La capacidad investigativa del algoritmo y el código: trazos sobre el ciclo práctica artística, tecnología e investigación

Aaron, Hernani

3 de junio de 2021

Capítulo 1

La capacidad investigativa del algoritmo y el código: trazos sobre el ciclo práctica artística, tecnología e investigación

Aaron Escobar Castañeda y Hernani Villaseñor Ramírez

1.1. Introducción

Escribir un texto colaborativo implica negociar perspectivas, ideas, términos y conceptos para delinear un problema y sus preguntas. Este capítulo parte de nuestras investigaciones doctorales bajo el reto de construirlo con temáticas en proceso de problematización. Ante ello hemos buscado en nuestros objetos de estudio —el algoritmo y el código— similitudes y diferencias que nos permitan expresar el tema que queremos exponer. De dichos objetos problematizamos su capacidad investigativa dentro de un ciclo de retroalimentación compuesto por la práctica artística, el desarrollo tecnológico y la investigación que se da en nuestros trabajos. Este ciclo es puesto a prueba en el laboratorio de experimentación que vemos en el performance de nuestras prácticas.

Así mismo, este ciclo lo encontramos activo en un ensamblaje sociomaterial conformado por humanos, máquinas, prácticas y tecnologías que se encuentra en la creación de música con computadoras. Específicamente en la improvisación libre con aprendizaje y escucha de máquinas y en el live coding, las que recaen en el marco de prácticas y técnicas musicales que se desprenden del cruce entre la música y las ciencias de la computación. El término nodal de nuestras prácticas en este artículo es el de *algoritmicidad* o *agencia algorítmica* desde donde

nos preguntamos ¿cómo transforman los algoritmos nuestras prácticas musicales? y ¿cómo entra el artista-programador-investigador en el ciclo mencionado?

1.2. Algortimo y código

Esta sección vincula los términos algoritmo y código fuente a partir de las siguientes preguntas: ¿qué relación tiene un algoritmo con su código? y ¿cómo se expresan a través de las prácticas musicales por computadora? Para abordar estas preguntas partimos de que el código computacional está compuesto por una serie de símbolos y caracteres organizados en un lenguaje de programación, los cuales son legibles para el humano y, posteriormente, al ser traducidos en código binario son ejecutables por una máquina. A su vez, el algoritmo define los procesos abstractos para resolver un problema e incluye las estructuras lógicas que lo integran así como los procesos de control que convierten esa lógica en una máquina capaz de producir un resultado específico o no. Por ello tiene una materialidad inherente al código y a los lenguajes de programación que harían posible su ejecución [1]. No está demás decir que en una computadora, el código posibilita la implementación del algoritmo. Así, podemos idear al algoritmo sin llegar a codificarlo y escribir código sin conceptualizar cabalmente al algoritmo, aunque involuntariamente seguiríamos conceptualizandolo, tal vez, en un estado primario, de tal modo que, podemos ver al algoritmo en el código.

Profundicemos más en la idea del algoritmo para visualizar sus potencias más allá del ámbito digital, aquellos espacios donde podrían adquirir otras formas de materializarse, generar significados y afectar las prácticas donde se insertan. Para ello, proponemos utilizar el concepto de *algoritmicidad* o *agencia algorítmica* (Holger, 2016), definida como la capacidad que tienen los algoritmos para establecer procesos de intercambio y afectación con su entorno tanto dentro como fuera de una computadora. Una función importante de la algoritmicidad es "la habilidad de componer algoritmos", así como la posibilidad que tenemos para experimentar con ellos, ya sea en un contexto performático en la web, en vivo, o en el desarrollo de software.

En la algoritmicidad se insertan las prácticas de los ingenieros y científicos de la computación pero también de artistas, programadores independientes y usuarios. Así, la algoritmicidad abarca la implicación que tienen los algoritmos dentro y fuera de los procesos ligados a su ejecución, de aquí se desprende, su agencialidad para transformar las prácticas vinculadas a su uso y devenir en una materialidad distinta a sus fines primarios. Esta perspectiva busca prevenir la estabilización y las oscilaciones dirigidas a la simple demostración del algoritmo a través de una prueba autocontenida que lo llevaría a perder su capacidad investigativa.

