, pero al ocurrir los cálculos en el mismo instante tiempo ocurrían fallos de sincronización que no se consiguieron solucionar.

Por ello, se llegó a la conclusión de que la forma de lograr una sincronización sin fallos sería la de incluir todos los cálculos de cada una de las secciones en un mismo instrumento y pasando los momentos de activación de uno a otro en forma de *flags*, siguiendo la estructura moestrada en ela figura III.6. Esto reduciría el número de instrumentos pasando a tener uno por sección en *trigger* y *calculador* a cambio de aumentar la complejidad de cada uno de ellos, pero aseguraría que en ningún caso se encontraran fallos de sincronización si se respeta la jerarquía de secciones dentro del propio instrumento.

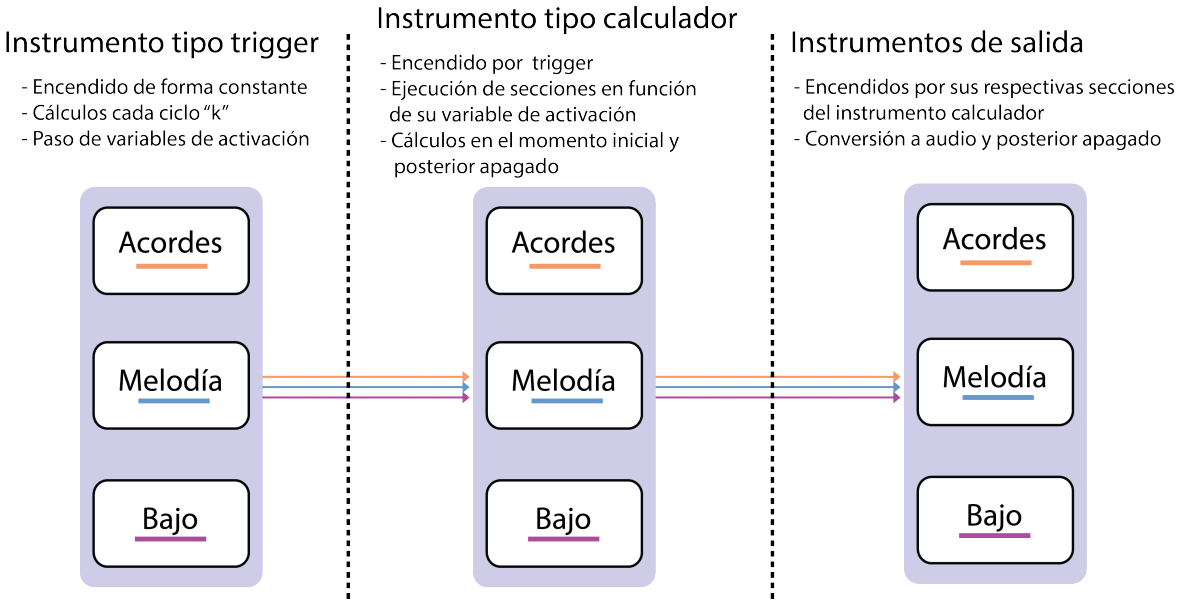

Figura **III. 6**: Diagrama de funcionamiento rediseñado.

Una vez aclarado el flujo de trabajo de la aplicación se pudo comenzar a desarrollar los algoritmos y las operaciones que formarán el esqueleto mediante el que se obtendrán las distintas notas que compondrán la secuencia musical. Cabe definir algunas secciones primordiales que serán detalladas en profundidad en el capítulo próximo.

Cálculo de duraciones musicales

Para cada uno de los elementos musicales se habrá de calcular su duración. Se pensó en definir una duración mínima que será la que establecerá el mínimo espacio de tiempo posible que pasará entre dos eventos consiguientes, y por tanto, el espacio de tiempo que sucederá entre las distintas activaciones del instrumento calculador.

El cálculo de duraciones se realizará por cadenas de *Márkov* como se explicó en el apartado II.2.1. Las duraciones de cada uno de los elementos se podrán realizar

de forma independiente y serán estas duraciones las que marcarán la activación de las siguientes notas en función de si la anterior ha terminado de sonar o no.

Cálculo de alturas musicales

Este quizás sea uno de los aspectos principales de la aplicación, ya que será el que defina le secuencia final. La forma de obtenerla será distinta para cada sección.

Acordes

Las secuencias de acordes se calcularán de acuerdo a la función tonal que realizará dicho acorde. Ésta se obtendrá mediante una cadena de *Márkov* de tres estados siendo cada uno de ellos función tónica, subdominante o dominante (apartado II.1.4). Una vez obtenida ésta, se procederá obtener el acorde aleatoriamente dentro de unas posibilidades definidas por la teoría musical. De este proceso se obtendrán variables que serán necesarias para el cálculo de alturas de la melodía.

Melodía

En un primer momento se barajó la opción de establecer la altura en base a una cadena de *Márkov* cuyos estados sean las distintas alturas establecidas por una escala. Esto fue rápidamente descartado por ser algo cerrado que dependería de la escala en todo momento y dadas los distintos tamaños de las mismas habría que definir distintas matrices de transiciones para cada una de ellas y contradeciría el requisito de que la aplicación se ampliable de forma sencilla.

La alternativa que se trabajó fue implementar una matriz de transición en función de intervalos o saltos[10]. Esto implicaba una complicación sustancial del diseño de algoritmos, pero a cambio ofrecería un mayor grado de independencia a la hora de introducir nuevas escalas u opciones, y, además, elevaría de forma exponencial el número de posibles salidas.

Para lograr esto, el modelo requerido se asemejaría más a un modelo oculto de *Márkov* (II.2.1) en el que el estado actual no sería conocido, ya que

[10] De ahora en adelante serán considerados como "saltos" dado que lo codificado no es estrictamente un tipo de intervalo sino la distancia en posiciones de una escala dada de un elemento al siguiente. Un salto de valor 2, por ejemplo, probablemente no tenga el mismo valor en intervalos de una escala a otra, y, ni siquiera, en una misma escala, empezando a contar el salto desde una posición u otra.

en el momento actual no se conocería de forma inmediata el número de saltos que ha dado lugar a esa salida. Podría no ser exacto del todo definir este proceso como un modelo oculto de *Márkov* en sentido estricto, ya que conociendo variables como el tipo de escala y la salida anterior (que no el estado), sería posible obtener el estado actual o salto que se ha realizado.

Este modelo a diseñar presenta también la complicación de los distintos tamaños de escalas y el calcular el salto a partir de una posición concreta que debe haber sido guardada de forma externa a la matriz de transición. Para el primer problema, se tomó en consideración el tamaño máximo de escalas a utilizar como 7, y por tanto los distintos estados posibles irán desde un salto de 0 posiciones hasta un salto de seis posiciones, que abarcaría todas las notas de la escala en el caso de encontrarse en la primera o la última. Para el segundo, se planteó la opción de diseñar una suerte de puntero que guarde la posición en la escala actual para ser utilizada en los cálculos posteriores.

Bajo

El bajo se planteará como un apoyo cuyas salidas se obtendrán a partir de aleatoriedad simple restringida en base a variables obtenidas en la sección de acordes y ciertas reglas musicales.

Todos estos aspectos serán modificables en función de reglas musicales establecidas que se irán definiendo conforme se vayan obteniendo resultados. Algunas que cabe implementar desde el principio serán límites en las alturas posibles, tanto por arriba como por debajo, para que no se obtengan sonidos desagradables en la salida y mayor probabilidad de permanecer en rangos centrales de frecuencia establecidos para cada sección.

Interfaz

El diseño de la interfaz se plantea como algo sencillo y visual dividida en bloques considerando cada una de las distintas secciones. Se pretende añadir distintas opciones musicales para el usuario como la selección de la tonalidad, el modo o el *tempo* de la secuencia musical, y la posibilidad de que éste modifique algunas variables de determinados algoritmos de forma que pueda formar parte del proceso de composición.

El diseño completo de ésta, por tanto, se llevará a cabo como añadido a la aplicación una vez se hayan concretado las distintas posibilidades que ésta ofrecerá.

IV

Diseño

En este apartado se procede a detallar el diseño final de la aplicación y el flujo de trabajo de la misma, así como el funcionamiento de cada una de sus partes de una forma visual desde la ejecución del código hasta el control de la secuencia creada en tiempo real por el usuario.

Para la implementación del sistema se ha tratado de tomar un enfoque técnico sin perder de vista el aspecto musical. Por ello, se han tomado ciertas consideraciones a tener en cuenta como tratar los términos temporales en función de figuras y tiempos musicales (explicados en la sección II.2.1), y referirse a los grados acordes o escalas en su nomenclatura musical, y de forma relativa ya que a este grado siempre se le deberá sumar el valor de la tonalidad correspondiente para obtener la nota en cuestión. En la figura IV.2[1] se muestran las etiquetas que por tanto harán referencia a los valores indicados.

Octave	Note Numbers											
	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
-1	0	1	2	3	4	5	6	7	8	9	10	11
0	12	13	14	15	16	17	18	19	20	21	22	23
1	24	25	26	27	28	29	30	31	32	33	34	35
2	36	37	38	39	40	41	42	43	44	45	46	47
3	48	49	50	51	52	53	54	55	56	57	58	59
4	60	61	62	63	64	65	66	67	68	69	70	71
5	72	73	74	75	76	77	78	79	80	81	82	83
6	84	85	86	87	88	89	90	91	92	93	94	95
7	96	97	98	99	100	101	102	103	104	105	106	107
8	108	109	110	111	112	113	114	115	116	117	118	119
9	120	121	122	123	124	125	126	127				

Figura **IV. 1**: Valores MIDI asignados a cada nota musical en sus respectivas octavas.

[1] Se emplea notación anglosajona donde C es Do, D es Re, E es Mi, F es Fa, G es Sol, A, es La y B es Si.

Se debe tener en cuenta también, que todo valor numérico relacionado con alturas de notas será expresado en su numeración correspondiente del formato MIDI por haber sido tratado así en la implementación de la aplicación. En la fase de salida del sistema, este valor se convertirá a hercios para hacer sonar la altura deseada. En la figura IV.1 se muestran las notas correspondientes de las distintas octavas a cada valor MIDI.

Grado	Valor Midi
I	0
IIb	1
II	2
IIIb	3
III	4
IV	5
Vb	6
V	7
VIb	8
VI	9
VIIb	10
VII	11

Figura **IV. 2:** Etiquetas asignadas a determinados valores MIDI.

IV. 1 Flujo del sistema y estructura

Dada la naturaleza del entorno Csound, la aplicación se ha diseñado no sólo para ejecutarse y hacer los cálculos correspondientes en tiempo real, sino también para responder a las posibles variaciones que se introduzcan desde la interfaz. Para ello, habrá un instrumento que estará en funcionamiento constante y marcará la activación de los instrumentos sucesivos, con sus distintos bloques, encargados de realizar los cálculos para obtener las notas en cuestión. A éste se le denominará *Trigger*. Todo instrumento diferente de este se apagará tras terminar los cálculos establecidos una vez se hayan encendido.

El instrumento *Trigger* junto con el instrumento llamado *Calculador,* formarán el esqueleto de la aplicación siendo los dos instrumentos en los que ocurrirán los eventos que marcarán las notas a sonar.

El flujo principal del sistema (figura IV.3)será secuencial a pesar de contar con eventos simultáneos y paralelos, y, sobre todo en el instrumento *Calculador,* prácticamente cada apartado dependerá del resultado obtenido en el apartado anterior. Las variables se sucederán de un instrumento a otro y entre las distintas secciones hasta llegar a las salidas de la aplicación en forma de frecuencia para la altura de cada una de las notas y tiempo para sus duraciones.

Cada vez que el tiempo global de ejecución sea igual al tiempo mínimo establecido entre la consecución de dos posibles eventos (como se explicará en el apartado IV.2.1), el instrumento *Trigger* activará el instrumento *Calculador,* que será el encargado de realizar todos los cálculos llamando a las funciones necesarias para obtener las notas de las distintas secciones de la secuencia musical: acordes, melodía y bajo. Cada una de estas secciones se activará, o no, en función de un parámetro activador en forma de *flag* que será proporcionado por el instrumento anterior. A su vez, si estas secciones se activan, encenderán los respectivos instrumentos que harán sonar las notas en cuestión.

Mientras sucede esto, otro bloque de instrumentos relativos a la interfaz, se activarán cuando el usuario active ciertas funcionalidades de ésta pudiendo introducir o cambiar parámetros empleados para los cálculos de cualquiera de los otros instrumentos.

Por tanto, la estructura de funcionamiento del sistema se dividirá en cuatro grandes bloques con distintas secciones:

- Instrumento *Trigger*.
    - Activador Acordes.
    - Activador Melodía.
    - Activador Bajo.
- Instrumento Calculador.
    - Calculador Acordes.
    - Calculador Melodía.
    - Calculador Bajo.

- Instrumentos de salida de notas.
    - Instrumentos Salida Acordes.

- Instrumento Salida Melodía.
- Instrumento Salida Bajo.

- Instrumentos de Interfaz.

Cabe mencionar que los instrumentos de salida de acordes serán cuatro, uno por cada nota. El motivo de esto se explicará en la sección dedicada a acordes del apartado IV.1.2.

