UNIVERSIDAD MICHOACANA DE SAN NICOLÁS DE HIDALGO

FACULTAD DE INGENIERÍA ELÉCTRICA
DIVISIÓN DE ESTUDIOS DE POSGRADO

"MONITOREO AUTOMÁTICO DE ESTACIONES DE RADIO QUE TRANSMITEN POR INTERNET"

TESIS

QUE PARA OBTENER EL GRADO DE
MAESTRO EN CIENCIAS EN INGENIERÍA ELÉCTRICA

PRESENTA
ING. OMAR NUÑEZ SAUCEDO

DIRECTOR DE TESIS
DR. JOSÉ ANTONIO CAMARENA IBARROLA

MORELIA, MICHOACÁN AGOSTO DE 2012

Esta tesis no esta dirigida a nadie en particular

Agradecimientos

Gracias al Dr. Antonio Camarena, por su inagotable paciencia con mi persona y su amplio conocimiento del tema que han hecho posible este trabajo, por su guía y por la confianza que ha depositado en mi y todos sus invaluables consejos.

Gracias al Dr Mario Graff, Dr Héctor Tejeda, Dr Edgar Chávez y Dr Félix Calderón por haber dedicado su valioso tiempo a la revisión de este trabajo y sus valiosas recomendaciones.

Gracias a mi Alma Máter, la Universidad Michoacana de San Nicolas de Hidalgo y a la Facultad de Ingeniería Eléctrica por poner a mi disposición valiosos recursos durante mi formación.

Gracias al Consejo Nacional de Ciencia y Tecnologia (CONACYT) por el apoyo brindado.

Gracias a mi familia y a mis amigos por el apoyo brindado.

Omar Núñez Saucedo.

Lista de Publicaciones

"Automatic Monitoring the content of audio broadcasted by Internet Radio Stations"
Omar Nuñez, Antonio Camarena
Mexican International Conference on Artificial Intelligence 2012
Noviembre, 2012. San Luis Potosi, México

Resumen

En esta tesis abordamos el monitoreo de contenidos de audio. Es un problema que merece atención debido a la habilidad técnica necesaria y la importancia económica de este (pautas publicitarias). En México, la legislación obliga a la autoridad electoral a llevar a cabo un monitoreo de los anuncios de los partidos políticos. Sin embargo algunas empresas dedicadas a la tarea hacen uso de testigos humanos y otras tantas usan programas de fuente cerrada. Esa situación, junto con la de otras empresas dedicadas a resolver este problema haciendo uso también de testigos humanos ha motivado el desarrollo de un sistema donde el monitoreo en tiempo real pueda ser llevado a cabo con la menor intervención humana posible.

Nos concentramos en el monitoreo de transmisiones por internet debido al alcance de este y de la facilidad de montar una estación de radio por internet, situación que ha llevado a algunas de las emisoras de radio mas grandes, a transmitir en tiempo real sus programaciones además de haber algunas empresas de publicidad importantes que han montado sus propias estaciones sin importar que no tengan frecuencias de radio asignadas.

Para poder llevar a cabo la tarea descrita, hacemos uso de una firma acustica del estado del arte llamada MBSES (Multi-Band Spectral Entropy Signature ó Firma de Entropía Espectral Multi-Banda), que ya ha sido usada con un alto grado de éxito para tareas similares.

También hacemos uso de índices para búsquedas por similitud para poder permitir al sistema monitorear una gran cantidad de estaciones a la vez. Algunos de estos índices hacen uso de espacios métricos, el otro presentado aprovecha la alta robustez de MBSES para permitir un indexado eficiente en forma de un índice invertido.

Los resultados que obtuvimos indican que la compresión con perdida y la ecualización no resultan en degradaciones significativas para la firma MBSES. Ademas, mientras que indexar las firmas haciendo uso de indices de proximidad no resultó tan efectivo. Nuestro ultimo enfoque, el índice estilo indice invertido, se desempeño muy bien, aun con las degradaciones mencionadas al inicio de este párrafo, logra obtener el mejor tiempo con una recuperación del 100 %.

Abstract

In this thesis we address the problem of monitoring audio content. This problem deserves attention due to the technical challenge it impies and its economic importance (advertisments). In México, legislation forces elective authority to perform monitoring over political parties advertisements on broadcasts. They, however address this issue with no techincal skill at all, by using human observers to carry out this task. Such situation, along with other companies devoted to this task also using human observers encouraged me to develop a system where real time monitoring could be performed, having as least human intervention as possible.

We focus on internet broadcasts due to the reach of the internet and the ease of setting up a internet radio station which has led to some of the biggest radio broadcasters to real-time relay their air broadcasts, and some other trascendent advertisment sites to set up their own regardles of not having an air broadcast frequency.