En vez de apostar por una estética probabilista y predictiva se parte de una "tendencia especulativa intrínseca de la computación, que produce una novedad genuina incapaz de explicarse por fuerzas externas o condiciones iniciales" (Parisi, 2013). Los algoritmos, incluidos

implícitamente en la estructura del hardware, no son entidades con estructuras fijas, sino que aparecen impermanentes a través de la posibilidad intrínseca de su ejecución, manipulación y actualización ad infinitum[2]. A este respecto el algoritmo presenta una agencialidad dinámica, espaciotiempomateriandose[3], recreándose y ampliando sus fronteras, manifestando una performatividad en continua transformación y surgimiento. Por otro lado, encontramos importante partir de una perspectiva crítica hacia esta agencialidad, es decir, una aproximación que permita indagar cómo, cuándo, dónde y porqué se produce la agencia, de manera que terminemos con una interfaz dinámica entre el investigador y su aparato técnico, una herramienta epistémica que serviría para desdibujar cada vez más las fronteras entre estos agentes y reforzar la injerencia/proximidad/entrelazamiento de la máquina en el artista-investigador y viceversa (Rheinberger 2013, Barad, 2012).

Lo anterior lo situamos desde nuestra propia práctica, que involucra una aproximación heurística hacia los algoritmos basada en la realización constante de pruebas y experimentos desde la actividad de la escritura del código, ya sea en el momento del concierto, en el desarrollo de objetos computacionales o en la práctica investigativa. Con esto no proponemos enfocarnos en el análisis del código fuente de un algoritmo, aunque esta sea una forma de aproximarnos a sus implicaciones, sino a aquello que posibilita en el contexto las prácticas artísticas con código, de manera específica, su algoritmicidad en las prácticas sonoras y musicales. En este contexto, podemos pensar, por el lado del live coding, en un panorama que ve al humano al centro de la creatividad con computadoras, y por el lado de la creación musical con aprendizaje y escucha de máquinas, en un espacio donde múltiples agencias tanto humanas y no humanas inciden en el proceso creativo. Estos escenarios nos dejan ver diferentes aproximaciones de la relación humano-máquina en la creación musical con lenguajes de programación, desde la modificación de algoritmos en vivo, pasando por la colaboración entre humanos y agentes artificiales hasta la creación/interpretación autónoma de las máquinas.

1.3. El ciclo de retroalimentación de la práctica artística, el desarrollo tecnológico y la investigación: la condición del artista investigador programador

En esta sección definimos nuestra condición de artistas, investigadores y programadores para aclarar el contexto desde donde escribimos. Al momento de redactar este capítulo ambos autores nos encontramos realizando el doctorado en música en el área de tecnología musical. Nuestras investigaciones, aunque tratan temas distintos dentro del ámbito de la música computacional, están conectadas por la aproximación a la investigación desde la práctica artística y el desarrollo tecnológico. En este sentido, escribimos sobre casos que se transforman conforme

avanza cada investigación y cuya materialidad está constituida por algoritmos escritos con código computacional. Por un lado, estos algoritmos están aplicados a la escucha y al aprendizaje de máquinas en el Sistema de Escucha de Automática para la Improvisación Libre (SEALI)[4]; por otro lado, a la escritura y modificación de código en vivo en el proyecto SonoTexto.[5]