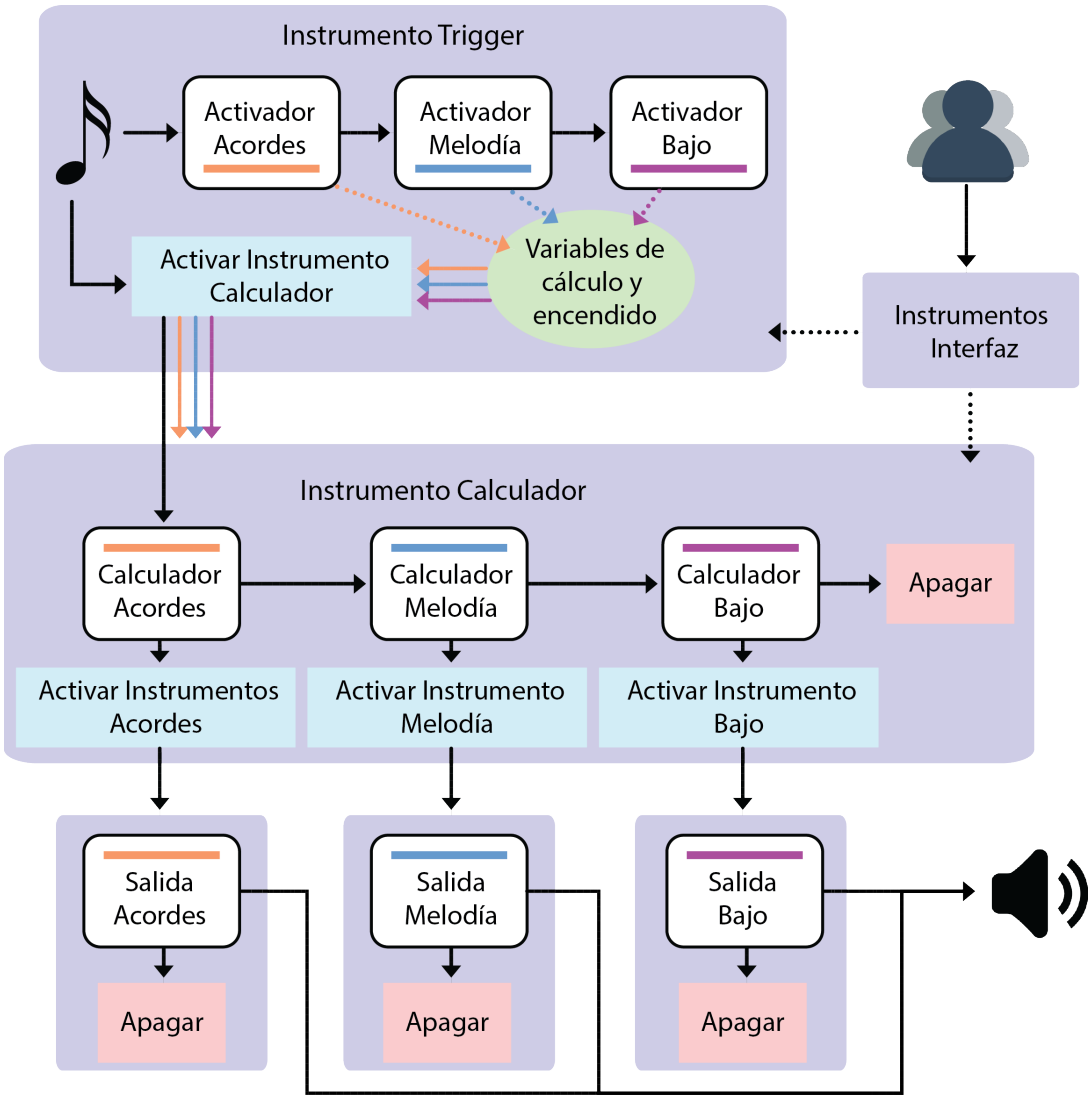

Figura IV. 3: Diagrama del flujo del sistema de la aplicación con los eventos principales.

En el fichero de implementación se encontrarán también otros dos grandes bloques: uno dedicado a la declaración e inicialización de variables y otro a la implementación de las diferentes funciones empleadas por los instrumentos. Éstos no se explicarán por ser algo perteneciente al ámbito concreto de la implementación, pero se hablará de algunos de ellos si se cree conveniente para la explicación de algún suceso de la ejecución.

IV. 1. 1 Instrumento *Trigger*

La principal función del instrumento *Trigger* será activar el instrumento que realizará los cálculos cada espacio de tiempo considerado como el mínimo posible entre un evento y el siguiente.

Para el funcionamiento de la aplicación se ha tenido en cuenta como división más pequeña del tiempo de un compás la semicorchea. Esto quiere decir que el espacio de tiempo relativo más pequeño que podrá haber entre un evento y el siguiente será 1/16 de la duración de cada compás. Esta duración vendrá dada por el parámetro *tempo* contabilizado en *bpms*, por tanto la duración absoluta de ese espacio de tiempo, o una semicorchea vendrá dada por la siguiente expresión:

$$Duración\ Semicorchea = \frac{60}{Tempo\ (bpm) \cdot 4}$$

Mediante la herramienta temporal "*metro*" de Csound, que devuelve un 1 cada vez que en el reloj interno de la aplicación pasa el tiempo introducido como parámetro[2], se consigue que el código diseñado para activar o no el instrumento que realiza los cálculos se ejecute en cada semicorchea de la secuencia. De ahora en adelante, por tanto, se hará referencia a lo relativo a tiempos de ejecución en términos musicales, ya que ha sido en estos términos como se ha llevado a cabo la implementación. Para la coordinación de las distintas secciones, tan sólo se ha requerido de una variable contador de semicorcheas que se incrementa cada vez que se activa la ejecución del código activador. A partir de ésta se obtendrán todas las variables temporales necesitadas a lo largo de la implementación como la posición actual en el compás o la duración de una secuencia introducida.

En cada una de estas ejecuciones se encuentran tres secciones principales que se corresponderán a la activación o no de cada uno de los bloques del instrumento que realiza los cálculos. En el gráfico IV.4 se muestra un ejemplo de las distintas activaciones que realiza el instrumento *Trigger* durante dos compases.

A continuación se comentan los principales eventos de cada una de las secciones del instrumento detallado.

[2] Este parámetro será proporcional al *Tempo* introducido

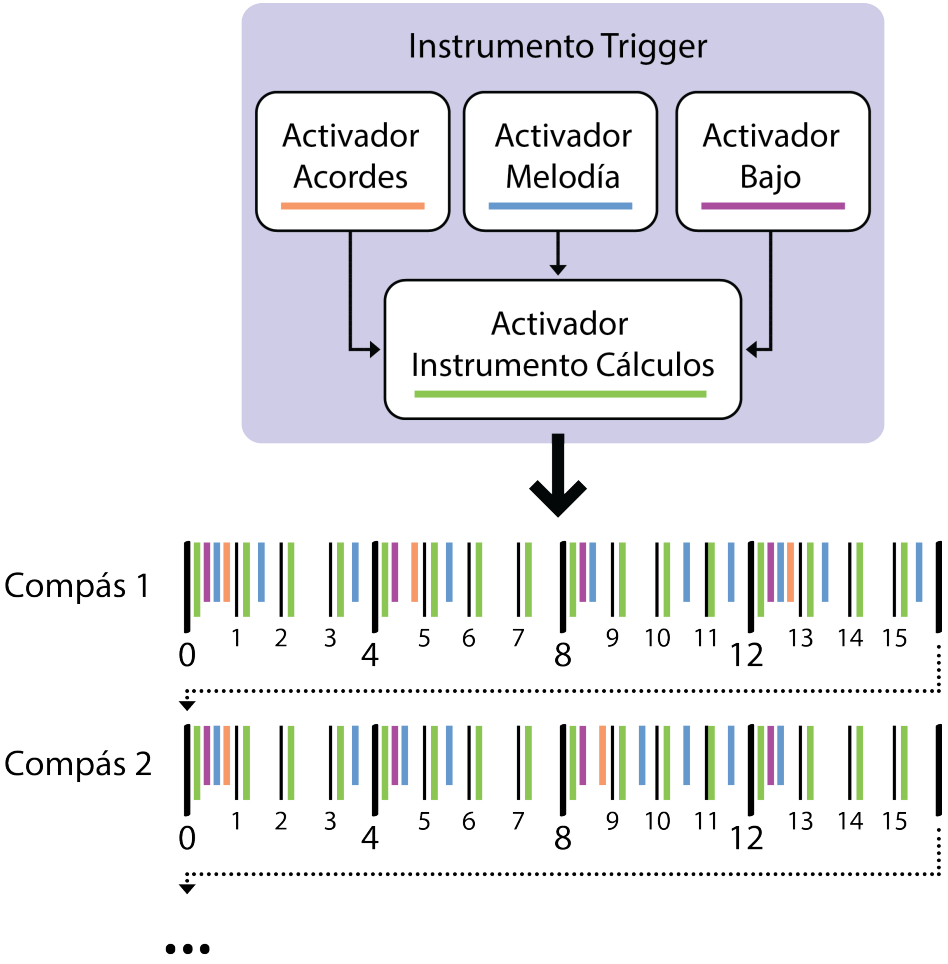

Figura **IV. 4**: Diagrama de funcionamiento del Instrumento *Trigger* con un ejemplo de las activaciones en cada instante semicorchea del instrumento y la sección pertinentes.

Activador de acordes

Para decidir si se activará o no el cálculo de acordes se tienen en cuenta dos variables: el número de semicorchea en el compás y la duración acumulada de los acordes en el compás. Si éstos son iguales, se ejecutará el código del bloque y se activará el *flag* para activar los cálculos en el siguiente instrumento.

Que se active un acorde o no, por tanto, depende de la duración del acorde anterior[3], es decir, un nuevo acorde se activará en el momento en que haya terminado de sonar el acorde anterior.

[3] El parámetro a tener en cuenta es la duración acumulada, por tanto en el instante inicial se activará de igual forma al ser una duración acumulada de 0 y encontrarse la ejecución en la semicorchea 0.

Si se va a activar un acorde en el instante actual, se procede a obtener su duración mediante una cadena de *Márkov* con tres estados posibles:

- Duración de redonda, o **4** tiempos.
- Duración de blanca, o **2** tiempos.
- Duración de negra, o **1** tiempo.

La tabla de probabilidades en que se codifica la cadena ha sido confeccionada mediante ensayo y error. Además, para facilitar la obtención de cierto sentido musical sin complicar demasiado los cálculos y posteriores secciones, se ha diseñado una función llamada RellenarDuracionesAcordes que modificará la tabla de probabilidades poniendo a cero aquellos valores que no permitan que suene un acorde en el primer tiempo de cada nuevo compás.

Conjuntamente con lo anterior explicado, se ejecutarán también las sentencias necesarias para guardar, leer o borrar la secuencia de duraciones de acordes en caso de ser requeridas. Ésta será una secuencia sencilla en forma de vector donde se guardará la duración de cada acorde para su posterior lectura secuencial.

Activador de melodía

De forma análoga al caso anterior, para decidir si se activará o no el cálculo de la nota de melodía, se tendrán en cuenta las variables de semicorchea en el compás y duración acumulada de melodía en el compás. Duración que será obtenida también mediante una cadena de *Márkov*, la diferencia reside tan sólo en las posibilidades de dicha duración.

La cadena de *Márkov* de la que se obtendrá la duración actual de la nota contará con cuatro posibles estados cuyas probabilidades, de nuevo, se han obtenido por ensayo y error:

- Duración de blanca, o **2** tiempos.
- Duración de negra, o **1** tiempo.
- Duración de corchea, o **½** de tiempo.
- Duración de semicorchea, o **¼** de tiempo.

Se ha implementado también una función RellenarDuracionesMelodía para modificar la tabla de probabilidades similar a la implementada para los acordes, con la salvedad de que en este caso no elimina ninguna posibilidad, sino que eleva

ligeramente las posibilidades de que suene una nota en un nuevo tiempo fuerte. Además, realiza las modificaciones necesarias para que una nueva secuencia empiece siempre con una nota en el primer tiempo de su primer compás.

Como añadido, se ha creado una función denominada NotaEnBeat, que devuelve la posición en la que coinciden, si coinciden, una nota de melodía con un acorde. La salida de esta función será usada más adelante como parámetro de entrada de NotaAcorde (función comentada en la sección de melodía del apartado IV.1.2)

De nuevo, se ejecutarán las secuencias necesarias para guardar, leer o borrar la secuencia de duraciones de notas de la melodía, que será también un vector donde se guardarán las duraciones de cada una de ellas.

Activador de bajo

El funcionamiento de este bloque es similar a los anteriores y la nota de bajo se activará en función de la duración anterior. En este caso la duración no se obtendrá mediante cadenas de *Markov* si no dependiendo del tipo de bajo que se quiera obtener. Las posibles duraciones, por tanto y como se detallará en la sección dedicada al cálculo de la nota de bajo, serán:

- Duración igual a duración de acorde.
- Duración de negra, o **1** tiempo.
- Duración de corchea, o **½** de tiempo.

En último lugar y tras la realización de todos los cálculos pertinentes, se ejecutará un evento mediante el que se activará el instrumento encargado de realizar los cálculos. Para que en este instrumento se realicen los cálculos adecuados de los elementos activados o no, se envían como parámetros un total de veintisiete variables que serán leídas en las distintas secciones en caso de ser necesario.

IV. 1. 2 Instrumento de cálculos

Este instrumento puede ser considerado el centro de la aplicación. En él se realizarán todos los cálculos y llamadas a funciones necesarias para obtener las notas que van a sonar de cada una de las secciones de la aplicación: acordes, melodía y bajo.

Su funcionamiento, como se ha comentado, será secuencial y se irán obteniendo variables mediante distintas funciones y cálculos hasta obtener las notas finales de cada una de ellas. Variables de la sección de acordes como grado y tipo de acorde serán necesarias para obtener la melodía y el bajo.

Cálculos de acordes

En este bloque del instrumento principal se realizan los cálculos y funciones necesarias para obtener las notas que sonarán en relación al acorde requerido. Es un proceso secuencial en el que se parte de tres parámetros: tonalidad y modo (que serán introducidos por el usuario como se verá en el apartado tal respecto a la interfaz), y la función tonal del acorde que se obtendrá de una tabla de probabilidades mediante una cadena de *Markov*.

El diagrama de bloques expuesto en la figura IV.5 se muestran los procesos que se llevan a cabo en el conjunto.

