ISSN 1870-4069

COMPOSAIN: Aplicación de un algoritmo genético para el diseño de contrapuntos musicales sin restricción de integridad de figuras

Natan Vilchis-Tavera, Adriana Lara

Instituto Politécnico Nacional,
Escuela Superior de Física y Matemáticas,
México

nvilchist1300@alumno.ipn.mx,
alaral@ipn.mx

Resumen. Durante las últimas décadas los algoritmos evolutivos han generado con éxito soluciones más allá de la intuición humana; es por esto que resulta interesante aplicarlos en ámbitos donde la creatividad y genialidad juegan un papel importante. En este trabajo se presenta la aplicación de un algoritmo genético operando sobre melodías musicales. El problema particular que se aborda es el de generar un contrapunto musical adecuado para una cierta melodía existente. A diferencia de otros trabajos disponibles en la literatura, nuestra propuesta maneja una línea melódica diversa, es decir que permite codificar las notas con libertad en la duración de los sonidos y silencios. Abordamos aquí la tarea de la creación de frases musicales como un problema de optimización, y extrayendo características basadas en teoría musical diseñamos criterios matemáticos que funcionan como guía para el algoritmo genético. Nuestra propuesta fue probada en cuatro melodías, arrojando conclusiones interesantes y rutas prometedoras para futura investigación. Consideramos que este tipo de enfoque puede ser útil como ayuda para la composición musical tradicional asistida por un algoritmo inteligente. Las melodías resultantes de este trabajo pueden consultarse en los enlaces dados en el texto.

Palabras clave: Contrapunto musical, algoritmo genético, composición automática, optimización, composición musical.

COMPOSAIN: A genetic Algorithm Application for the Design of Musical Counterpoints without Figures Length Limitation

Abstract. Over the past decades, evolutionary algorithms have successfully generated solutions beyond human intuition; This is why it is interesting to apply them in areas where creativity and genius play an essential role. In this work, the application of a genetic algorithm operating on musical melodies is presented.

The tackled problem here is the design of a suitable musical it counterpoint for a particular existing melody. Unlike other previous works, our proposal handles a diverse melodic line. It allows the notes to be freely encoded regarding the length of the sounds and silences. We approach the task of creating musical phrases as an optimization problem. By extracting characteristics based on musical theory, we design mathematical criteria that guide the genetic algorithm. Our proposal was tested on four melodies, yielding interesting conclusions and promising routes for future research. We believe that this type of approach can be helpful as an aid to traditional music composition assisted by an intelligent algorithm. The melodies resulting from this work are available by the links given in the text.

Keywords: Musical counterpoint, genetic algorithm, automatic composition, optimization, musical composition.

1. Introducción

Los algoritmos genéticos son una herramienta poderosa que ha tenido éxito en diversas áreas del quehacer humano. En particular, en la generación automática de melodías, podemos encontrar sistemas de improvisación de jazz como GenJam [2, 14] o los sistemas propuesto por Dragan [12] y Garay [6], que generan melodías musicales a manera de primera voz. Estos trabajos han abierto exitosamente la discusión sobre si la creatividad musical es exclusiva del ser humano.

El presente trabajo propone dar al usuario un sistema que tiene como base un algoritmo genético para encontrar la mejor melodía que actúe como segunda voz para una melodía dada llamada *cantus firmus* [3] usando las reglas de contrapunto; es decir, COMPOSAIN permite encontrar la mejor segunda voz de contrapunto tal que se escuche agradable al interpretarse simultáneamente con la melodía dada por el usuario.

El concepto de segunda voz es esencial en la composición y la producción musical, pues para hacer ensambles con diversos instrumentos, es necesario que cada melodía pueda acompañar de la mejor manera a una melodía principal ya establecida, sin ser necesariamente la misma en estructura. Esta es la base de la polifonía.

Un pilar de la teoría musical del contrapunto son las reglas (del contrapunto) de Fux [10]. Existen propuestas, basadas en estas reglas, para generar automáticamente un contrapunto, pero en la literatura esto se reporta únicamente para el trabajo de *cantus firmus* que contienen únicamente notas redondas (𝅝) (e.g., Optimuse [8], Foox [15], Counterpoint Composer [19] y EVOC [18]).

En contraste, en el presente trabajo, consideramos las reglas de Fux en un sentido más amplio y abordamos el problema de generar un contrapunto musical general a partir de un *cantus firmus* rítmicamente libre; es decir, considerando que el *cantus firmus* pueda tener figuras musicales de diferente duración, por ejemplo 𝅗𝅥, ♩, ♩., entre otras y no solamente notas redondas como se ha hecho de forma clásica en la literatura. Para hacer esto posible, hemos definido una serie de criterios matemáticos para adaptar las reglas de Fux como funciones para guiar la búsqueda en un algoritmo genético.