La forma de aproximarnos a la investigación, desde la música y su relación con la tecnología, nos lleva a pensar en cómo estructuramos nuestros trabajos de investigación y a qué procesos se ven sujetos. Desde la condición de artista programador, a veces no se parte de una perspectiva ingenieril o un método científico riguroso que, dadas ciertas condiciones, busque resolver un problema específico y replicable. Contrario a esto partimos de un proceso heurístico que muestra fenómenos intrincados generados por la interconexión de múltiples agencias entrelazadas. Además, en este laboratorio experimental, partimos de perspectivas sonoras ligadas a la improvisación libre y la exploración compleja de patrones rítmicos, con envolventes y tímbricas caprichosas, que, más que buscar una estructuración ligada a las prácticas musicales de occidente trata de generar sus propias lógicas formales. Al trabajar con un algoritmo (e.g. LSTM, Mean Shift Clustering, PCA)[5] que responde de manera no lineal a cambios mínimos, a veces resulta difícil establecer una imagen clara sobre la metodología o los pasos a seguir para obtener el resultado esperado. Incluso, en algunos casos el científico de datos se vuelve un manipulador de perillas hasta que obtiene el resultado deseado. Desde esta limitante o posibilidad, encontramos que emerge cierta reestructuración de la problemática inicial, la cual se vuelve móvil y permanece en estado de transformación durante todo el proceso. La integración del agente no humano a la fórmula creativa, el algoritmo, necesariamente modula los objetivos ampliando los espacios de posibilidad. A partir de las interrelaciones de todos los agentes implicados en el sistema conformado por la práctica artística, el desarrollo tecnológico y la investigación es que encontramos, en la retroalimentación de este ciclo, una herramienta donde humanos y no humanos traspasan sus límites biotecnológicos para .[en]carnarün estado de constante afectación durante el acontecer del proceso.

Por otro lado, la aparente disociación de campos especializados, como el existente entre la música y las ciencias de la computación, nos lleva a cuestionarnos cómo nos relacionamos e intentamos borrar cada vez más la brecha entre campos de especialización. Una alternativa es pensarnos desde la pluralidad de roles; mientras somos programadores seguimos siendo compositores, improvisadores, livecoders, investigadores e incluso nuestro propio público. Estos roles, juegan un papel de suma importancia dentro de lo que concebimos como el ciclo de retroalimentación de la investigación artística y tecnológica, los cuales no se pueden separar, sino que actúan a través de múltiples hilos entrelazados; vale aquí la metáfora a los hilos entrlazados (multiple threats) que realiza una computadora al mismo tiempo para .[a]ctuar"de manera óptima. En este sentido el contexto de la investigación en tecnología musical está inmerso en el giro que han hecho las escuelas de arte al convertirse en facultades. Carbone et al. (2019)

relatan el encuentro de los artistas con el mundo científico de la academia en este giro y ven en el experimento un punto nodal entre los ámbitos del arte y la ciencia. Es así que el vocabulario y metodologías científicas permean la investigación en el ámbito del arte, por ejemplo cuando nos referimos al espacio de trabajo como estudio y laboratorio; o cuando intercambiamos los términos creación artística y experimentación. Los autores dan al experimento realizado en la ciencia una connotación de tratar y probar, mientras que el mismo término en el arte se trata de nuevas formas de hacer.

Una aproximación del artista investigador, en el contexto de la investigación, estudia la creatividad humana con lenguajes de programación sonora al centro de esta relación. Desde esta perspectiva, que tiene que ver con la práctica del live coding, el artista investigador transcribe y modifica sus algoritmos en el momento de una presentación, los cuales se caracterizan por ser pequeños y de parámetros modificables. Esta práctica implica probar y memorizar código en varias circunstancias, es decir, los algoritmos no emergen de la nada sino que provienen del bagaje de conocimientos, experiencias y latitudes de quien práctica live coding. Marije Baalman (2015) se refiere a esta forma de escritura internalizada en la memoria y en el cuerpo como la incorporación del código. Durante este procedimiento, el humano memoriza el código del algoritmo y lo que posiblemente obtendrá a su salida y se familiariza con el proceso físico de teclearlo. En este sentido, la habilidad de lecto escritura del código, que algunos autores refieren como literacidad de la programación (Marino, 2020) y la capacidad estética de la programación (Soon y Cox, 2020) conforman un campo de reflexión en el que el código del algoritmo, en sus procesos de iteración, introducen los referentes teóricos y estéticos del artista investigador, ampliando el bucle de la práctica artística al incluir prácticas de investigación y desarrollo de software a su quehacer artístico.