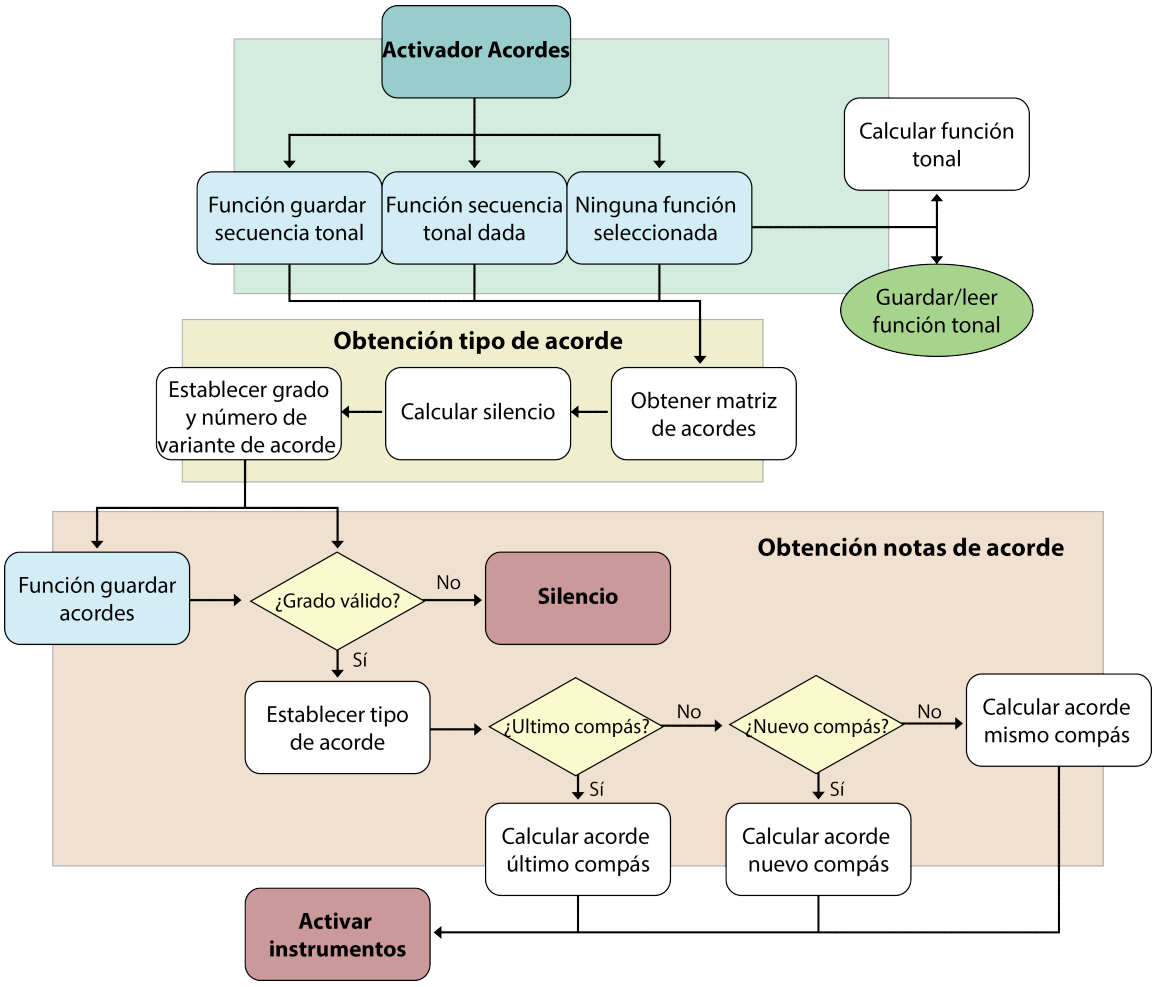

Figura **IV. 5:** Diagrama de bloques de la sección de acordes del Instrumento Calculador.

Los principales eventos se detallan a continuación:

Cálculo de la función tonal

Por lo explicado anteriormente en la sección II.1.4, se ha considerado la función tonal como el principal motivo variable de la elección de un acorde contando ya con la tonalidad y el modo de la secuencia musical. Partiendo de la base de la composición algorítmica, se ha creado una matriz de probabilidades para representar una cadena simple de *Márkov* de tres estados:

- Tónica (0)
- Subdominante (1)
- Dominante (2)

Mediante la correspondiente función, se obtendrá de ella la función tonal del presente acorde. A partir de esta función tonal y dependiendo del modo elegido, se obtendrá la matriz contenedora de los distintos acordes que pueden usarse en el caso que se trata.

Para no entrar en complicaciones de teoría musical, se ha simplificado el problema de cuándo elegir una nueva función tonal dentro de la secuencia musical y se ha decidido realizar este cálculo una vez a principio de cada compás. Por tanto, todos los acordes de un mismo compás realizarán la misma función tonal.

Como se verá en detalle en el apartado dedicado a la interfaz (sección IV.2), se ha implementado la funcionalidad de leer la función tonal de cada compás de una secuencia introducida por el usuario, en cuyo caso el flujo del bloque saltaría los cálculos y directamente obtendría la matriz de acordes mediante la función tonal leída.

Obtención del tipo de acorde

Una vez obtenida la matriz de acordes, se elegirá uno al azar de entre los posibles. Cada una de las matrices contiene los grados y los acordes relacionados con cada grado de acuerdo con la teoría vista en el apartado II.1. A cada tipo de acorde se le corresponde un número que coincide con la columna en la que se encuentra codificado en la matriz de acordes posibles.

Es aquí donde, en función de unas probabilidades obtenidas mediante prueba y error en diversas secuencias generadas, se establece si en lugar de sonar el acorde se producirá un silencio. Se decidió de esta forma para poder codificar el silencio en la matriz secuencia de acordes como grado -1.

Obtención de las notas del acorde

Las notas que van a sonar como acorde se obtienen en función de la situación del acorde mediante dos funciones distintas.

- Acorde de principio de compás: Para el primer acorde de cada compás se emplea una función llamada NotasAcordeNF de cinco entradas. Tres de ellas las variables a partir de las cuales se construye el acorde: tonalidad, grado del acorde y tipo de acorde; y dos de interfaz para modificar su funcionamiento. Ya que en la fase de diseño se estudiaron las distintas formas de selección de la inversión del acorde en función de distintos tipos de aleatoriedad y posibles parámetros, se decidió en la fase de implementación añadir varias de estas como posibles métodos a elegir por el usuario. Éstas serán de forma aleatoria, de forma aleatoria dentro de unos márgenes definidos por un valor relativo de altura introducido, de forma análoga al anterior pero introduciendo también el grado del acorde como variable a tener en cuenta, y de forma totalmente controlada tan sólo por la altura.

- Acorde de último compás: Para dar sensación de finalidad al oyente, se ha añadido un "último compás" que por definición realizará función de tónica en grado I de la tonalidad en la que se trabaje. Éste se obtendrá añadiendo estas variables a la función explicada en el caso anterior.

- Acorde en el mismo compás: Esta función, llamada NotasAcorde se basa en el vector de notas previas y el de posible acorde para buscar la inversión del acorde que más parecido sea al acorde tocado previamente.

Se ha buscado esta solución tratando de pensar en cómo se realizaría un cambio de acordes de la misma función en un piano. En ese caso, el pianista buscaría mover la mano lo menos posible a izquierda o derecha tratando de

encontrar la mínima variación entre un acorde y el siguiente, al ser esta variación un recurso estético más que un cambio sustancial en la secuencia musical. En la figura IV.6 se muestra un diagrama de los principales opcodes y un esquema sencillo de funcionamiento.

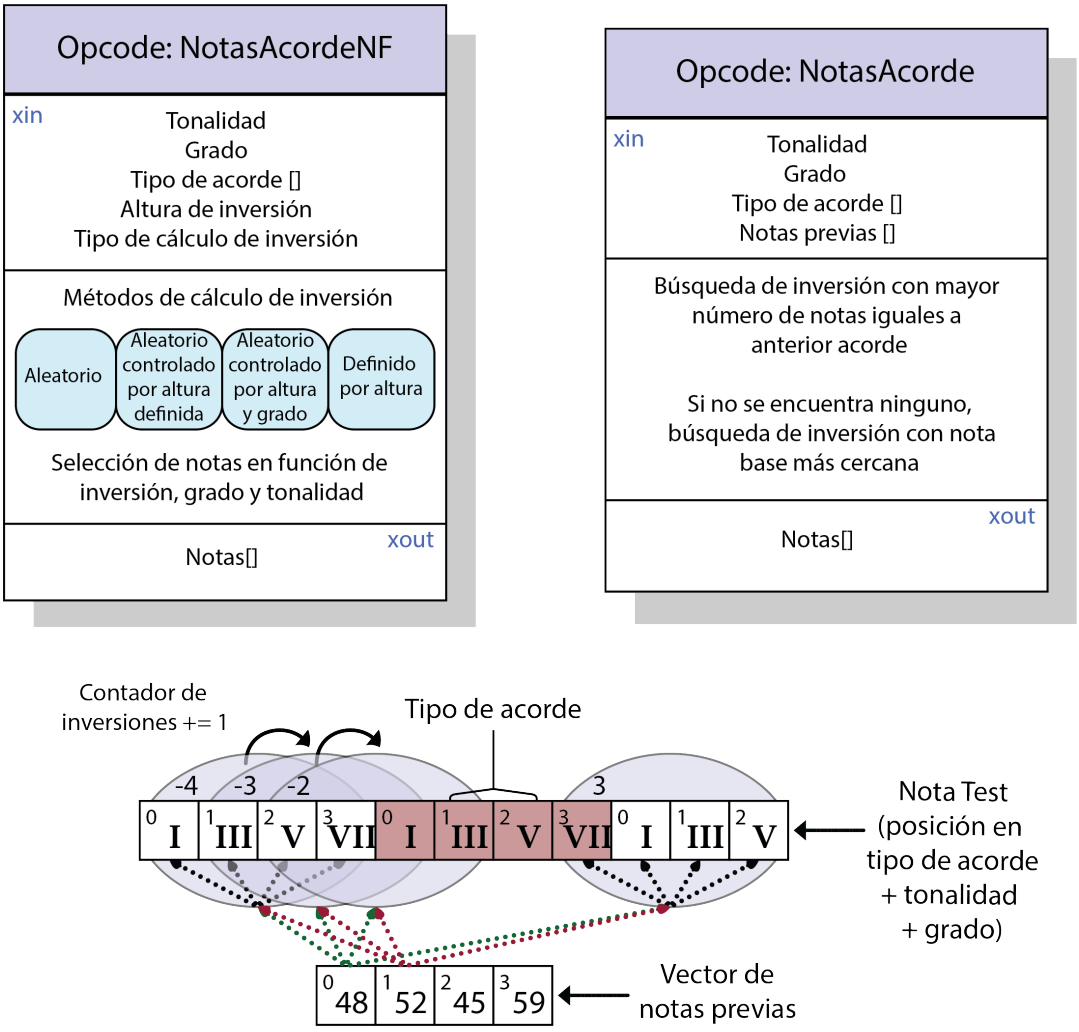

Figura IV. 6: Diagramas de entradas, salidas y principales eventos de las funciones *opcodes* NotasAcordeNF y NotasAcorde junto con un ejemplo de búsqueda de inversiones.

Activar instrumentos

Una vez obtenido el vector de notas, se produce un evento por cada una de ellas activando los cuatro correspondientes instrumentos. A cada uno se le pasan valores aleatorios de tiempo de retardo para tratar de "humanizar" la forma en que suena el acorde.

No se ha querido entrar en complicaciones en cuanto al sonido final en sí por no ser competencia de este proyecto, por ello se ha empleado un *opcode* propio de Csound llamado *prepiano* que simula un teclado con distintos

parámetros modificadores del sonido que se introducen en cada ocasión como valores aleatorios entre correspondientes máximos y mínimos hallados mediante prueba y error.

Funciones de secuencias, guardado y lectura

El conjunto está diseñada para funcionar de forma independiente realizando los cálculos pertinentes en tiempo real para cada acorde, pero se han implementado funcionalidades de guardado y lectura de secuencias para añadir grados de interacción que serán de utilidad al usuario.

El tamaño de la secuencia será establecido por el usuario, y una vez completada, la secuencia pasará de modo escritura a modo lectura. Para que esta funcionalidad sea completa y más atractiva a nivel usuario, se ha diseñado de forma dinámica en tiempo real, de modo que en cualquier momento de la secuencia el usuario pueda decidir empezar a guardar o "congelar" la secuencia del tamaño establecido que se repetirá hasta que éste desactive la función, y por tanto volverán a seleccionarse las variables de forma aleatoria. Se da la opción de guardar sólo funciones tonales o funciones tonales y tipos de acordes.

Las funciones tonales y los tipos de acordes se guardarán en una matriz cuya primera columna contendrá la función tonal y las siguientes los acordes de cada una de ellas (parte entera para el grado y decimal para el tipo de acorde) de la forma que se visualiza en la figura IV.7.

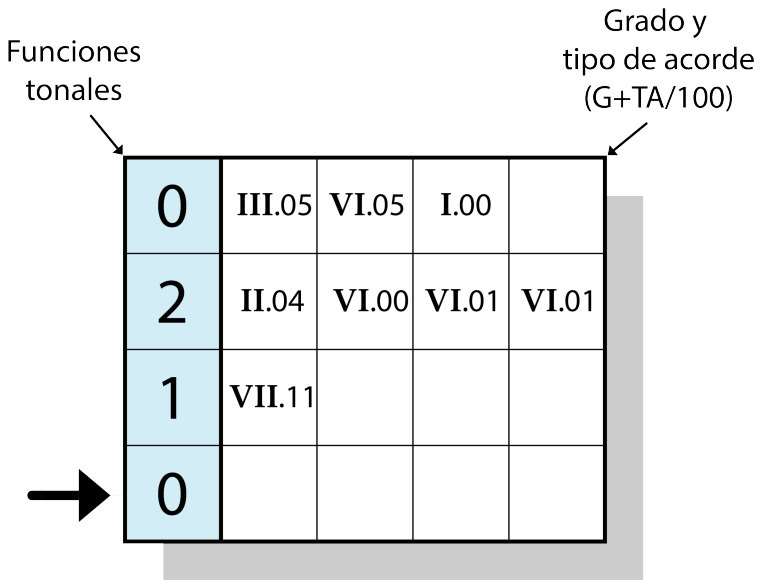

Figura **IV. 7:** Matriz de guardado de funciones tonales y tipos de acordes.