Uno de los retos a la hora de desarrollar este trabajo fue diseñar una función objetivo capaz de producir una melodía que actúe como contrapunto a partir de la información

Tabla 1. Valores de noteToNumbered.

Nota musical	C	$C\sharp$	D	$D\sharp$	E	F	$F\sharp$	G	$G\sharp$	A	$A\sharp$	B
Nota musical numerada	0	1	2	3	4	5	6	7	8	9	10	11

de calidad sonora de *cantus firmus* (i.e., armonía, intervalos musicales, rango de la solución, escala deseada, entre otros).

En la siguiente sección, presentamos conceptos necesarios para entender la Sección 3, donde presentamos la creación del contrapunto como un problema de optimización; el enfoque algorítmico usado se encuentra en la Sección 4. Los resultados se presentan en la Sección 5 (y archivos complementarios). Finalmente, exponemos las conclusiones y recomendaciones para posibles caminos de investigación como trabajo futuro.

2. Conceptos básicos

En esta sección, establecemos la terminología y conceptos necesarios para comprender el resto de este trabajo.

Definición 1 (Nota musical) Una *nota musical* w será un elemento del conjunto W definido por $W := \{C, C\sharp, D, D\sharp, E, F, F\sharp, G, G\sharp, A, A\sharp, B\}$. La correspondiente *nota musical numerada* de w está dada por la función:

$$noteToNumbered : W \to \mathbb{Z}.$$

Descrita en la tabla 1.

Debido a la necesidad de diferenciar entre dos notas musicales de diferente altura[1], usamos la *Notación Científica de Tono* definida de la siguiente manera.

Definición 2 Una nota musical $w_{(i)}$ se considera en *Notación Científica de Tono (NCT)* cuando el subíndice $i \in \mathbb{Z}$ denota la octava correspondiente de w. El conjunto $\mathbb{SPN}$ de notas en NCT es:

$$\mathbb{SPN} := \{\ldots, C_0, C\sharp_0, D_0, D\sharp_0, E_0, F_0, F\sharp_0, G_0, G\sharp_0, A_0, A\sharp_0, B_0, C_1, \ldots\}.$$

Un ejemplo de la relación existente entre una nota musical en NCT y la partitura musical se muestra en la Figura 1, para ello se ha mostrado una pequeña melodía con notas de diferente duración, además se muestra la relación de las notas musicales y las teclas en el piano.

Con estas definiciones, es posible relacionar cada nota en $\mathbb{SPN}$ con un número entero para facilitar la gestión por computadora. Es por ello que se propone la siguiente definición, basada en el protocolo MIDI [1].

[1] La altura musical es la cualidad del sonido que determina si el sonido es grave o agudo [11].

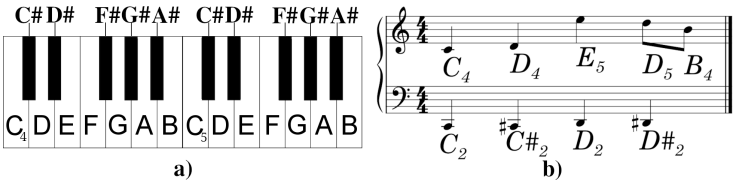

Fig. 1. a) Notas musicales en el piano. b) Ejemplo de correspondencia entre notas musicales en NCT y la partitura.

Definición 3 (Nota MIDI) Sea $w_{(i)} \in \mathbb{SPN}$, la *nota MIDI* correspondiente a $w_{(i)}$ está determinada por la función $SPNtoMIDI : \mathbb{SPN} \to \mathbb{Z}$, donde:

$$SPNtoMIDI(w_{(i)}) := noteToNumbered(w) + 12(i+1).$$

Definición 4 (Transformar nota MIDI a una *nota musical numerada*) Sea x una nota MIDI, la nota musical numerada correspondiente a x está determinada por la función $MIDIToNumbered : \mathbb{Z} \to \mathbb{Z}$, donde:

$$MIDIToNumbered(x) = x \ mod \ 12.$$

Para ejemplificar estas definiciones tomemos la nota musical A_0. La nota MIDI correspondiente es 21 ($noteToNumbered(A) + 12(0+1) = 9 + 12 = 21$). Ahora bien, la nota musical numerada correspondiente a esta nota MIDI es 9 ($21 \bmod 12 = 9$), la cual corresponde a la nota musical A de la Tabla 1.

Definición 5 (Melodía) El vector $x = (x_1, \ldots, x_n) \in \mathbb{Z}^n$ se dice ser una *melodía*, si cada x_i es una nota MIDI, para $i = 1, \ldots, n$.