1.4. Ensamblajes agenciales

Vayamos pues a problematizar sobre los tipos de agencias implicadas en este sistema complejo, implícito en el ciclo de retroalimentación que intentamos analizar. Al hablar de agencias en términos de co-creación nos referimos al espacio donde agentes humanos y no humanos espaciostiemposmateriandose a distintos niveles están entrelazados en la generación de una obra o una improvisación. Aquellos casos que detectamos son: 1) la programación creativa, donde el artista-programador se sitúa como creador o modificador del algoritmo el cual es empleado como recurso de improvisación, experimentación y conocimiento. 2) El proceso de co-creación de la obra donde el algoritmo y el artista están implicados generando una asociación de actantes, que de acuerdo con Latour respondería al segundo significado de la mediación técnica (Latour, 2001). 3) El propio algoritmo visto como obra/desarrollo en sí mismo. Y 4) el algoritmo como creador/ejecutante/intérprete de un proceso artístico generativo (creatividad computacional).

Para analizar estos cuatro puntos hay que considerar las implicaciones del desarrollo e implementación en código que conlleva la creación de una obra artística. Lo primero es afirmar que ningún conocimiento parte de cero, sería ingenuo pensar que el artista-programador apartado de todo contexto es capaz de generar un desarrollo tecnológico que no requiera de la línea evolutiva de la algoritmicidad. En este punto entendida como la actividad humana que gira en torno al proceso de estudio, modificación, transformación, reapropiación y distribución al que están sujetos los algoritmos y su implementación. Segundo, todos los agentes humanos y no humanos interconectados sufren un proceso de afectación mutua que los lleva a transformar sus objetivos iniciales. Bruno Latour plantea el concepto de rodeo para referirse a los procesos de transformación de los objetivos iniciales que surgen del proceso de afectación de dos o más agentes implicados en un problema. Resulta interesante analizar este concepto para referirnos a cómo el algoritmo, el artista-programador y el contexto, están en un flujo de constante intercambio energético, de ideas, conceptos y propuestas donde "las técnicas actúan como modificaciones de las formas". (Latour, 2001)

Partir de esta perspectiva donde cada uno de los agentes implicados juega un papel de afectación directa sobre los otros, deja ver que sería imposible pensar en desvincularlos dentro del proceso de generación de un producto artístico. En este sentido, el caso del artista-programador que se posiciona como el creador absoluto de la obra resulta un tanto ficticio, aunque en contextos artísticos, por ejemplo, los que se sitúan en la producción museística es común ver este fenómeno. En este caso, solo vemos en la ficha de la obra, el nombre de la obra y del artista, la técnica empleada y el año de producción, más no el nombre de los técnicos, programadores, gestores o curadores que materializaron el proyecto. Si vamos más a fondo, podemos ir abriendo cada una de las cajas negras que a su vez contienen más cajas negras implicadas en la concreción de una obra artística computacional; desde los entrelazamientos de múltiples librerías y software, hasta los desarrolladores de lenguajes de programación, y todo el ensamblaje de agentes intermediarios que se adhieren al proceso de producción. Por tanto, los objetos estéticos producidos articulan el espacio de la algoritmicidad o agencialidad algorítmica (Holger, 2016), en la cual el humano y la máquina se someten a un proceso abierto, no lineal, en constante evolución, posibilitado por la dinámica de intercambio y afectación mutua de actantes.