Cada nuevo compás se añade una fila de cinco columnas a la matriz en cuya primera columna se guarda la función tonal. En las columnas siguiente se guardarán tantos acordes como suenen en el compás. Para que los acordes suenen en la misma posición temporal se habrá requerido previamente de la matriz de duraciones en el instrumento disparador.

La función de introducir secuencia de funciones tonales se salta en cualquier caso el cálculo de éstas y las lee de la secuencia establecida en la interfaz. Mediante un sistema de *flags* que se detallará en el apartado relativo a la interfaz, las distintas opciones son intercambiables en cualquier momento de la reproducción.

Cálculos de melodía

Una vez obtenida la base de acordes ya sea mediante los cálculos pertinentes o su lectura, se tienen los parámetros necesarios para elegir la escala a usar, que será la variable necesaria sobre la que se creará la melodía. El proceso es igualmente secuencial y se irán obteniendo variables en función de alguna anterior hasta obtener la nota a sonar.

El diagrama de funcionamiento de este apartado se muestra en la figura IV.8. A continuación se detallan los principales eventos.

Obtención de la escala

La aplicación ofrece dos formas de obtener la escala a usar: una estática, en la que la escala será la misma para toda la secuencia musical dependiendo tan sólo de la tonalidad de la misma; y otra dinámica, en la que mediante funciones creadas para ello se obtienen las escalas posibles en cada momento dependiendo del grado y tipo del acorde y se selecciona la adecuada. Para esta selección de escala se ha creado una variable denominada Escala Principal que será la primera candidata en cada momento de elección y por tanto la que regirá en cierta medida la secuencia melódica. La Escala Principal podrá ser elegida por el usuario u obtenida de forma aleatoria uniforme entre las disponibles.

Para definir la escala, en primer lugar se realiza una búsqueda de las posibles escalas en los posibles grados para el acorde que suena mediante la función PosiblesEscalas. La salida de esta función será una matriz binaria

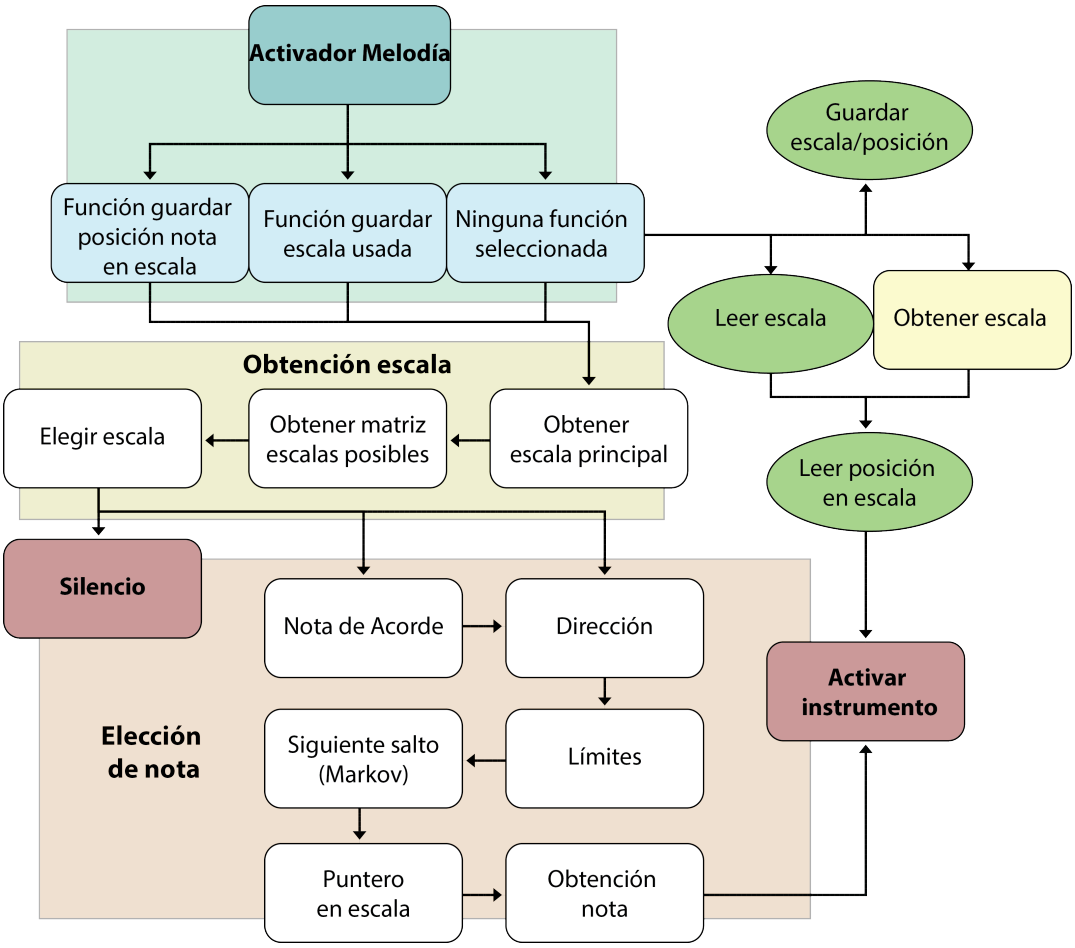

Figura **IV. 8**: Diagrama de bloques de la sección de melodía del Instrumento Calculador.

(posible o no posible para cada pareja grado – tipo de escala) que servirá de entrada a la función ElegirEscala, que elegirá una entre las posibles atendiendo a cuatro criterios ordenados jerárquicamente:

- Criterio 1: Se usará la escala principal en grado **I** si está entre las posibles.

- Criterio 2: Se usarán la escala y grado usados inmediatamente anterior si está entre las posibilidades.

- Criterio 3: Se busca una posible escala entre las posibles escalas en el grado actual. Si hay varias, se elegirá una de forma aleatoria dando siempre prioridad a la Escala Principal.

- Criterio 4: Si no hay ninguna posible escala entre las opciones anteriores, se realiza un barrido de grados por escala empezando por

la escala inmediatamente anterior hasta encontrar una escala con algún grado posible. Si hay varios, se elige uno al azar dando siempre prioridad al grado actual del acompañamiento. En la figura IV.9 se muestran de nuevo diagramas de los *opcodes* y funcionamiento.

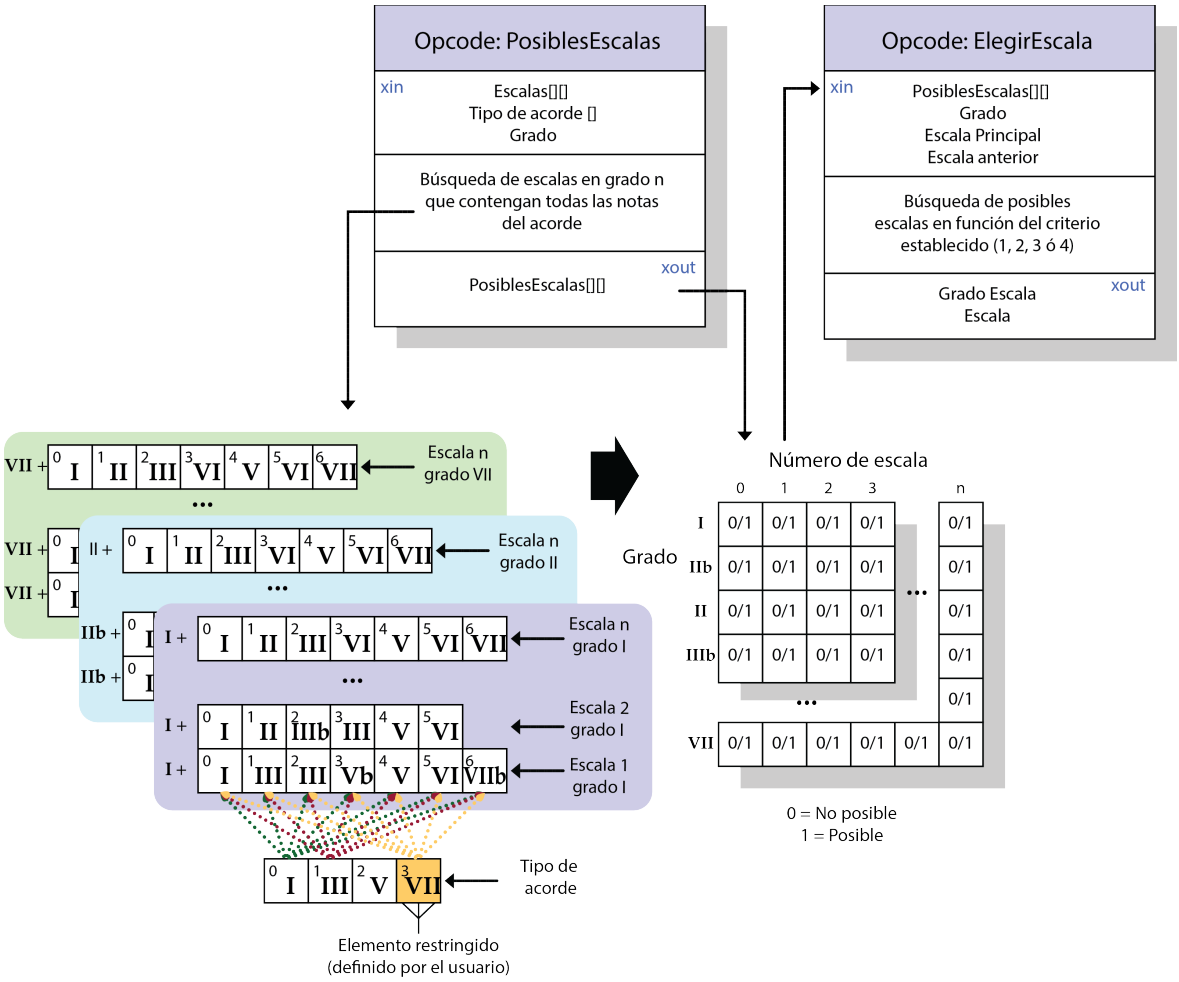

Figura IV. 9: Diagramas de las funciones PosiblesEscalas y ElegirEscala y el método de selección de escalas posibles.

Tras elegir la escala se decidirá si sonará una nota o se hará un silencio de forma análoga a en el apartado de acordes.

Elección de nota

La nota a sonar se obtiene mediante dos formas distintas en función de si se requiere una nota que esté contenida en el acorde o no. Para obtener una nota contenida en el acorde se usará la función NotaAcorde. Para calcular la nota se suceden una serie de funciones asociadas a los distintos eventos. Las

entradas, salidas, principales eventos y pequeño esquema del proceso en conjunto se pueden observar en la figura IV.10.

Se establecen dos fases para simplificar su explicación:

Primera Fase: Nota de acorde, dirección y límites.

Mediante las funciones NotaAcorde y Límites se modificará la tabla de probabilidades en función de los criterios establecidos. La primera, pondrá a 0 las probabilidades de todos aquellos saltos que no lleven a una nota del acorde (realizando la búsqueda en la octava actual, la anterior y la siguiente) que esté sonando en ese momento. Se dará prioridad a dirección ascendente en el salto, y si, en cada octava, no se encuentra ninguna nota que coincida con alguna nota del acorde, se buscará en dirección descendente.

Límites, realizará una operación similar, pero poniendo a 0 las probabilidades de saltos a notas que se encuentren fuera de los límites establecidos por un valor central establecido por el usuario. En caso de que en la dirección actual no se encuentre ningún salto posible, ésta se invertirá.

El uso de la función NotaAcorde lo decidirá el usuario mediante distintas opciones establecidas en la interfaz. Las funcionalidades de estas opciones serán:

- No usar la función, por tanto el acorde no influirá en la elección de nota de melodía en ningún momento.
- Usar la función si coinciden nota de melodía y acorde en primer tiempo de compás.
- Usar la función en cualquier tiempo que coincidan nota de melodía y acorde.
- Usar la función en todas las notas.

Dirección obtendrá de forma aleatoria si el salto a realizar será ascendente o descendente. La probabilidad será dinámica en función de la distancia al centro establecido por el usuario: a mayor distancia por debajo de la nota anterior, más probable será la dirección ascendente; y a mayor distancia por encima, más probable será la dirección descendente.

Tanto NotasAcorde como Límites tienen prioridad a la hora de establecer la dirección del salto e impondrán su criterio si en su

funcionamiento se estableció una dirección distinta a la obtenida de forma aleatoria.

Segunda fase: Siguiente salto, puntero en escala y obtención de nota

siguiente salto a realizar se obtendrá como se explica en el apartado tal siguiendo la cadena de *Márkov* con la tabla de probabilidades diseñada para el presente caso. Esta tabla podrá haber sido modificada por las funciones anteriores facilitando así que la nota elegida tenga cierto sentido musical.

La tabla de probabilidades diseñada se ha obtenido mediante prueba y error por los motivos explicados en el apartado de diseño, siendo ésta algo que podría variar en cierta medida o incluso, en futuras implementaciones, ofrecer la opción de ser modificada por el usuario. Al poder usar escalas de distintos tamaños, se ideó la tabla con el tamaño máximo posible de estas escalas, es decir, siete, para que las posibilidades vayan desde quedarse en la misma posición (en cualquier escala sería un intervalo de unísono, elemento 0) hasta realizar un salto de seis notas (con lo que se cubrirían todas las posibles posiciones de todas las escalas).

La función Puntero se diseñó para facilitar el proceso de realización de los saltos teniendo en cuenta posibles longitudes variables en las distintas escalas a usar. Ésta guarda la posición de la escala en la que se encuentra la nota actual y se mueve a la siguiente posición en función del salto y la dirección. Al existir la posibilidad de que los saltos sobrepasen la escala, se creó una variable llamada Contador Escala que identificará la altura global respecto a la inicial establecida. En el ejemplo mostrado en la figura IV.10 se observa cómo un salto descendente de cinco posiciones desde la posición uno de la escala (grado **II**) con el contador de escala a 0, lleva a la posición tres de la misma (grado **IV**) con el contador de escala a -1.