In order to achieve the task described above, we made use of state of the art acustic fingerprint called multi-spectral band entropy signature (MBSES) which has already been used with a high degree of success on a similar task.

We also made use of some similarity search indexes in order to enable the system to monitor many anouncements at once. Some of these indexes take advantage of metric spaces, another one takes advantage of the high robustness of MBSES to allow an efficent indexing in the form of an inverted index.

The results we got indicate that losely compression and equalization are not significant degradations with MBSES. While indexing using proximity indexes turns out to not be so efective for our dataset. Our last approach o address the indexing issue, the inverted index-like performs really well regardles of the degradation stated above, we still managed to get 100 % recall.

Contenido

Dedicatoria III
Agradecimientos V
Lista de Publicaciones VII
Resumen IX
Abstract XI
Contenido XIII
Lista de Figuras XV
Lista de Tablas XVII

1. Introducción 1
   1.1. Planteamiento del Problema 1
   1.2. Antecedentes 3
      1.2.1. Testigo Humano 3
      1.2.2. Informe de transmisiones 4
      1.2.3. Metadatos Adjuntos 4
      1.2.4. Marcas de agua digitales 5
      1.2.5. Firmas de audio 7
      1.2.6. Sumario de métodos de detección 8
      1.2.7. Fuentes de audio 11
   1.3. Objetivos de la Tesis 12
      1.3.1. Objetivo general 12
      1.3.2. Objetivos particulares 12
   1.4. Descripción de Capítulos 13
   1.5. Conclusiones 13

2. Caracterización y modelado de una señal de audio 15
   2.1. Introducción 15
   2.2. Esquemas de Extracción de características para audio 16
      2.2.1. Firma de Audio “Altamente Robusta” 20
      2.2.2. Descriptores MPG7 21
      2.2.3. MBSES 23
   2.3. Conclusiones 29

3. Indexado y recuperación de elementos de una base de datos 31
3.1. Introducción . . . 31
3.2. Espacios métricos . . . 32
3.3. Estructuras de datos métricas . . . 33
3.3.1. Árbol B-K . . . 37
3.3.2. Arreglo de consulta fija . . . 42
3.4. Indice invertido . . . 43
3.4.1. Ejemplo . . . 45
3.5. Conclusiones . . . 46

4. Implementación 49
4.1. Introducción . . . 49
4.2. Captura del Audio transmitido por Internet . . . 49
4.3. Determinación de la Firma MBSES . . . 50
4.3.1. Calculo de la distancia . . . 52
4.4. Indexado . . . 54
4.4.1. Árbol B-K . . . 55
4.4.2. Arreglo de consulta fija . . . 57
4.4.3. Indice invertido . . . 57
4.5. Conclusiones . . . 60

5. Resultados 63
5.1. Introducción . . . 63
5.2. Resultados de Monitoreo fuera de linea . . . 64
5.3. Resultados de Monitoreo en linea . . . 71

6. Conclusiones 73
6.1. Conclusiones Generales . . . 73
6.2. Conclusiones particulares . . . 74
6.3. Trabajos Futuros . . . 76

Referencias 79

Lista de Figuras

1.1. Concepto básico de incrustación de marca de agua en audio 6
1.2. Figura que ilustra un esquema general de monitoreo de audio por internet . 8

2.1. Esquema general de un sistema de extracción de características 16
2.2. Imagen donde se representa el proceso de enmarcado 17
2.3. Curva Bark/Hertz, imagen de [Camarena06] 19
2.4. Diagrama del proceso de extracción de la firma de mpeg7 22
2.5. Entropigrama,imagen tomada de [Camarena06] 26
2.6. Determinación de MBSES, imagen tomada de [Camarena06] 27

3.1. Figura que muestra un anillo respecto al punto G con ancho $w{=}d_{max}$, en la imagen, se puso el anillo alrededor de todos los puntos lo mas ajustado, pero un anillo con $w{=}d_{max}$, abarcaría mas espacio. 34
3.2. En esta figura se muestran 3 anillos usando a los puntos D e I como pivotes 35
3.3. Podemos observar como se distribuyen los puntos respecto a los pivotes en los anillos de estos 36
3.4. Podemos observar al elemento q 36
3.5. Aquí ya con los anillos dibujados, podemos ubicar a los elementos mas cercanos a q 37
3.6. Primera iteración del árbol B-K 40
3.7. Árbol Burkhard-Keller completo para el ejemplo 41
3.8. Arreglo de consulta fija para los puntos de la figura 3.1 42
3.9. Ilustración del índice invertido, imagen tomada de [Haitsma02] 43
3.10. Imagen de un índice invertido 45