Como hemos visto, el algoritmo por "naturaleza"no permanece estático, aunque hay algoritmos que permanecen inamovibles en el tiempo debido, por ejemplo, a cuestiones burocráticas (Peeters, 2020). Un elemento fundamental de la dinámica de los algoritmos es que son capaces de procesar datos sujetos a distintas temporalidades a través del tiempo. Ya sean algoritmos que procesan bases de datos compuestas por series de tiempo sobre algún fenómeno natural, económico, político, social, en constante evolución, o, una base de datos que contenga muestras de sonido pero que requieren de la ejecución de un proceso temporal para poder analizarse. A medida que se desarrolla el algoritmo es evidente que evoluciona y que sus propiedades ejecuta-

bles hacen posible su naturaleza experimental, cambiante, abierta a la posibilidad, la afirmación y la negación. En las prácticas que lo usan para producir objetos artísticos, hay que considerar la cualidad performativa (como el intérprete de SuperCollider) del algoritmo así como la performatividad del live coder con el código. Considerando estas posibilidades no es importante plantear la pregunta sobre si las máquinas pueden o no ser creativas sino más bien cómo se articulan estos espacios de posibilidad entre agentes humanos y no humanos y cómo dan lugar a la capacidad investigativa y creativa de tal composición de actantes.

Tal ensamblaje lo podemos ver en ciclos que actúan en conjunto por los que pasa el pensamiento y el código en nuestro cuerpo y en la computadora. Primero pensamos un concepto en forma de código y a través de nuestro sistema motor lo tecleamos al editor; el código aparece en la pantalla y es percibido por nuestro sistema sensorial que conecta nuevamente con el concepto. En este ciclo, el código que ingresa a la computadora a través del teclado va a la memoria del programa, es interpretado, va al procesador y sale como datos que son convertidos a sonido o luz, los cuales son percibidos por nuestro sistema sensorial junto al código mostrado en la pantalla. un bucle de retroalimentación que incluye al humano y al código en la práctica de live coding. Alex McLean (2011) describe dicho bucle en el proceso de acción y reacción en la programación bricolage, el cual conecta un concepto que se codifica en un algoritmo a su salida lo que es percibido y modificado para continuar el bucle constituido por concepto, algoritmo, salida, percepción, modificación. Por su parte, Marije Baalman (2015), describe el proceso de traducción de nuestros conceptos a código computacional cuando realizamos live coding. La autora traza un bucle compuesto por dos

1.5. Algoritmicidad en el live coding

La práctica del live coding es un ámbito en el que podemos ver expresada la algoritmicidad. En esta práctica el artista programador experimenta e interactúa con algoritmos sencillos generalmente escritos en forma de patrones. Aquí la prueba y error en la escritura de código para generar sonido es una constante durante un concierto. Generalmente la escritura de código es empleada para modificar los valores de los algoritmos a lo largo del concierto. El manifiesto del live coding, escrito por el colectivo Toplap en el año 2004 (Ward et al.), expresa en uno de sus puntos que los algoritmos son pensamientos. De esta manera adjudica al humano la conceptualización del algoritmo en el momento de una presentación o concierto de live coding. Aunque hay que precisar que muchos de los mecanismos algorítmicos, o patrones, usados en el live coding han sido previamente implementados en el software por sus desarrolladores. Esto es, la interacción con el algoritmo consiste en transcribir los patrones incluidos en el software utilizado, por ejemplo SuperCollider o TidalCycles, durante el concierto para modificar sus valores de entrada y, de esta manera, obtener un resultado sonoro. La relación con el algoritmo

durante una presentación de live coding no consiste en diseñarlo sino en escribir los valores de sus parámetros en vivo. En este sentido, Brookshear (2012) menciona que el algoritmo es un contenedor de conocimiento y que una vez que se programa para cumplir una tarea determinada no es necesario entender sus principios ya que la solución que resuelve dicha tarea se encuentra codificada en él. Entonces, la interacción con algoritmos en el live coding se da desde una abstracción que nos permite manipular sus valores de entrada para construir el discurso sonoro durante la improvisación. Veamos el ejemplo del patrón Pbrown de SuperCollider.

```
Pbrown(0.0, 1.0, 0.125, inf);
```

Ejemplo 1. Patrón de SuperCollider que realiza la tarea de generar valores aleatorios con un comportamiento browniano.