Para la obtención de la nota basta con realizar el cálculo numérico teniendo en cuenta las variables anteriores: posición del puntero, escala a emplear, grado de la melodía y contador de escala.

En último lugar, se procede a activar el instrumento que hará sonar la nota seleccionada. De forma análoga a en el caso de los acordes, se ha empleado el *opcode* de Csound *prepiano*.

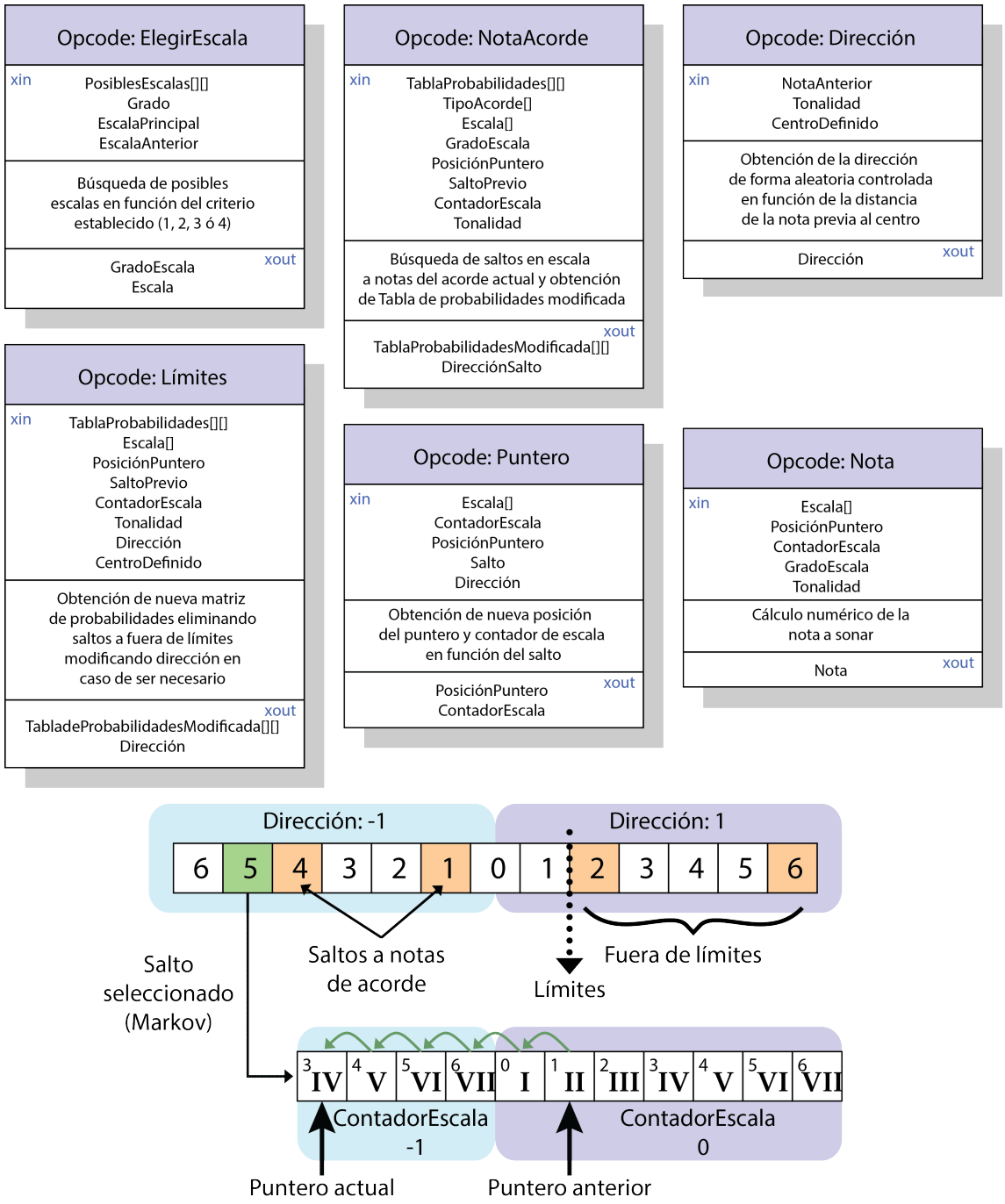

Figura **IV. 10**: Diagramas de las funciones asociadas a la elección de nota de melodía y esquema del proceso.

Funciones de secuencia, guardado y lectura

De igual forma que en el caso de los acordes, se ha añadido la función de guardar secuencias de melodía en forma de posiciones en una escala, y por otro lado las escalas usadas para dicha melodía. La melodía se guarda en un vector de posiciones donde cada nueva posición estará seguida con tantos ceros como dure la nota y los silencios se codifican con -1. A la vez que se

guardan las alturas, se guardan en otro vector las duraciones, y ambos serán activados a la vez en el momento requerido.

Al guardarse posiciones en una escala y no las notas concretas, se permite que la melodía se adecúe al acorde que esté sonando en ese momento mediante el cálculo de la escala correspondiente como se ha explicado anteriormente. En caso de querer mantener las alturas totales y que no sean relativas, se da la opción de guardar las escalas usadas para obtener dicha melodía. Esto puede dar lugar a disonancias ya que si la secuencia de acordes está siendo generada de forma aleatoria, la escala usada para guardar la melodía puede encontrarse con un acorde al que no se corresponde. A pesar de esto, se consideró que esta función podía ser de utilidad en muchos otros caso, como si se quieren guardar a la vez melodía y acordes o si se quieren probar distintas opciones de acompañamiento.

Cálculos de bajo

La línea de bajo se ha diseñado como un acompañamiento a partir del acorde que está sonando en cada momento. Por ello, el funcionamiento de esta sección de la aplicación es sencillo y consistirá tan sólo en seleccionar un algoritmo de cálculo de la nota del bajo u otro. Estos posibles algoritmos son cinco atendiendo a distintas formas teóricas de tocar una línea de bajo:

Una nota por acorde

Se tocará una nota de bajo siempre que suene un acorde y su altura será la tónica definida por el grado de dicho acorde (posición 0 del tipo de acorde).

Este algoritmo predominará en todos los casos siguientes, y se ejecutará en cualquiera de ellos siempre que se cumpla una de las siguientes condiciones:

- Suene la nota en el primer tiempo del compás.
- El acorde que suena involucre un cambio de grado respecto al acorde inmediatamente anterior.

Tónica más quinta

Sonará una nota en cada tiempo del compás (a negras*). Como se ha comentado, se elegirá la tónica en los casos anteriores, en cualquier otro, se elegirá de forma aleatoria si se tocará la tónica o la quinta de el acorde que esté sonando (posiciones 0 y 2 del tipo de acorde).

Nota aleatoria del acorde

Se tocará una nota en cada tiempo del compás. En los casos que no se deba tocar la tónica del acorde, se seleccionará una nota aleatoria de las cuatro posibles en el tipo de acorde.

Walking Bass

Se tocará también una nota en cada tiempo del compás En los casos que no se deba tocar la tónica del acorde, la siguiente nota del bajo será obtenida de forma similar a la siguiente nota de la melodía, salvo por que el salto entre notas será siempre de una posición de distancia dentro del vector posiciones del acorde. La probabilidad de que el salto sea ascendente o descendente variará en función de la nota anterior del bajo: cuanto más lejos por debajo del centro establecido por el usuario más posible será que el salto sea ascendente y cuanto más lejos por encima más posible será un salto descendente. El vector podrá ser excedido por ambos extremos gracias a una variable análoga a Contador Escala para la melodía.

Walking Bass a doble tempo:

En este caso, se procederá de forma exactamente igual al anterior, salvo que sonarán dos notas por cada tiempo del compás (a corcheas).

Por último, se procederá a activar el instrumento que hace sonar la nota de bajo de forma análoga a los apartados de melodía y acordes, con el *opcode prepiano*, salvo que para esta sección, ya que se trata de "emular" en la medida de lo posible a un bajo, se bajará la altura de la nota obtenida una octava.

IV. 2 Interfaz de usuario

Mediante esta interfaz se pretende hacer partícipe al usuario del proceso llevado a cabo por la aplicación de forma que éste tenga cierto control de algunos aspectos del mismo. La interfaz constará de dos ventanas entre las que se dividirán cuatro secciones relativas al funcionamiento general, la sección de acordes, la sección de melodía y la sección de bajo respectivamente. De esta forma, se pretende que el cambio de parámetros sea sencillo y accesible.

Por secciones, y sin entrar en mucho detalle, las funcionalidades de ésta serán:

Ejecución

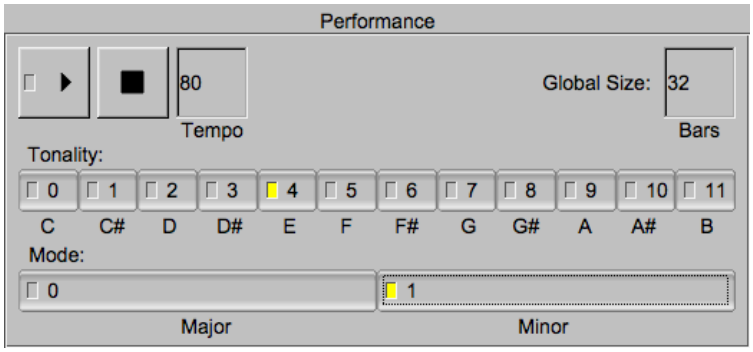

Figura **IV. 11**: Sección de ejecución o *performance* de la interfaz de usuario.

En esta sección mostrada en la figura IV.11, se agrupan, por un lado, los botones de control de la aplicación como play o stop, el tempo de la secuencia y el tamaño total en compases de la composición hasta que ésta termine de forma automática.

Por otro, se ofrece al usuario la opción de modificar en tiempo real tanto la tonalidad como el modo de la misma.

Acordes

En este apartado dedicado a los acordes (figura IV.12) se ofrecen por un lado las opciones de guardado comentadas en la sección referente a acordes del apartado IV.1.2. Éstas serán independientes y funcionales en cualquier momento de la

ejecución y se activarán en conjunto con el tamaño de secuencia seleccionado o la secuencia de funciones tonales introducida en caso de que ésta se active.

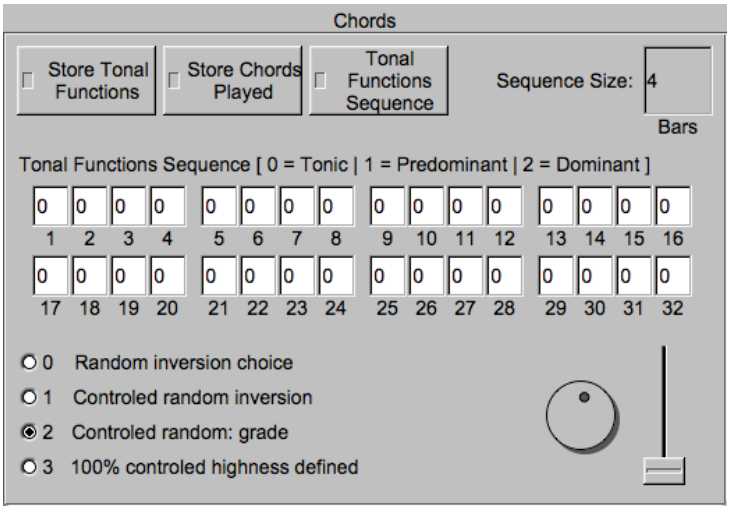

Figura **IV. 12:** Sección de acordes o *chords* de la interfaz de usuario.

Por otro lado (abajo a la izquierda en la figura IV.12), se permite seleccionar el distinto tipo de selección de inversión del acorde para los acordes de nuevo compás descritos en el apartado IV.1.2.

Y para terminar, de igual forma a la que se verá en el resto de secciones, se encontrará un *knob* de volumen y un slider mediante el que poder variar el parámetro de altura relativa de los acordes que se usará en los algoritmos de selección de inversiones.

Melodía

En este apartado (figura IV.13)se podrán seleccionar variaciones y funcionalidades de la creación de la melodía tales como los momentos de aplicación de la función NotaAcorde descrita en la sección dedicada a los cálculos de melodía del apartado IV.1.2, el criterio de selección de escala y si éste tendrá la restricción de la cuarta nota del acorde o no, y el tipo de escala para ser utilizada de la forma que se haya seleccionado.

En la parte superior se encuentran las funciones de guardado y tamaño de la secuencia, y en la inferior derecha el volumen y la altura relativa al igual que en la sección anterior.

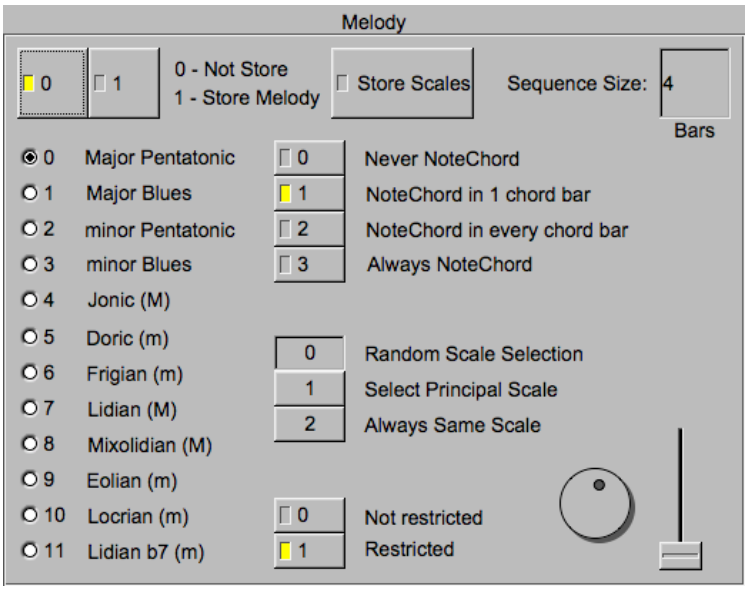

Figura **IV. 13:** Sección de melodía o *melody* de la interfaz de usuario.