4.1. Determinación de MBSES, imagen tomada de [Camarena06] 51
4.2. Ilustración del buffer a través de la transmisión 52
4.3. Código fuente para el llenado de la tabla de distancias 54
4.4. Ejemplo de operaciones de corrimiento para concatenar bytes 58
4.5. Indice invertido con la modificación propuesta 59

5.1. Curva ROC para el traslape de 19/20 64
5.2. Resultados utilizando búsqueda lineal y la tabla de búsqueda 68
5.3. Resultados utilizando 3 segundos de audio 69

5.4. Resultados utilizando 5 segundos de audio 70
5.5. Resultados utilizando 7 segundos de audio 71

Lista de Tablas

2.1. Ecuaciones para aproximar la curva Bark/Hertz (z son barks, f son hertz, y los ángulos en radianes) 18

3.1. Tabla con las distancias de los puntos 39

5.1. Detecciones logradas usando un umbral de 30 % con una búsqueda secuencial 65
5.2. Diferentes traslapes y detecciones logradas usando la tabla de búsqueda . . 66
5.3. Diferentes traslapes y detecciones logradas utilizando la tabla de búsqueda . 67
5.4. Diferentes traslapes, detecciones logradas en 5 horas, y tiempo requerido . . 72

Capítulo 1

Introducción

En este capítulo introducimos el problema de monitorear una transmisión de audio por Internet y el por que resulta de mucho interés la solución a dicho problema, se plantean las ventajas de realizar el monitoreo de las estaciones de radio por internet y algunos de los enfoques que se han utilizado para abordar el problema mas general de identificación de audio.

1.1. Planteamiento del Problema

La radio es un medio de comunicación de suma importancia, usada con diferentes fines, tales como medio para proporcionar información oficial, escuchar programas con fines de entretenimiento, promocionar materiales discográficos recientemente lanzados, promocionar productos y servicios, entre otros. Era incluso una recomendación oficial en caso de desastre contar con una pequeño radio de baterías. Esta versatilidad y penetración llevó a la radio a convertirse en un buen negocio e hizo que rápidamente surgieran radiodifusoras tanto de alcance local, como de alcance nacional.

El crecimiento de la radio generó además el interés de saber que contenidos se transmitían, en que momento y cada cuando, esto por diferentes razones, aquellos que pagaban por un espacio en la radio querían estar seguros de que su mensaje había sido transmitido, empresas dedicadas a las estadísticas muchas veces deseaban saber que cosas eran emitidas y que tan a menudo, de esta forma podían establecer que tan popular resul-

taba tal o cual contenido, a veces cuando surgen desacuerdos entre una casa discográfica y una radiodifusora la primera remueve cualquier derecho de que la segunda transmitiera contenidos propiedades de la disquera. Para asegurarse de que esta restricción se cumpla, es necesario llevar a cabo el monitoreo. Incluso en el ambiente político nacional actual tiene su utilidad el monitoreo, como la radio es un medio sumamente importante para los partidos políticos a estos les interesa que sus anuncios sean transmitidos; mientras que por otra parte cuando han surgido anuncios que infringen la normatividad electoral mexicana es igualmente necesario verificar que estos no sean transmitidos esta tarea se puede completar monitoreando las estaciones en busca de anuncios vetados. Estos son algunos ejemplos de la utilidad del monitoreo de la señal de audio que emiten las estaciones de radio.

En los últimos años pareciera que la velocidad de crecimiento de la radio ha disminuido, esto puede ser verdad dependiendo de como se observe, pero hay que considerar esto, la penetración que han tenido las computadoras domésticas, que además cuentan con poder de cómputo mucho mayor que hace algunos años (y aumenta cada vez), y que cada vez son mas accesibles los servicios de internet con un amplio ancho de banda, esto ha propiciado que se generen diferentes protocolos que ofrecen el servicio de transmisión de contenidos multimedia por internet, entre ellos audio, algunos hacen alusión directa a la radio. El costo cada vez mas accesible de anchos de banda cada vez mayores ha provocado además que el proveer servicios por internet sea mas barato; incluidos aquellos mencionados. Esto ha hecho que estaciones de radio populares (por citar algunos ejemplos, “RMX”, “radioformula”, “LA Z” y otras muchas) decidan retransmitir sus emisiones por internet, con la ventaja de que existe una menor degradación del contenido ya que la señal de audio transmitida por internet no sufre degradaciones inherentes a la transmisión; lo que resulta en una mucho mejor calidad de audio, no solo eso, si no que adicionalmente han surgido “emisoras” que transmiten exclusivamente por internet, teniendo en cuenta esto, ya no resulta tan cierto afirmar tajantemente que el crecimiento de la radio ha disminuido, hay que señalar además, aunque en el sentido estricto no sean transmisiones de radio, coloquialmente son conocidas de esa forma. Estas situaciones, motivan a que este trabajo este destinado a monitorear los contenidos de estaciones que transmiten por internet, sin embargo, la parte central de cualquier trabajo como este, es lograr un monitoreo exitoso, sin importar la fuente de la