Definición 6 (Escala musical numerada) Un subconjunto $S \subseteq W$ se nombra como escala musical. La *escala musical numerada* correspondiente a S se define como el conjunto:

$$nToN_{(S)} = \{z \in \mathbb{Z} \mid \ z = noteToNumbered(n), \text{ para cada } n \in S \ \}.$$

La escala musical numerada es útil porque el usuario puede decidir qué escala utilizar para el contrapunto. Para ejemplificar esta definición tomemos la escala de C mayor[2], con ayuda de la Definición 6 se obtiene el conjunto de notas musicales numeradas $\{0, 2, 4, 5, 7, 9, 11\}$. Para definir la función objetivo se utiliza la siguiente proporción, que se deriva de una media para poder evaluar determinadas características como el movimiento de la melodía y la proporción de intervalos musicales en la misma. La *proporción S-W-N* se describe a continuación.

Definición 7 (Proporción S-W-N) Sea $s \in \mathbb{N} \cup \{0\}, w \in [0,1], n \in \mathbb{N}$. La proporción S-W-N está dada por:

$$H(s,w,n) = \begin{cases} 1 & \text{si } \ s \leq w \cdot n, \\ -1 & \text{en caso contrario.} \end{cases}$$

[2] La escala de C mayor es el subconjunto de notas musicales $\{C, D, E, F, G, A, B\}$.

3. Planteamiento del problema

El problema de optimización se describe a continuación:

$$\max_{x \in \mathbb{Z}^n} f(a, b, c, d, v, w, x, y, S').$$

Sujeto a :

- $a, b \in \mathbb{SPN}$. Nota más baja y más alta correspondiente al contrapunto.
- $c, d \in \mathbb{SPN}$. Primera y la última nota deseada para el contrapunto.
- $v \in \mathbb{R}^3, v_i \geq 0, \sum_{i=1}^{3} v_i = 1$. Vector de pesos de movimiento opuesto v_1, movimiento oblicuo v_2 y movimiento directo v_3 entre el *cantus firmus* y el contrapunto.
- $w \in \mathbb{R}^{13}, w_i \geq 0, \sum_{i=1}^{13} w_i = 1$, es el vector de pesos de los intervalos musicales entre el *cantus firmus* y el contrapunto. Los pesos $w_1,\ w_2,\ dots,\ w_{13}$, son: perfecta unísona, segunda menor, segunda mayor, tercera menor, tercera mayor, cuarta perfecta, tritono, quinta perfecta, sexta menor, sexta mayor, séptima menor, séptima mayor, y octava perfecta, respectivamente.
- $y \in \mathbb{Z}^n$, es el vector de notas MIDI, *cantus firmus*.
- $S' = nToN_{(S)}, S \subseteq W$, es la escala numerada que representa la escala musical deseada para el contrapunto.

La función objetivo propuesta se describe mediante:

$$\begin{aligned} f(a, b, c, d, v, w, x, y, S') = {} & h_1(x, a, b) + h_2(x, c) + h_3(x, d) \\ & + h_4(x, y, v) + h_5(x, y, w) \\ & + h_6(x, y, S'), \end{aligned}$$

donde cada término h_i es una función para evaluar una característica musical particular deseada descrita por:

1. h_1: Esta función clasifica el cromosoma basándose en si cada una de sus notas está dentro de un rango establecido por el usuario; esta idea surgió de las reglas de Fux [10]. La función h_1 se ha adaptado para trabajar con melodías de diferentes longitudes.
   La idea principal de la función h_1 es calificar las notas de la melodía y que las notas de la melodía no tengan saltos tan precipitados. Para ello, deben estar dentro de un intervalo determinado por el usuario. Con esto, el usuario es libre de decidir qué rango de notas (graves o agudos) deberán estar en la solución del contrapunto; esto es particularmente útil si desea obtener el contrapunto para una flauta dulce o un bajo:

$$h_1(x, a, b) = \sum_{i=1}^{n} g_1(x_i, a, b), \text{ con:}$$

$$g_1(x_i, a, b) = \begin{cases} 1, \text{ if } SPNToMIDI(a) \leq x_i \leq SPNToMIDI(b), \\ -1, \text{ en otro caso.} \end{cases}$$

2. h_2 y h_3: Estas funciones clasifican un cromosoma en función de si comienza y termina con las notas deseadas por el usuario. Este criterio se toma de las reglas del contrapunto clásico [9, 16] y se adapta para ajustarse a las notas deseadas por el usuario. Ha sido posible que el usuario elija la nota inicial y final del contrapunto (notas clásicas de octava) para proporcionar al músico libertad y un mejor control del contrapunto final:

$$h_2(x, c) = \begin{cases} 1, \text{ si } x_1 = NTIToMIDI(c), \\ -1, \text{ en otro caso.} \end{cases}$$

$$h_3(x, d) = \begin{cases} 1, \text{ si } x_n = NTIToMIDI(d), \\ -1, \text{ en otro caso.} \end{cases}$$