Para mostrar cómo sucede la interacción con algoritmos en el live coding tomaremos un patrón de SuperCollider. El ejemplo 1 muestra el patrón Pbrown, el cual genera valores aleatorios entre dos puntos. Sus parámetros, en el orden que están escritos, indican lo siguiente: lo es el valor más bajo, hi es el valor más alto, step es el paso máximo que se puede alcanzar entre ambos valores y length las veces que genera valores entre los puntos bajo y alto. Una vez que memorizamos la estructura y función del patrón podemos escribirlo en el momento de la improvisación y modificar sus parámetros en el transcurso de esta. Los valores que dicho patrón genera los podemos asignar a la altura, la duración entre sonidos o a otros parámetros musicales como muestra el ejemplo 2. De esta manera estamos diseñando comportamientos e interactuando con un algoritmo transcrito en el momento cuya funcionalidad está implementada en el programa a través de los patrones.

```
Pdef(\brown, Pbind(
            \instrument, \default,
            \dur, Pbrown(0.125, 0.5, 0.1, inf),
            \note, Pbrown(0, 12, 1, inf)
            )
).play
```

Ejemplo 2. La rutina contenida en Pdef muestra la asignación del patrón Pbrown a los parámetros de duración y nota. Podemos decir que esta estructura es la composición de un algoritmo con los recursos de SuperCollider, es decir, su algoritmicidad.

```
length.value(inval).do {
            loVal = loStr.next(inval);
```

```
        hiVal = hiStr.next(inval);
        stepVal = stepStr.next(inval);
        if(loVal.isNil or: { hiVal.isNil } or: { stepVal.isNil } ) { ^inval };
        cur = this.calcNext(cur, stepVal).fold(loVal, hiVal);
        inval = cur.yield;
    };
```

Ejemplo 3. Extracto del código fuente de Pbrown que muestra el algoritmo para obtener un movimiento browniano. Se puede consultar el código entero en el archivo Pattern.sc del código fuente de SuperCollider.

El ejemplo 3 muestra un extracto del mecanismo que calcula el movimiento browniano. Lo que expresa la variable cur es un cálculo aleatorio del siguiente valor comprendido entre los valores bajo y alto. Este cálculo seguirá ocurriendo mientras el elemento yield siga requiriendo un valor. Lo que quiero señalar con este ejemplo es que el mecanismo para obtener un movimiento browniano ya ha sido implementado y no requerimos saber como funciona, más bien necesitamos saber que valores ingresar en la abstracción de la clase mostrada en los ejemplos 1 y 2. De esta forma la algoritmicidad en el live coding ocurre sobre una serie de abstracciones que permiten relacionarnos con los algoritmos desde un nivel alto de escritura de código. En palabras de Christopher Haworth (2018), la idea del algoritmo como pensamiento en el live coding lo pone cerca de la idea abstracta musical más que del uso de herramientas en un paso de la materialidad del código a lo que llama la cuasi inmaterialidad de los algoritmos.

La relación con los algoritmos en el live coding lleva a sus practicantes a relacionarse con el código de programación en una actividad exploratoria que convierte la ejecución del código de los algoritmos en música o sonido. Este escenario plantea además del lugar ocupado por la práctica artística uno para la práctica investigativa que empuja la observación y la acción de quien investiga desde perspectivas de análisis, reflexión y creación. Sin embargo, esta relación no ocurre de facto, hay que colocar sus elementos en el bucle de retroalimentación de la práctica y la investigación.

1.6. Algoritmicidad en la improvisación libre con aprendizaje y escucha de máquinas

Un caso de estudio interesante a mencionar que permite entender el entrelazamiento del ciclo práctica artística, desarrollo tecnológico e investigación y ver cómo surgen de manera iterativa los procesos de intercambio entre éstas así como las interacciones entre agentes humanos y no humanos, es el proyecto SEALI y su interacción en contextos de improvisación libre.