Bajo

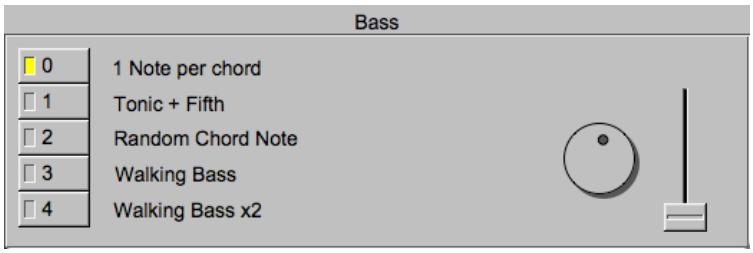

Figura **IV. 14:** Sección de bajo o *bass* de la interfaz de usuario.

En esta sección se ofrecerá la posibilidad de seleccionar el algoritmo para la elección de la nota de bajo y, como en el resto de secciones, volumen y altura relativa.

El conjunto global que forma la interfaz se muestra en la figura IV.15.

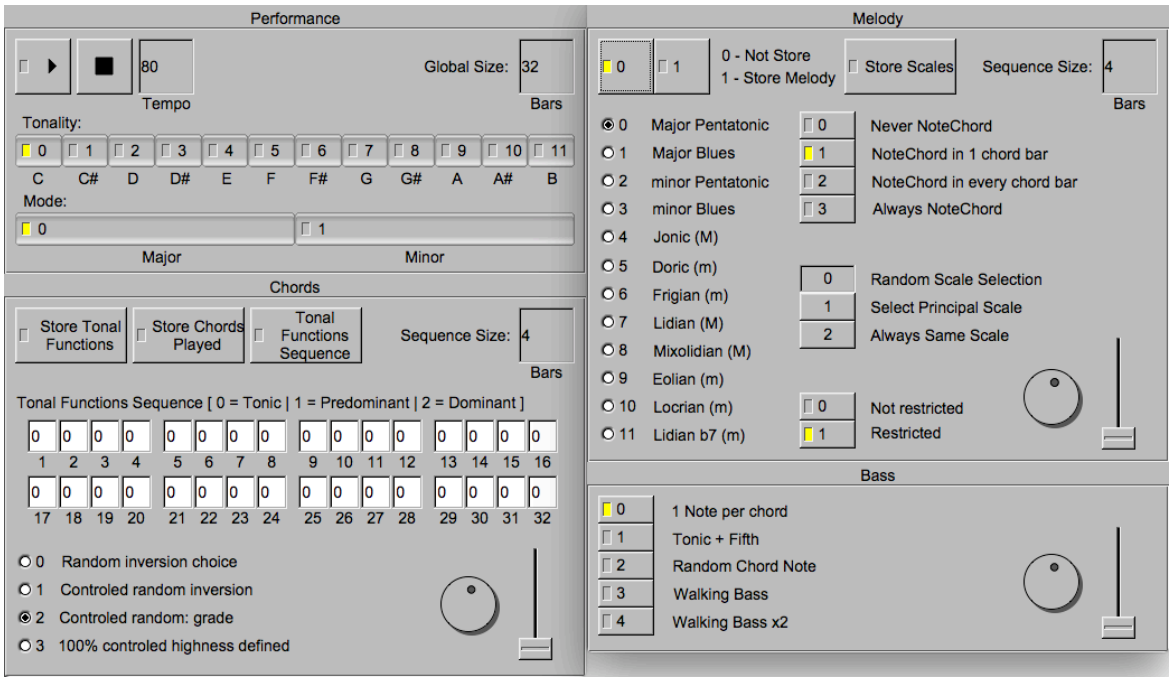

Figura **IV. 15**: Vista global de la interfaz de usuario.

V

Resultados

En este capítulo se procederá a comprobar y analizar algunas de las salidas de la aplicación. Dada la enorme cantidad de posibilidades y las distintas formas de obtener esas salidas, se ha decidido priorizar la demostración de lo que sería la coherencia musical por encima de las distintas opciones de cálculos de algoritmos implementadas, ya que esto ha sido extensamente expuesto en el capítulo IV, así como la comprobación del funcionamiento de dichos algoritmos. Por tanto, se transcribirán y comentarán a modo de ejemplo secuencias musicales cortas de cuatro compases en vistas a observar posibles errores o incongruencias que serán posteriormente analizadas. Estas secuencias podrán ser escuchadas en una página creada de Soundcloud[1] dedicada a la aplicación cuyos enlaces se añadirán a pie de página para cada una de ellas, así como muchas otras para su valoración particular por el oyente.

V. 1 Secuencia 1

Se analiza aquí una secuencia generada en tonalidad de La menor[2] de cuatro compases habiendo introducido una secuencia de funciones tonales para cada compás de tónica – subdominante – dominante – tónica (0 – 2 – 1 – 0) con las opciones de interfaz tal como se muestra en la figura V.1. Se profundizará en detalles técnicos musicales para ejemplificar la lógica de la aplicación, algo que no se repetirá en apartados posteriores por no incidir en información redundante.

[1] https://soundcloud.com/user-390085590

[2] https://soundcloud.com/user-390085590/am-4bar-seq1

Se han seleccionado como opciones a remarcar el hecho de usar el algoritmo de elegir la escala principal, en este caso elegida la escala de blues menor, y con la elección de escala restringida por las cuatro notas del tipo de acorde. El bajo será de la forma *walking bass* a negras[3]. La secuencia terminará en el cuarto compás como se ha seleccionado en el tamaño global de la misma. Las distintas alturas relativas se han establecido en una altura media – baja para disminuir las probabilidades de obtener unos resultados estridentes.

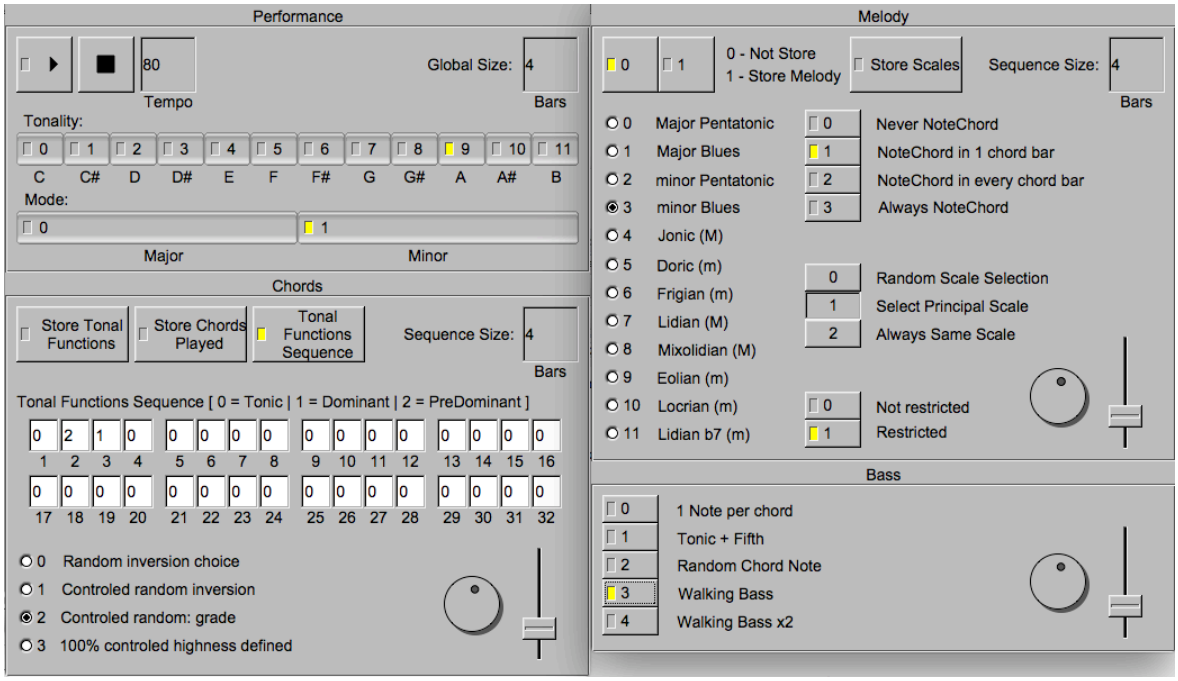

Figura **V. 1**: Estado de la interfaz para la generación de la secuencia 1 (La menor).

Con estas opciones, la transcripción de la secuencia obtenida sería la mostrada en la figura V.2.

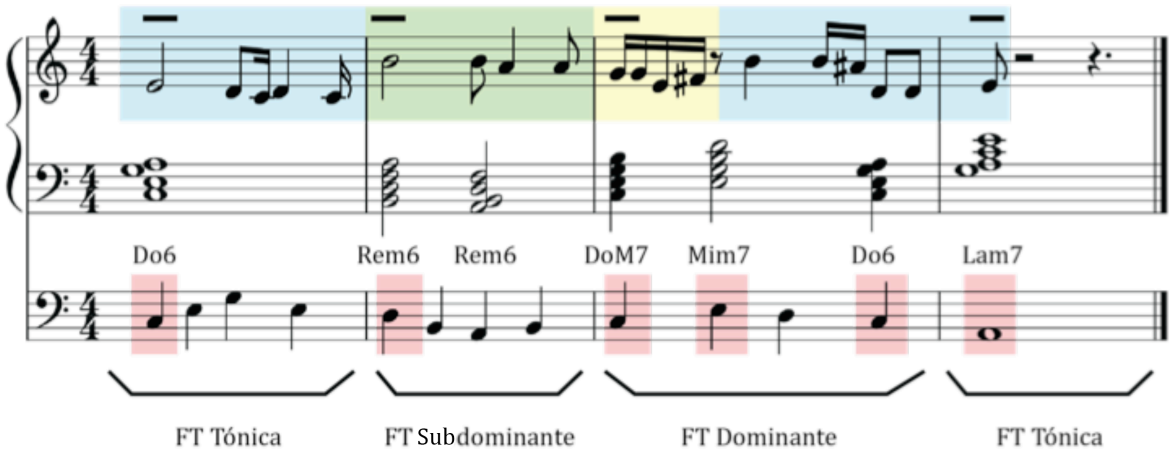

Figura **V. 2:** Transcripción a notación musical de la secuencia 1. Por orden descendente se muestran la partitura de la melodía, los acordes y el bajo.

[3] Cuatro notas por compás.

V. 1. 1 Acordes

La secuencia de acordes obtenida se muestra en la partitura intermedia de la figura V.2. En ésta, como se puede observar, la secuencia de funciones tonales es la introducida: tónica, predominante, dominante y tónica; correspondientes a cada uno de los cuatro compases descritos. De acuerdo con las reglas expuestas en la sección II.1.4, se comenta la lógica de los resultados obtenidos:

Primer compás

- Do6: En la tonalidad de La menor[4] el tercer grado será Do. Éste es un grado mayor, por tanto Do6 entra dentro de las posibilidades de acordes seleccionables, y, por definición según las reglas comentadas, es un acorde cuyas funciones posibles en tonalidades menores son de tónica en grado III y tónica y predominante en grado VI. Es remarcable que, aunque las posibilidades de que en función de tónica se obtenga un acorde en grado I para favorecer la sensación de tonalidad, ésta es una posibilidad igualmente válida según las reglas musicales.

Segundo compás

- Rem6: Re es el cuarto grado de La menor, en este caso un grado menor. Por tanto el acorde debería ser menor, como es el caso, y para realizar función de subdominante se considera los tipos "m6" y "m7". Los dos acordes del compás son el mismo, pero de acuerdo con el algoritmo y para darle cierto sentido a que se toque otro acorde, se busca una inversión distinta del mismo.

Tercer compás

- DoM7: Como se ha comentado, Do mayor es el tercer grado de La menor. Entre las posibilidades para realizar función dominante de este grado se encuentran los acordes tipo "6" y los "M7". La única diferencia entre ellos es la

[4] La menor natural, sin alteraciones, compuesta por las notas la, si, do, re, mi, fa y sol; intervalos T – S – T – T – S – T - T .

sexta o la séptima que son un La y un Si respectivamente, ambas incluidas en la tonalidad de La menor.

- Mim7: Mi será el quinto grado de la tonalidad de La menor, siendo éste un grado menor por definición. La única posibilidad de que este grado realice la función de dominante será mediante un acorde tipo "m7".
- Do6: Igual que en el caso del primer compás. Éste es un acorde versátil que puede también realizar la función de dominante.

Cuarto compás

- Lam7: Para terminar la secuencia siempre se buscará el primer grado de la tonalidad, en este caso La. Al ser una tonalidad menor, los acordes posibles serán los de tipo "m7".

V. 1. 2 Melodía

La melodía obtenida en la secuencia se muestra en la partitura superior de la figura V.2. En ésta se pueden observar en azul aquellas notas obtenidas mediante la escala principal (escala de blues menor) y otros colores cualquier otro tipo de escala. Las notas marcadas en la parte superior serán aquellas en las que se haya empleado la funcionalidad de NotaAcorde descrita en el apartado de cálculos de melodía de la sección IV.1.2. al coincidir[5] nota de melodía con acorde.

Primer compás

Para el acorde de Do6 la escala seleccionada ha sido la escala marcada como principal, la de blues menor en primer grado; es decir, en La. Cabe mencionar que la relativa mayor de La menor es Do, por tanto tiene sentido en un acorde mayor de Do tocar una escala menor de La.

Segundo compás

En el acorde de Rem6, por definición al ser un acorde menor de debería tocar una escala menor. Las notas de este acorde son Re, Fa, La y Si. Fa es la nota que impide

[5] Cabe remarcar que esta coincidencia no se busca, es un hecho aleatorio que puede ocurrir o no.

que se pueda usar la escala de blues de La menor por no encontrarse en ésta. Siguiendo el orden de prioridades del algoritmo, se ha buscado una escala para el cuarto grado de La menor, Re, y se ha obtenido la escala de blues mayor en este caso. En primera instancia puede resultar chocante que se toque una escala mayor en un grado menor, pero las escalas de blues son especiales en su formación y ésta tiene el tercer grado mayor, como todas las escalas mayores, y como nota de blues añadida el tercer grado menor, lo que da lugar a una escala compuesta por las notas Re, Mi, Fa (tercera menor, nota de blues), Fa#, La y Si, entre las que se encuentran todas las notas del acorde en cuestión.