3. h_4: Esta función contempla movimientos oblicuos, directos y opuestos, estas ideas fueron tomadas por las propuestas de Fux [10] y adaptadas para que se puedan trabajar con los pesos elegidos por el usuario. Para la puntuación, la proporción *S-W-N* verifica los movimientos directos, opuestos y oblicuos desde la primera nota hasta la i-ésima nota:

$$h_4(x, y, v) = \sum_{i=1}^{n-1} g_2(i, x_{(i+1)} - x_i, y_{(i+1)} - y_i, v), \text{ con:}$$

$$g_2(i, \Delta x, \Delta y, v) = \begin{cases} H\left(\sum\limits_{\substack{j=1 \\ [y_{(j+1)}-y_j][x_{(j+1)}-x_j]<0}}^{i} 1, v_1, n\right), \text{ si } \Delta x \Delta y < 0, \\ H\left(\sum\limits_{\substack{j=1 \\ [y_{(j+1)}-y_j][x_{(j+1)}-x_j]=0}}^{i} 1, v_2, n\right), \text{ si } \Delta x \Delta y = 0, \\ H\left(\sum\limits_{\substack{j=1 \\ [y_{(j+1)}-y_j][x_{(j+1)}-x_j]>0}}^{i} 1, v_3, n\right), \text{ si } \Delta x \Delta y > 0. \end{cases}$$

4. h_5: Esta función califica el posible contrapunto en función de la proporción de intervalos musicales que actúan entre el *cantus firmus* y el contrapunto. Esta idea se deriva clásicamente de las reglas de Fux [10], porque debe estar presente una proporción adecuada a lo largo de la melodía del contrapunto:

$$h_5(x, y, w) = \sum_{i=1}^{n} g_3\left(i, |x_i - y_i|, w\right), \text{ con:}$$

Algoritmo 1 Algoritmo genético.

1: Inicializar P una población aleatoria de n individuos.
2: establecer $generation \leftarrow 0$
3: **while** $generation < maxGenerations$ **do**
4: Hacer *selección* de la población P.
5: Hacer *cruza* de los individuos seleccionados con una probabilidad p_c.
6: Hacer *mutación* para cada individuo con una probabilidad p_m.
7: Realizar *elitismo*.
8: Hacer $generation \leftarrow generation + 1$
9: **end while**

$$g_3(i, \Delta x, w) = \begin{cases} H\left(\sum\limits_{\substack{j=1 \\ x_j - y_j = 0}}^{i} 1, w_1, n\right), & \text{si } \Delta x = 0, \\ H\left(\sum\limits_{\substack{j=1 \\ |x_j - y_j| \bmod 12 = 0}}^{i} 1, w_{13}, n\right), & \text{si } \Delta x \bmod 12 = 0, \Delta x \neq 0, \\ H\left(\sum\limits_{\substack{j=1 \\ |x_j - y_j| \bmod 12 = \Delta x \bmod 12}}^{i} 1, w_{(1+[\Delta x \bmod 12])}, n\right), & \text{en otro caso.} \end{cases}$$

5. h_6: Esta función califica el cromosoma basándose en si cada una de sus notas está dentro de una escala dada. Se prefieren consonancias sobre las disonancias. Derivado de las reglas de Fux y la teoría del contrapunto [9, 10]:

$$h_6(x, y, S') = \sum_{i=1}^{n} g_4\left(MIDIToNumbered(x_i), MIDIToNumbered(y_i), S'\right), \text{ con:}$$

$$g_4(p, q, S') = \begin{cases} 1 \text{ si } (p \in S' \wedge q \in S') \vee (p \notin S' \wedge q \notin S'), \\ -1 \text{ en otro caso.} \end{cases}$$

4. Algoritmo propuesto

El algoritmo 1 describe el algoritmo genético propuesto utilizado en este trabajo, donde:

– Los individuos son melodías (definición 5) del tamaño del *cantus firmus*.
– Selección se hizo por torneo con tres rondas como se describe en [4].
– Cruza de un punto (dos hijos por cada par de padres).
– Elitismo se aplica de manera estándar.
– Mutación sólo afecta a una sola nota del individuo.

El proceso para obtener el contrapunto usando COMPOSAIN dado el *cantus firmus* se ilustra en la Figura 2, la entrada consiste de un archivo MIDI [17] conteniendo el *cantus firmus*.