Este proyecto surge de la necesidad por expandir las posibilidades creativas dentro de la improvisación libre caracterizada por elementos que tal vez no responden a una aproximación tradicional musical sino que buscan cierta expansión a través de la creación de nuevos instrumentos, formas de interactuar y exploraciones que indagan sobre el proceso creativo desde el contenido espectral, la energía, el movimiento, la complejidad rítmica, la novedad y los intersticios que se puedan formar de todas ellas que en combinación producen una estructura compleja que da forma a la improvisación.

En el proceso de desarrollo de este sistema interactivo, podemos destacar 5 partes fundamentales las cuales se encuentran entrelazadas por múltiples procesos de intracción. Estos son; Desarrollo algorítmico, generación de materiales sonoros (conocimiento tímbrico), consolidación de un corpus de improvisaciones libres (conocimiento estructural), sistema interactivo en tiempo real (que incluye identificación, predicción estructural y reacción) y el propio sistema de interacción humano-máquina tanto en el performance como en el desarrollo conceptual algorítmico.

Por un lado está el desarrollo algorítmico del sistema que implica desde el comienzo un proceso recursivo ya que parte de la generación de una base de datos de materiales sonoros los cuales fueron generados por los improvisadores tomando como elementos detonadores la propia forma en la que el sistema "escucha". Esto lo hace desde las descripciones numéricas que diferentes descriptores de audio (flatness, spectral centroid, MFCCs, loudness, spectral complexity, etc) posibilitan. [6]

Una vez conformada esta información el sistema la segmenta con base en cambios tímbricos sustanciales, estos segmentos son analizados por los descriptores de audio y posteriormente clasificados por un algoritmo de agrupamiento, finalmente esta nueva información es presentada a una red neuronal profunda la cual se encarga de generar un modelo que puede identificar las diferentes clases propuestas por el algoritmo de agrupamiento. El modelo puede ser ejecutado en tiempo real y hacer predicciones sobre nuevos materiales presentados. Para el conocimiento estructural de las improvisaciones se recurre a una aproximación similar al proceso de segmentación y esta información es reorganizada en series de tiempo y analizada por una red neuronal recurrente (LSTM) la cual es capaz de predecir secuencias temporales a corto mediano y largo plazo. A través de estos procesos de reconocimiento el sistema interactivo en tiempo real va tomando decisiones en términos de novedad, a nivel tímbrico, energético e interactivo de manera tal que propone materiales similares o contrastantes a los que está escuchando de acuerdo con el conocimiento que tiene sobre lo que ocurre comúnmente dentro de una improvisación.

En el performance ocurren fenómenos de retroalimentación que interpelan desde la adecuación de los diferentes elementos tanto al improvisador como al propio sistema, si este se está escuchando a sí mismo. Uno de éstos es la propuesta musical en términos estructurales que hace el sistema basándose en el estilo particular que algún improvisador propuso en un álbum.

De esta manera se conserva cierta esencia a nivel estructural con la cual el sistema transita la improvisación desde ese bagaje de conocimiento que le permite adecuar su comportamiento dadas ciertas situaciones sonoras. Este proceso podría ser visto como un agente que encapsula el estilo musical y que trasciende el tiempoespaciomateria insertandose en ese continuo e indivisible fluir iterativo en contextos completamente distintos. Al conservar ciertos rasgos cadenciales identificables por un improvisador, la improvisación puede conducirse en términos estructurales desde ese gran conocimiento que caracteriza este estilo.