Tercer compás

Para el acorde de DoM7 la escala seleccionada ha sido la lidia de Do. Ésta igual a la escala natural de do mayor salvo por una alteración en Fa (Do, Re, Mi, Fa#, Sol, La, Si). Luego es una escala que se puede tocarse sobre el acorde en cuestión y añadiría un matiz particular a la melodía al introducir una nota fuera de la tonalidad como es Fa#.

En el cambio a Mim7 se vuelve a buscar la escala menor de blues, en este caso la que se podría tocar según el acorde sería la propia de Mi. Esta escala contiene una alteración en tonalidad de La menor que sería precisamente La#.

Estas dos alteraciones no son algo común en la tonalidad de La menor, y quizás su uso vendría determinado por el propio intérprete, pero mientras no se usen en tiempos fuertes de forma que rompa la congruencia de la tonalidad pueden ser empleadas como notas de paso y matices.

Cuarto compás

En este caso, al volver a la tónica y caer en un acorde La menor, se tocará la escala principal en primer grado, es decir, escala menor de blues en La.

V. 1. 3 Bajo

En la figura V.2 se muestran en rojo las notas que son tónicas del acorde en el momento en que suenan a la vez. El resto serán el resto de notas del acorde que esté sonando

siempre a una distancia máxima de una posición. Por ejemplo, para el acorde de Rem6, se toca Re (tónica), Si (sexta), La (quinta) y Si de nuevo.

V. 2 Secuencia 2

En esta ocasión la secuencia generada será en la tonalidad de Re mayor[6] y formada por cuatro compases. Se ha usado la funcionalidad de guardar la secuencia melódica de tamaño un compás sin guardar las escalas, luego se en cada compás se deberían repetir las posiciones dentro de la escala del anterior. El bajo seleccionado ha sido el bajo de notas aleatorias dentro del acorde y las alturas relativas se han establecido en una altura media-alta. La escala principal se obtendrá de forma aleatoria y las inversiones serán controladas por el grado del acorde dentro de la aleatoriedad.

Las opciones de la interfaz que han dado lugar a esta secuencia se pueden observar en la figura V.3. No se hará esta vez un exhaustivo análisis musical, para un ejemplo de esto véase la sección V.1.

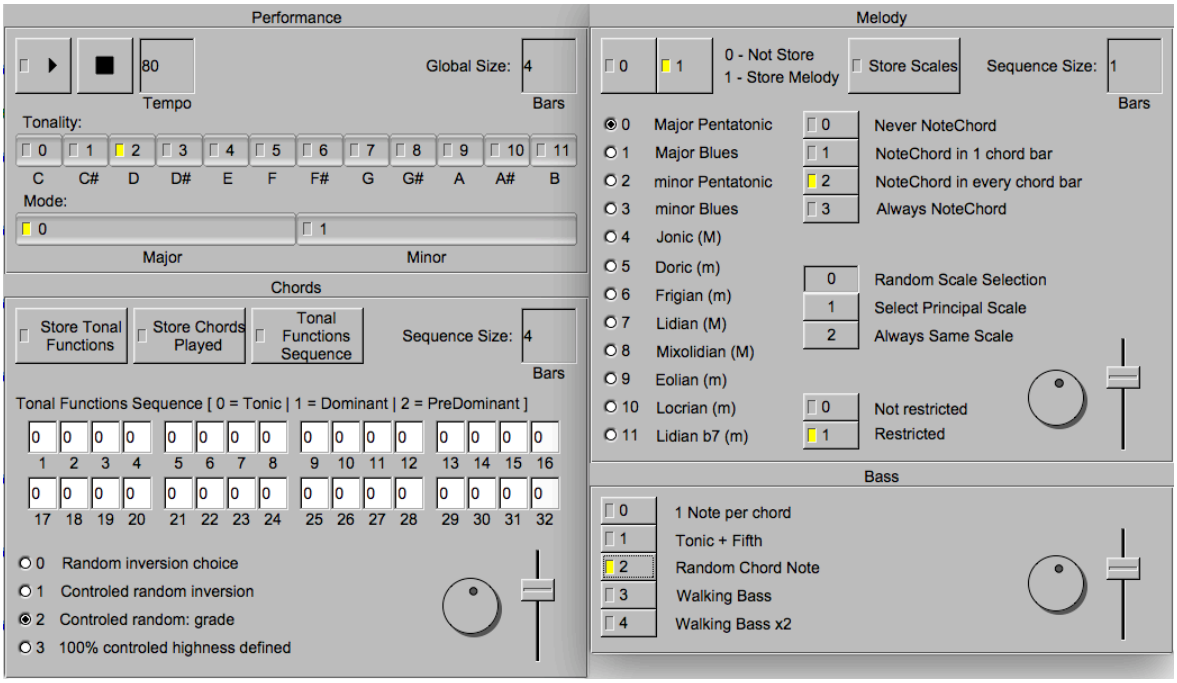

Figura **V. 3**: Estado de la interfaz para la generación de la secuencia 2 (Re mayor).

La transcripción a partitura de las distintas secciones se muestra en la figura V.4.

[6] https://soundcloud.com/user-390085590/dmaj4bar

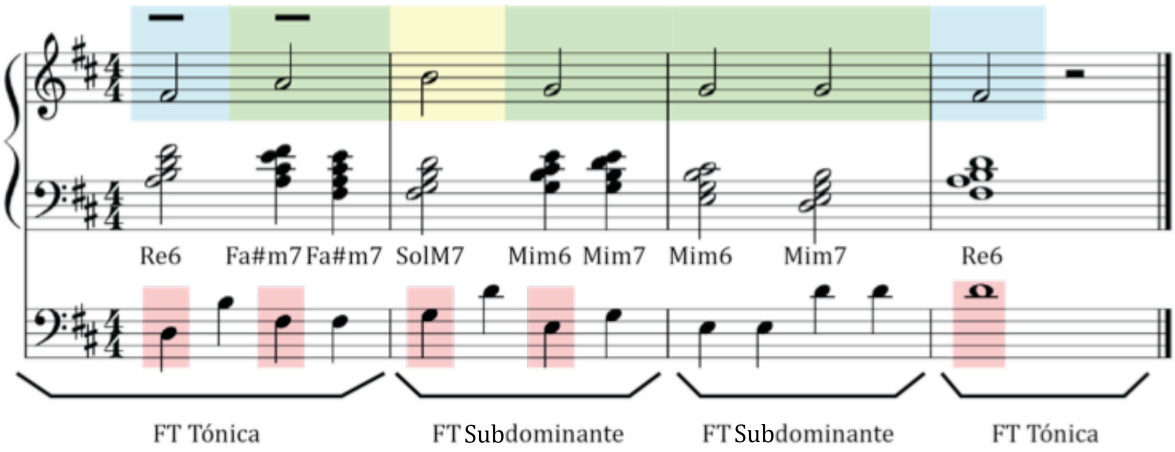

Figura **V. 4:** Transcripción a notación musical de la secuencia 2. Por orden descendente se muestran la partitura de la melodía, los acordes y el bajo.

V. 2. 1 Acordes

Las funciones tonales en esta secuencia se han obtenido de forma aleatoria dando lugar a una sucesión de tónica, subdominante, subdominante y tónica. Los acordes obtenidos están dentro de la lógica musical y dan lugar a una secuencia estilísticamente atractiva. Para la función tónica, el Re6, primer grado de la tonalidad de Re mayor, grado mayor, hace perfectamente función de tónica. Fa# es el tercer grado, en este caso menor, y el tipo de acorde "m7" también puede realizar esta función. Para los compases de función de predominante se obtiene un grado IV (Sol), grado mayor, acorde "M7" adecuado, y un grado ii (Mi), menor, con variaciones "m6" y "m7" ambas dentro de las posibilidades.

El acorde final, de nuevo, de función tónica, cierra el círculo introducido por la tonalidad de Re mayor con un acorde en primer grado.

V. 2. 2 Melodía

La melodía en este caso es menos variada dada la opción seleccionada de guardar melodía. Sólo se buscaron nuevas opciones dentro del primer compás, y, como cualquier otra posibilidad, se obtuvieron tan sólo dos notas cuya secuencia se repitió en los posteriores.

Las alturas de estas notas vendrán definidas por la escala a usar en cada momento ya que lo que se guarda de ellas es la posición dentro de la misma. Esto se detalla brevemente a continuación:

Primer compás

Ambas notas habrán sido obtenidas mediante la funcionalidad de NotaAcorde por haber coincidido con el lanzamiento de un nuevo acorde. Las escalas usadas serán la pentatónica mayor (azul en la figura V.4) de Re sobre el acorde Re6, dando lugar a un Fa#, posición número 3 de dicha escala; y la escala Dórica (verde) de Fa# sobre el acorde de Fa#, obteniendo un La, posición 3 de nuevo.

Segundo y tercer compás

Sobre el SolM7 se ha empleado una escala jónica de Sol en la que se ha buscado la posición 3 anteriormente obtenida. Si analizamos la escala jónica de Sol obtenemos la secuencia Sol, La, Si, Do, Re, Mi y Fa#; de donde se extrae la posición 3: Si.

De forma análoga, sobre el Mim7 se ha empleado la escala dórica de Mi, cuya secuencia se inicia como Mi, Fa#, Sol, La… De donde se extrae la posición 3: Sol.

En el tercer compás, la escala se mantendrá, por lo que las notas se repetirán.

Cuarto compás

En este caso se vuelve a la escala pentatónica mayor de Re sobre el acorde de Re6, y su tercera posición será la misma obtenida en el primer compás: Fa#.

V. 2. 3 Bajo

En este caso el bajo será obtenido de forma aleatoria dentro de las notas del compás. Aquellas que coincidan con un cambio de grado en el acorde serán tónicas, como en todos los casos, y el resto serán tercera, quinta o séptima. Por ejemplo, en la secuencia 2, en el momento que empiezan los acordes de Mim, el bajo realiza la secuencia: Mi (tónica), Sol (tercera), Mi, Mi, Re (séptima menor).

V. 3 Secuencia 3

Se incluye a modo de ejemplo un último caso cuyo nivel estilístico podría ser considerado menor por los distintos parámetros empleados para su generación[7]. Se trata de una secuencia de cuatro compases generada en Do mayor con secuencia de funciones tonales aleatoria. Se ha seleccionado la opción de no usar nunca la funcionalidad NotaAcorde, por lo que la nota que suene en el mismo instante de un acorde en cualquier caso podrá no formar parte de éste. Se ha seleccionado una escala principal, en este caso la Lidia b7, sin restricción, luego esta escala se usará independientemente de si la séptima del acorde esté incluida o no. El bajo será walking bass a doblre tempo, las inversiones estarán definidas tan sólo por la altura relativa que será baja en todas las secciones de la composición. Estas opciones se muestran en la figura V.5.

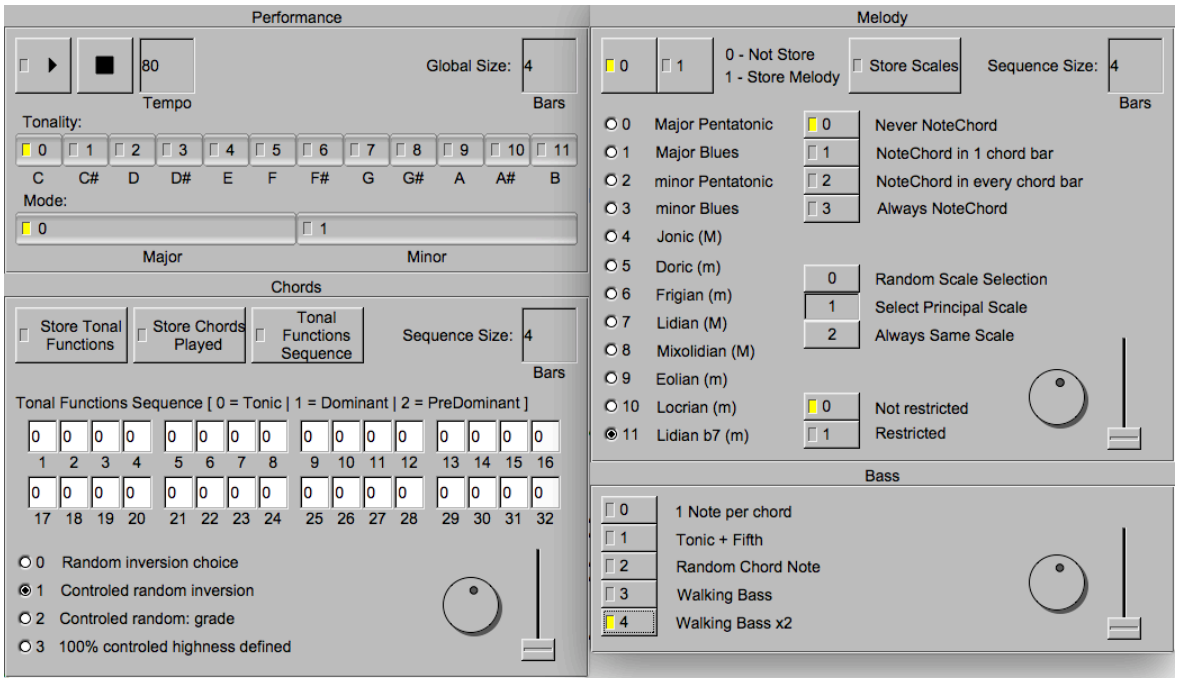

Figura V. 5: Estado de la interfaz para la generación de la secuencia 3 (Do mayor).

Con estas opciones, la transcripción quedaría de la forma que se muestra en la figura V.6.