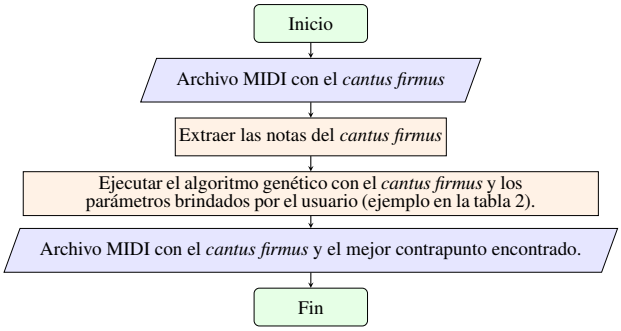

Fig. 2. Esquema general de COMPOSAIN.

Cabe señalar que se ha considerado que la función objetivo f permita definir la aptitud para los contrapuntos durante la búsqueda evolutiva. La aptitud considera el valor de la función objetivo para el *cantus firmus* y el contrapunto dado. Los individuos fueron representados como vectores de números, para esto nos inspiramos en el proyecto Foox [15] y el algoritmo genético desarrollado por Dragan [12].

La valoración del cromosoma según la cantidad que actúa en los intervalos musicales es una idea que proviene de [12], que ha sido adaptada para que a partir de un vector de pesos proporcionado por el usuario, sea posible determinar la proporción deseada en cada uno de los los intervalos musicales presentes en toda la melodía del contrapunto.

También calificamos el cromosoma usando un sistema de recompensa y castigo como se utiliza en *Foox* [15] (es decir, la función objetivo asigna una puntuación positiva (+1) si cumple con ciertas características; de lo contrario, asigna una puntuación negativa (-1)), para este caso, tal idea ha sido utilizada plenamente en la función objetivo de COMPOSAIN; por ejemplo, para la función h_1 de la función objetivo, se contará un punto positivo (+1) si la nota x_i está dentro del intervalo deseado por el usuario, mientras que si la nota x_i está fuera del intervalo, se contabilizará un punto negativo (-1) en la función objetivo, de esta forma, la función objetivo es una suma de puntuaciones donde el mejor contrapunto es el que tiene la puntuación más alta.

5. Resultados

El algoritmo propuesto se implementó en el lenguaje C++ y se ejecutó en un procesador Intel Core i7-4720HQ, con 16 GB en RAM. Presentamos aquí los resultado obtenidos para cuatro melodías elegidas como *cantus firmus*. Para efectos de que los resultados puedan ser replicados, el generador de números pseudoaleatorios utilizado en los operadores del algoritmo genético fue Mersenne twister [13]. A continuación se muestra una visualización del proceso de optimización para cada uno de ellos, junto con observaciones relevantes sobre los resultados[3].

[3] Para escuchar los resultados considerados en este trabajo, el lector puede acceder a https: //natanvilchis.org/composain/

Tabla 2. Parámetros utilizados para *Andante grazioso*.

Parámetro	**Valor**
(n, G, p_c, p_m)	$(100, 50000, 0{,}99, 0{,}07)$
(a, b, c, d)	(E_2, D_4, A_3, A_2)
v	$\left(\frac{6}{35}\ \frac{6}{35}\ \frac{23}{35}\right)^T$
w	$(0, \frac{1}{36}, 0, \frac{20}{36}, \frac{8}{36}, \frac{1}{36}, 0, \frac{1}{36}, 0, \frac{1}{36}, \frac{1}{36}, \frac{1}{36}, \frac{2}{36})^T$
S'	$\{1, 2, 4, 6, 8, 9, 11\}$

Para cada uno de los resultados se mostrará el gráfico de 50 ejecuciones independientes del algoritmo, con la finalidad de medir el desempeño del mismo con diferentes semillas aleatorias.

5.1. Andante Grazioso de W. A. Mozart (1756-1791)

En la figura 3 se muestra el proceso de maximización para *Andante Grazioso*, el valor óptimo para los parámetros proporcionados en la tabla 2 resultó de 145, que se obtiene usando la semilla 7827 para 50,000 generaciones. La solución de contrapunto resultante suena muy similar al contrapunto original compuesto por humanos.

Cabe destacar que el valor de la función objetivo del contrapunto realizado por Mozart es de 145, el mismo que el obtenido por COMPOSAIN. Por otro lado, en la melodía obtenida se controlan los saltos que realiza el contrapunto, es decir, da como resultado una melodía tranquila que no ensombrece el *cantus firmus* cuando escuchan simultáneamente. Finalmente, existe un movimiento contrario en las dos últimas notas del contrapunto que resalta la última parte del *cantus firmus* porque el contrapunto termina en notas graves.

Los parámetros utilizados se mencionan en la tabla 2. Como se muestra en la figura 3, la mediana de los datos y el máximo están en 145; además, hay una mayor cantidad de datos con un valor de aptitud en torno a 145. Se concluye con esto que la mayoría de las ejecuciones en las 50,000 generaciones obtienen una puntuación de 145.