Aquí podemos observar los múltiples bucles de agencialidad y afectaciones mutuas humano-máquina que se producen desde el proceso de conceptualización, programación e interpretación, de aquí se derivan procesos de actualización del sistema dadas las problemáticas que surgen del performance. Posteriormente viene un proceso de adecuación, consolidación, pruebas y re-ensamblaje de las partes modificadas de acuerdo con las problemáticas detectadas. Esto genera la apertura a nuevas funcionalidades, cambios metodológicos y posibilidades del sistema donde el concepto de rodeo de Latour permea todo el proceso creativo. Los objetivos iniciales no solo cambian sino que se van complejizando cada vez que este ciclo hace un giro. Cabe resaltar que el giro no siempre sucede completo, en la misma dirección ni en el mismo orden. Por momentos el algoritmo tiene que readaptarse para coincidir con ciertos principios perceptivos de escucha dentro la estética particular de la libre improvisación, en otros, simplemente se toma en cuenta como principio constructivo el punto de partida que la complejidad e interrelación de los algoritmos proponen.

1.7. Conclusiones preliminares

Los términos y preguntas que arrojó este texto quizá son imposibles de resolver en un escrito de las dimensiones propuestas, no obstante nos han permitido identificar una forma de relacionarnos con nuestros materiales de trabajo y objetos de estudio a partir del concepto algoritmicidad. Hemos planteado que la secuencia de los ejes de la práctica artística, el desarrollo tecnológico y la investigación están afectados por la agencialidad de los algoritmos en las prácticas de la escucha y aprendizaje de máquinas y el live coding.

Las prácticas artísticas con algoritmos involucran ciclos dentro de los procesos que realiza la computadora. El artista investigador programador puede controlar e incorporarse a esos ciclos como en el caso del live coding o diseñarlos para ejecutar desiciones, selecciones y catalogaciones como en el caso del aprendizaje y la escucha de máquinas. Para incorprar la actividad de la investigación al ciclo de la práctica artística y el desarrollo tecnológico, este se elonga en un proceso de reflexión que provoca preguntas surgidas de la práctica y son contestadas a través de la teoría. Las respuestas son traídas de vuelta al ciclo de la programación para retroalimentarla, aunque también pueden provocarse desde la teoría y responderese o comprobarse en la práctica.

La formulación de prácticas en el ámbito científico académico nos lleva al juego de lo experimental en el sentido de crear una metáfora para la práctica performativa al verla como un laboratorio de experimentación y, por otro lado, a pensar la escritura de textos cómo si fueran procesos de programación. En este sentido, incorporamos el pensamiento del arte, la programación y la investigación así como las ideas y problemas que circulan en las actividades que nos competen. Una actividad en la que se tocan y operan varios ejes a la vez.

1.8. Referencias

Baalman, M. (2015). Embodiment of code.

Kleiman, A (2012). Interactions: an interview with Karen Barad, Mousse Magazine, (34) 76-81.

Kowalski, R. (1979). Algorithm = logic + control. Communications of the ACM, 22(7), 424–436.

Rheinberger, H-J. (1998) Experimental systems, graphematic spaces. In T Lenoir (Ed.), Inscribing Science: scientific texts and the materiality of communication (pp 285-303). Palo Alto: Stanford University Press.

Manovich, L. (2013). Software takes command. New York: Bloomsbury Academic. Parisi, L. (2013). Contagious architecture: computation, aesthetics, and space. Cambridge, MA:MIT Press.

Rheinberger, H.-J. (2013). Forming and Being Informed: Hans-Jörg Rheinberger in conversation with Michael Schwab. In M. Schwab (Ed.), Experimental systems. future knowledge in artistic research (pp. 198–219). Leuven: Leuven University Press.

Schwab, M. (Ed.). (2013). Experimental systems. future knowledge in artistic research. Leuven: Leuven: Lueven University Press. Xambó, A., Lerch, A. y Freeman, J. (2019). Music Information Retrieval in Live Coding: A Theoretical Framework. Computer Music Journal, 42(4), 9-25.

Soon, W. y Cox, G. (2020). Aesthetic Programming: A handbook of software studies. Open Humanities Press.

./bib/bib1