Quedan patente distintos aspectos en la secuencia a estudio. En primer lugar, predomina de forma mucho más exagerada la seleccionada como escala principal (en azul, escala lidia b7) al no tener la restricción del séptimo grado del acorde. Esto puede resultar

[7] https://soundcloud.com/user-390085590/emaj4bar-3

algo positivo a nivel estilístico si se quiere dar un matiz concreto a la secuencia o algo negativo si se tiene en cuenta junto a otros aspectos como el hecho de no usar la funcionalidad de NotaAcorde nunca. En el tercer compás, por ejemplo, sobre un acorde de tónica DoM7 (Do, Mi, Sol, Si), se toca un Fa# que no se encuentra en la tonalidad y tampoco forma parte del acorde. Esto es algo que un intérprete trataría de evitar al menos en ese instante, pudiendo usar la nota como nota de paso pero no siendo algo estilísticamente adecuado en un tiempo fuerte con un acorde de tónica. Estas apreciaciones pueden considerarse algo subjetivo y las normas estrictamente musicales y objetivas en ello serían algo complicado y que podría llevar a debate.

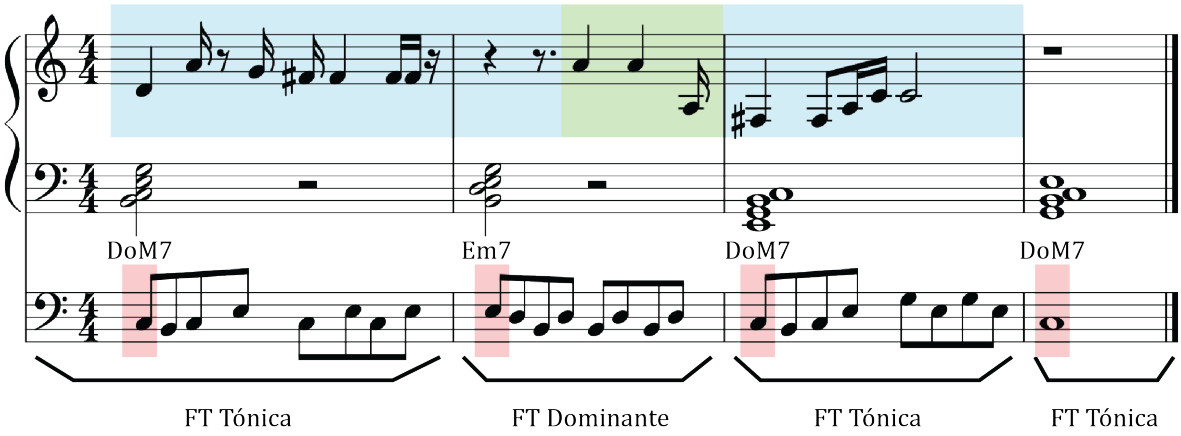

Figura **V. 6:** Transcripción a notación musical de la secuencia 3. Por orden descendente se muestran la partitura de la melodía, los acordes y el bajo.

El bajo, por su lado, se genera de la forma explicada en la sección dedicada al bajo de la sección IV.1.2, y en concreto al *walking bass* a doble tempo. Serán, como se puede observar, ocho notas por compás moviéndose desde la tónica en cada nuevo acorde, de posición en posición entre las notas de dicho acorde.

VI

Conclusiones

En este capítulo se expone una valoración en vista de los resultados obtenidos. Se tratará de justificar si se han cumplido los objetivos propuestos teniendo en cuenta los requisitos expuestos en el apartado III.4 o no, así como los problemas que se han encontrado en el desarrollo del proyecto. También se comentarán posibilidades de mejora, ya que el campo abarcado es un campo con numerosas posibilidades en el que tan sólo se han llegado a conocer y trabajar ciertos aspectos.

VI. 1 Evaluación del sistema

Recapitulando respecto a dichos requisitos anteriormente comentados se pueden concluir los siguientes aspectos:

- No salir del entorno Csound. A pesar de las dificultades encontradas y que se expondrán en un apartado posterior, se ha logrado desarrollar e implementar por completo la aplicación en CsoundQT. De esta forma se han estudiado las posibilidades de este entorno en cuanto a composición algorítmica y se puede concluir que las herramientas que ofrece y las características de ejecución de aplicaciones en este lenguaje facilitan y ayudan a desarrollar sistemas de composición algorítmica de forma que muchos otros no consiguen. Sin embargo, la implementación de sistemas complejos y extensos puede ser laborioso dado el escaso enfoque a la estructuración y facilidades sintácticas que ofrece el compilador CsoundQT, por lo que cabría

mencionar que el uso de la API de Csound desde otros lenguajes como Python o C++ sería una forma de trabajar más adecuada en casos en los que la complejidad del programa supere ciertos aspectos.

- Lograr un sistema en el que el estudio estadístico no sea el elemento central. Esto ha sido posible gracias a la adecuada implementación de algoritmos de forma que puedan modificar las probabilidades definidas por estadísticas. Lo que se ha conseguido es un sistema en el que dichas probabilidades puedan ser definidas de manera "lógica" para cualquiera con cierto sentido musical, y teniendo en cuenta, por supuesto, que éstas podrían ser redefinidas de forma más concreta mediante estudios estadísticos.

- Conseguir un sistema ampliable. En este apartado influyen negativamente las limitaciones comentadas de la herramienta CsoundQT. Aun así, los algoritmos y distintas secciones han sido definidas y separadas de forma que la inclusión de nuevos elementos no supongan complicadas modificaciones a nivel de implementación ni a nivel de lógica del programa, ya que los principales algoritmos están diseñados para trabajar con distintas variables y tamaños de las mismas. Por tanto, inclusiones y modificaciones relativamente menores serían algo completamente plausible y, de hecho, algo que se plantea en un futuro próximo. No así modificaciones grandes o en vistas a un posible punto de vista más atractivo a nivel comercial, para las que se debería trasladar el desarrollo a la API desde otro lenguaje de programación.

- Establecer las sucesiones de acordes como la base musical. Esto ha sido posible como se comenta en el capítulo IV, haciendo que los acordes sean el primer elemento generado a partir de una secuencia o un cálculo aleatorio de funciones tonales. Sin embargo, a lo largo del desarrollo, se barajó la posibilidad de que el usuario pudiera introducir directamente los acordes y no las secuencias tonales. Esto quizás favorecería que la aplicación fuera algo más interesante para personas con conocimientos musicales. Sin embargo, las opciones en cuanto a distintos acordes son tan amplias que las opciones de interfaz ofrecidas por CsoundQT no podrían ofrecer una forma sencilla de llegar al usuario convirtiendo esta tarea en algo sumamente complejo y, en parte, fuera del ámbito de estudio del presente proyecto. De nuevo, es algo que se baraja en futuras implementaciones desde otros entornos que permitan este acercamiento de forma más sencilla.

- Conseguir un sistema que pueda funcionar de forma independiente. Dada la opción de introducir las secuencias tonales o generarlas de forma aleatoria, esto es algo que ha sido posible en la implementación. La simple ejecución de la aplicación sin intervenir en la interfaz ya permite la generación musical aleatoria, si bien es cierto que la selección de distintas opciones favorecerá unos resultados u otros.

- Lograr una interacción en tiempo real. La interfaz no ha sido un elemento principal en la implementación de la aplicación y quizás las opciones ofrecidas no sean excesivamente extensas o sustanciales, sin embargo, éstas son variables en tiempo real como se pretendía y ofrecen un atractivo interesante a nivel usuario.

- Lograr una salida musical coherente y estilísticamente interesante. Este es un aspecto difícil de valorar dada la naturaleza subjetiva de la interpretación por parte del oyente de secuencias musicales. La coherencia de las mismas ha sido estudiada y demostrada en el capítulo V, en cuanto al estilismo de las mismas se podría decir que es algo que poseen la mayoría de ellas en cierta medida, y valoraciones más concretas deberían llevarse a cabo por oyentes de las mismas.

VI. 2 Posibilidades de mejora

Como se ha visto en el apartado anterior, los principales objetivos se han cumplido en su mayoría de forma satisfactoria. Sin embargo, en los distintos aspectos del desarrollo del proyecto se han ido viendo posibilidades más allá de lo propuesto inicialmente que podrían ayudar a crear una aplicación más completa e interesante.

- Implementación a partir de la API de Csound desde otro lenguaje. En este punto se ha incidido anteriormente, es una de las opciones más obvias en cuanto a mejoras posibles de la aplicación y por la que habría que pasar casi de manera obligada en el caso de querer convertir el presente proyecto en una aplicación comercial o con fines más allá de la investigación.

- Inclusión de bases de datos de secuencias musicales. Otra de las opciones que se barajó para obtener unos resultados estilísticamente positivos fue introducir bases de batos de secuencias, ya fueran de funciones tonales, de acordes, o incluso secuencias melódicas; para ser empleadas a petición del usuario o introducidas de forma

aleatoria en cualquier momento. Esto suponía dos problemas, para que fuera algo que aportara más posibilidades de las que quitaría, la base de datos debería ser lo suficientemente extensa como para ofrecer distintas opciones en cada momento; y por otro lado la inclusión de esto en la interfaz no resultaba sencillo desde CsoundQT. Sin embargo, dedicándole tiempo y estudio podría convertirse en otra función interesante de la aplicación.

- Aumento de las posibilidades en cuanto a acordes y escalas. Como se ha comentado, la teoría musical es enormemente extensa y tan sólo se han incluido un pequeño número de posibilidades. Éstas siempre serán ampliables en forma de nuevas escalas, nuevos acordes y tipos de acorde y la implementación de las reglas que los rigen.

- Inclusión de más secciones musicales. En concreto se plantea la posibilidad de incluir una base rítmica que terminara de cerrar el círculo de la composición.

- Inclusión de síntesis de sonido para una salida de audio más atractiva. Esto es algo que desde un primer momento se consideró fuera de ámbito del presente proyecto. Aun así, no deja de ser una opción que mejoraría el atractivo de la aplicación.

- Definición de distintos algoritmos en función de estilos musicales. Para este proyecto se ha partido en todo momento de un estilo musical más cercano al jazz, pero una vez vista la forma en la que esto se puede desarrollar ha sido inevitable plantearse si podría hacerse para diversos estilos musicales de una forma similar. Esto sería posible en conjunción con los apartados de posibles mejoras comentados anteriormente, convirtiendo así la aplicación en algo mucho más amplio.

Prácticamente todos los puntos mencionados deberían pasar por el primero. Trabajar únicamente en lenguaje Csound y desde CsoundQT puede ser problemático en aplicaciones con un nivel elevado de complejidad, algo que se ha comenzado a alcanzar tan sólo con las opciones ya implementadas, por lo que un aumento sustancial de las mismas debería pasar por algún sistema que permita al menos su estructuración y la división de tareas o mayores posibilidades a nivel interfaz.

VI. 3 Conclusiones del proyecto

Como se ha visto desde el primer capítulo de la presente memoria, la composición algorítmica es un campo enorme con inmensas posibilidades. La presente aplicación abarca tan sólo algunos ámbitos de este campo, logrando, aún así, un conjunto interesante y unos resultados satisfactorios.

El desarrollo de este proyecto ha supuesto una investigación constante tanto en el ámbito musical y de la composición algorítmica como en el de la programación en Csound. Esta investigación ha ofrecido muchas herramientas a la hora de favorecer el resultado de las distintas tomas de decisiones, sin embargo, algunas de estas decisiones han debido ser tomadas de manera subjetiva dada la propia naturaleza de los resultados que se pretendían obtener. El hecho de que uno de los principales objetivos del proyecto se focalice en interpretaciones de un supuesto oyente y no la obtención de unos resultados matemáticos o concretos (más allá de la lógica musical obtenida expuesta en el capítulo V), ha sido, por tanto, un arma de doble filo al ofrecer una libertad que quizás haya dado lugar a decisiones o resultados que puedan ser debatidas por entendidos de la teoría musical u oyentes con criterios particulares.

A pesar de esto, poniendo en práctica conceptos personales del autor como son la creatividad, el gusto por la música y conocimientos técnicos en distintos campos, se ha llegado a desarrollar la aplicación prácticamente desde cero, partiendo tan sólo de ideas relativamente simples como son las cadenas de *Márkov* y la teoría de funciones tonales, y un entorno de programación definido y con ciertas limitaciones. Dejando de lado soluciones, quizás más sencillas, dadas por opciones ya existentes o desarrolladas anteriormente.

Y esto, quizás, haya sido en definitiva el objetivo último de la realización del presente proyecto.

Bibliografía

[1] Raasted, J. (1979). *A neglected version of the anecdote about Pythagora's Hammer experiments.* Roskilde, Dinamarca: Cathiers de l'Institut du moyen-âge grec et latin.

[2] Diaz-Jerez, G. (2000). *Algorithmic music: Using mathematical models in music composition.* The Manhattan School of Music.

[3] Terefenko, D. (2014). *Jazz theory from based to advandec study.* Roudletge.

[4] Alpern, A. (1995). *Techniques for algorithmic composition of music.* Hampshire College.

[5] Maurer, J. A. (Marzo de 1999). *Center for Computer Research in Music and Acoustics.* Obtenido de A Brief History of Algorithmic Composition : *https://ccrma.stanford.edu/~blackrse/algorithm.html*

[6] Papadopouluos, G., & Wiggins, G. (1999). *AI Methods for Algorithmic Composition: A Survey, a Critical View and Future Prospects.* Edinburgh, Scotland: School of Artificial Intelligence, Division of Informatics, University of Edinburgh.

[7] Rincón, L. (2012). *Introducción a los procesos estocásticos.* México DF, México: Facultad de ciencias UNAM.

[8] De Roux, D. (7 de Septiembre de 2015). *Modelo oculto de Màrkov.* Obtenido de Quantil S.A.S.: *http://www.quantil.com.co/site/media/files/7_%20Cadenas%20de%20Markov%20Ocultas.pdf*

[9] Fernández, J. D., & Francisco, V. (2013). *AI Methods in Algorithmic Composition: A Comprehensive Survey* (Vol. 48). Universidad de Málaga, Málaga, España: Journal of Artificial Intelligence Research.

[10] Abeßer, J., Frieler, K., Pfleiderer, M., & Zaddach, W.-G. (2013). *Introducing the Jazzomat project - Jazz solo analysis using Music Information Retrieval methods .* Weimar, Germany: The Liszt School of Music.

[11] Bozhanov, B. (2014). *Computoser - rule-based, probability-driven algorithmic music composition .* (No publicado).