5.2. If You're Happy and You Know It

El proceso de maximización de *If You're Happy and You Know It* se muestra en la figura 4, donde la mayoría de las ejecuciones permanecen en 189 para el valor de la función y solo hay dos valores atípicos. El valor óptimo para los parámetros proporcionados en la tabla 3 es 189, que se obtiene usando la semilla 11 para 25,000 generaciones.

El contrapunto óptimo obtenido realiza una melodía que juega entre notas bajas y altas; por ejemplo, para las últimas ocho notas, el contrapunto tiene dos partes: una parte con notas más altas y otra con notas más bajas, lo que permite tocar octavas con *cantus firmus*.

5.3. Oda a la alegría de L. Van Beethoven (1770-1827)

El proceso de maximización de *Ode to Joy* se muestra en la figura 5.

Tabla 3. Parámetros utilizados para *If you're happy and you know it*.

Parámetro	Valor
(n, G, p_c, p_m)	$(100, 25000, 0{,}99, 0{,}05)$
(a, b, c, d)	(C_4, B_6, C_4, C_6)
v	$\left(\frac{11}{48}\ \frac{26}{48}\ \frac{11}{48}\right)^T$
w	$(0, 0{,}027777, 0, 0{,}555561, 0{,}222222, 0{,}027777, 0, 0{,}027777,$ $0, 0{,}027777, 0{,}027777, 0{,}027777, 0{,}055555)^T$
S'	{0, 2, 4, 5, 7, 9, 11 }

Tabla 4. Parámetros utilizados para *Ode to Joy*.

Parámetro	Valor
(n, G, p_c, p_m)	$(100, 50000, 0{,}99, 0{,}05)$
(a, b, c, d)	(E_2, D_4, A_3, C_3)
v	$\left(0{,}171429\ 0{,}171429\ 0{,}657142\right)^T$
w	$(0, 0{,}0277778, 0, 0{,}5555555, 0{,}2222222, 0{,}0277778, 0,$ $0{,}0277778, 0, 0{,}0277778, 0{,}0277778, 0{,}0277778, 0{,}0555555)^T$
S'	{0, 2, 4, 5, 7, 9, 11 }

Tabla 5. Parámetros utilizados para *The first Noel*.

Parámetros	Valores
(n, G, p_c, p_m)	$(100, 15000, 0{,}99, 0{,}05)$
(a, b, c, d)	(E_2, D_4, A_3, D_3)
v	$\left(0{,}171429\ 0{,}171429\ 0{,}657142\right)^T$
w	$(0, 0{,}0277778, 0, 0{,}5555555, 0{,}2222222, 0{,}0277778, 0,$ $0{,}0277778, 0, 0{,}0277778, 0{,}0277778, 0{,}0277778, 0{,}0555555)^T$
S'	{1, 2, 4, 6, 7, 9, 11 }

El valor óptimo para los parámetros proporcionados en la tabla 4 es de 233, que se obtiene usando la semilla 2892 para 50,000 generaciones. El contrapunto resultante ejecuta una melodía que juega entre notas bajas y altas; por ejemplo, para las últimas ocho notas, el contrapunto tiene dos partes: una parte con notas más altas y otra con notas más bajas que permiten tocar octavas con *cantus firmus*.

Vale la pena notar que la mayoría de las ejecuciones mostradas en la figura 5 obtienen 233 para el valor de la función y solo hay un valor atípico. De esta forma, se ha comprobado que la mayoría de ejecuciones alcanzaron el valor óptimo.

5.4. The First Noel

El proceso de maximización de *The First Noel* se muestra en la figura 6, donde la mayoría de las ejecuciones llegan a 187 como valor de la función objetivo y solo hay dos valores atípicos. El valor óptimo para los parámetros proporcionados en la tabla 5 es 187, que se obtiene usando la semilla 28 para 15,000 generaciones.

Este caso de prueba nos muestra resultados interesantes. Uno de ellos es que determinadas notas graves están presentes a lo largo de la melodía, por lo que el contrapunto podría adaptarse a instrumentos como un bajo.

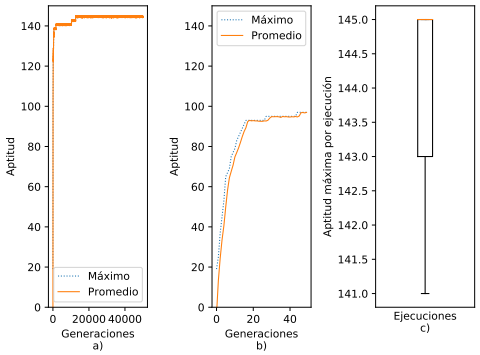

Fig. 3. *Andante grazioso* a) Aptitud máxima y aptitud media de la población por generación. b) Detalle del desempeño del algoritmo en las primeras iteraciones. c) Resultados de 50 ejecuciones independientes.

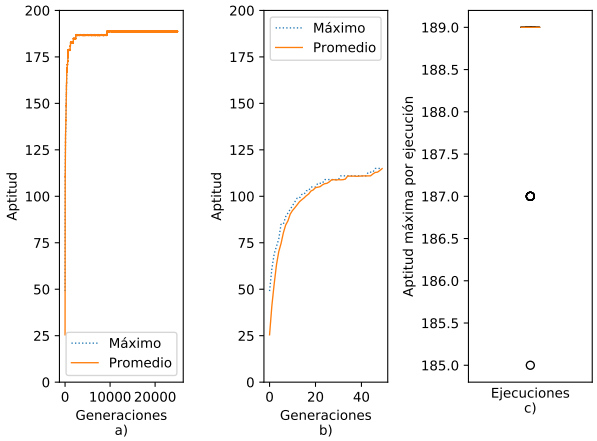

Fig. 4. *If you're happy and you know it* a) Aptitud máxima y aptitud media de la población por generación. b) Detalle del desempeño del algoritmo en las primeras iteraciones. c) Resultados de 50 ejecuciones independientes.

Por ejemplo, para el compás 10 (aproximadamente en el medio de la melodía), la ejecución del bajo es más acentuada que el contrapunto. Por otro lado, la forma en que el contrapunto y el *cantus firmus* realzan toda la canción es agradable de escuchar.

A continuación se muestran las tablas con los parámetros utilizados para cada pieza, donde n es el número de individuos, p_c es la probabilidad de cruza, p_m es la probabilidad de mutación y G es el número de generaciones.

6. Conclusiones y trabajo a futuro

En este trabajo abordamos el problema de generar un contrapunto musical, a partir de un determinado *cantus firmus*.

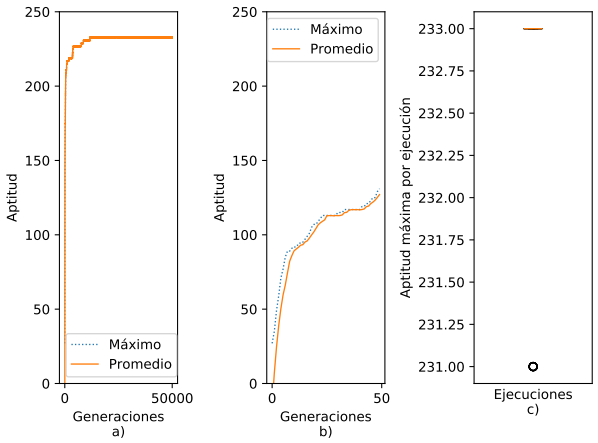

Fig. 5. *Ode to Joy* a) Aptitud máxima y aptitud media de la población por generación. b) Detalle del desempeño del algoritmo en las primeras iteraciones. c) Resultados de 50 ejecuciones independientes.

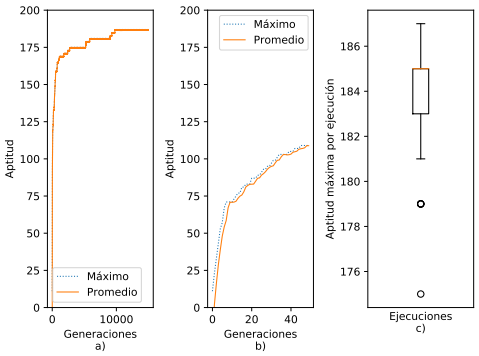

Fig. 6. *The first Noel* a) Aptitud máxima y aptitud media de la población por generación. b) Detalle del desempeño del algoritmo en las primeras iteraciones. c) Resultados de 50 ejecuciones independientes.

A diferencia de otros se consideró una figura musical de duración variable. Basados en teoría musical, planteamos algunos criterios matemáticos para adaptar las reglas de Fux como funciones para guiar la búsqueda en un algoritmo genético y generar automáticamente una melodía que pudiera escucharse en consonancia con la melodía de un usuario.

Se presentaron los resultados obtenidos para cuatro melodías de prueba. Proporcionamos un enlace web a videos para evaluar los resultados audibles donde mostramos que este trabajo tiene potencial como ayuda en la composición musical. Los resultados obtenidos son consistentes con lo que esperábamos en términos del diseño de la melodía: la nota inicial y final está de acuerdo con los parámetros y la proporción de movimientos e intervalos musicales aparece a lo largo de la melodía.

Además, mencionamos que aparece una estructura creativa como solución del algoritmo genético. Una situación interesante es que los contrapuntos obtenidos combinan notas graves y agudas resultando en una solución más versátil. En otros casos se logró igualar resultados de contrapuntos originalmente creados por el humano.

Observamos a través de las ejecuciones independientes que se obtuvieron contrapuntos ligeramente diferentes. Esto tiene sentido ya que es posible a partir de una melodía hacer adornos ligeros sin alterar la estructura principal. Como trabajo adicional, es posible extender la función objetivo propuesta para considerar otros aspectos musicales, como la estructura, los silencios y la diversificación de notas entre el contrapunto (es decir, cuando el contrapunto puede tener un número de notas diferente al del cantus firmus).

La integración de las emociones en el contrapunto (tranquilidad, tristeza, entre otras) es también un camino de investigación prometedor para dar una mayor calidad a los resultados. Para este caso, existen diferentes estudios de las emociones en la música (por ejemplo, la importancia relativa de seis variables [5], o modelo circumplex de emociones [7]), que permite conocer las características específicas que deben tener las obras en para lograr la emoción deseada de la mejor manera. También continuamos tabajando con el planteamiento del problema con un enfoque de optimización multiobjetivo.

Agradecimientos. Proyecto IPN-SIP 20211847, Natan Vilchis agradece el apoyo de la beca Conacyt para estudios de maestría.

Referencias

1. Association of Musical Electronics Industry AMEI and MIDI Manufacturers Association MMA: Universal MIDI packet (UMP) format and MIDI 2.0 protocol, pp. 1–82 (2020)
2. Biles, J. A.: Improvizing with genetic algorithms: Genjam. In: Proceedings of Evolutionary Computer Music, pp. 137–169 (2007) doi: 10.1007/978-1-84628-600-1_7
3. Bloxam, M. J.: Cantus firmus. Oxford University Press (2001) doi: 10.1093/gmo/9781561592630.article.04795
4. Eiben, A. E., Smith, J. E.: Introduction to evolutionary computing. vol. 53, pp. 52,57,84,89 (2003) doi: 10.1007/978-3-662-44874-8
5. Farnsworth, P. R.: The social psychology of music (1958)
6. Garay-Acevedo, A.: Fugue composition with counterpoint melody generation using genetic algorithms. In: Proceedings of International Symposium on Computer Music Modeling and Retrieval, vol. 3310, pp. 96–106 (2005) doi: 10.1007/978-3-540-31807-1_7
7. Gerber, A. J., Posner, J., Gorman, D., Colibazzi, T., Yu, S., Wang, Z., Kangarlu, A., Zhu, H., Russell, J., Peterson, B. S.: An affective circumplex model of neural systems subserving valence, arousal, and cognitive overlay during the appraisal of emotional faces. Neuropsychologia, vol. 46, no. 8, pp. 2129–2139 (2008) doi: 10.1016/j.neuropsychologia.2008.02.032
8. Herremans, D., Sörensen, K.: Composing fifth species counterpoint music with a variable neighborhood search algorithm. Expert systems with applications, vol. 40, no. 16, pp. 6427–6437 (2013) doi: doi.org/10.1016/j.eswa.2013.05.071
9. Jeppesen, K.: Counterpoint: The polyphonic vocal style of the sixteenth century (2013)
10. Mann, A.: The study of counterpoint. vol. 1943 (1971)

11. Martín, I. C.: Iconografía musical infantil. ICONO 14, Revista de Comunicación y Tecnologías Emergentes, vol. 4, no. 1, pp. 1–24 (2006)
12. Matić, D.: A genetic algorithm for composing music. Yugoslav Journal of Operations Research, vol. 20, no. 1 (2010)
13. Matsumoto, M., Nishimura, T.: Mersenne twister: A 623-dimensionally equidistributed uniform pseudo-random number generator. ACM Transactions on Modeling and Computer Simulation, vol. 8, no. 1, pp. 3–30 (1998) doi: 10.1145/272991.272995
14. Nam, Y. W., Kim, Y. H.: Automatic jazz melody composition through a learning-based genetic algorithm. In: Proceedings of International Conference on Computational Intelligence in Music, Sound, Art and Design, pp. 217–233 (2019) doi: 10.1007/978-3-030-16667-0_15
15. Nicholas, T.: Foox (2012) https://github.com/ntoll/foox
16. Randall, R.: Class Notes for Counterpoint. pp. 1–3 (2007)
17. Sapp, C. S.: C++ library for parsing standard midi files (2021) https://midifile.sapp.org/
18. Weale, T., Seitzer, J.: Evoc: A music generating system using genetic algorithms. In: Proceedings of International Joint Conference on Artificial Intelligence, pp. 1383–1384 (2003)
19. Weaver, J. L.: Counterpoint composer (2021) https://counterpointservice.cfapps.io/