transmisión, y la parte central debería servir para cualquier aplicación que requiera identificación de audio. Es importante mencionar, que el problema puede ser resuelto de dos maneras, que se conocen en línea y fuera de línea, la diferencia de cada uno, consiste en que en el monitoreo en linea busca identificar el audio durante su transmisión, mientras que en lo concerniente a fuera de linea, se hace el monitoreo a partir de archivos donde la señal previamente transmitida ha sido guardada después de que estas fueron emitidas .

1.2. Antecedentes

Como se menciona en la introducción, el interés por monitorear contenidos transmitidos por radio no es nuevo, en esta sección de antecedentes se explican de manera breve algunos de los enfoques utilizados a la fecha, si el lector desea puede ir a la Sección 1.2.6 y simplemente revisar el resumen donde se listan los pros y contras de cada enfoque. Hay que resaltar, que algunos de estos enfoques, solo son posibles gracias al creciente poder de computo y la investigación que ha surgido tanto en protocolos de transmisión por Internet, como en estándares de transmisión por aire, que permiten enviar información adicional y no solo la señal de audio.

1.2.1. Testigo Humano

Básicamente emplea un ser humano que identifica los contenidos de las transmisiones. Pese a ser un enfoque muy antiguo, tiene un grado confiable de efectividad y es aún hoy en día es ampliamente utilizado como se indica en [Hellmuth01]. Este enfoque, si se hace en linea implicaría hacer que un ser humano escuchase las transmisiones y generase reportes al mismo tiempo, si se hace fuera de linea se podría pensar en aumentar la eficiencia al poder saltar contenidos que no son de interés (por ejemplo, si lo que se desea monitorear son anuncios, el humano podría adelantar las grabaciones durante canciones o programas). Este enfoque es fácil de poner en práctica, no se requiere hardware especializado, ni conocimientos especiales, además un humano es capaz de identificar contenidos pese a no haberlos escuchado previamente, pero tiene las desventajas de que resulta costoso. El humano es propenso a la fatiga, por lo que puede cometer errores. El costo aumenta linealmente para

varias transmisiones en el caso de monitoreo en linea ya que se tendría que contratar a una persona por transmisión.

1.2.2. Informe de transmisiones

Para identificar los contenidos de una transmisión, es posible también pedir un informe de transmisión a la emisora, este informe se puede generar antes de la transmisión o durante esta. Este enfoque expuesto igualmente por Oliver Hellmuth en [Hellmuth01] resulta realmente barato en su implementación, incluso antes era protocolario entregar este informe. Hoy en día, las estaciones además de no estar obligadas a entregar algo así, muchas veces ni siquiera están seguras de lo que se va a transmitir. Pongamos de ejemplo los programas de música a petición del publico, en estos el conductor del programa recibe llamadas telefónicas de sus escuchas y solicitan alguna canción, y enseguida el conductor del programa, debe elegir las canciones para el bloque de canciones de entre las que pidió el público. Otro inconveniente, es que en este enfoque no se analiza el contenido de la transmisión, solo el reporte, y si hubo cambios de ultimo minuto, difícilmente serán reflejados.

1.2.3. Metadatos Adjuntos

Para entender que son los metadatos adjuntos consideremos las colecciones de audio en diferentes formatos que hay, almacenadas en las computadoras. En prácticamente cualquier reproductor de audio; este busca las etiquetas que tienen los archivos y las lee para mostrar información acerca del contenido que se está reproduciendo. A esto se refieren los metadatos, información adicional a la señal de audio, enviada de manera adjunta, como menciona Oliver Hellmuth en [Hellmuth01] . Las ventajas que hay al leer los metadatos es que resulta un tanto trivial, no se requiere mucho poder de cómputo, ni comparar contra otros archivos. Una de las desventajas de este enfoque es que en transmisiones por aire regulares no se cuenta con espacio para metadatos, aunque en la transmisiones de radio digital y por internet si se pueden poner metadatos, la realidad es y será al menos por algunos años, que las emisoras de radio no ponen metadatos a sus transmisiones, y las que lo hacen por internet (el caso de radioformula, entre las emisoras que se revisaron), la etiqueta con la que cuentan solamente tiene información sobre el nombre del programa que

están emitiendo actualmente, y no dan información sobre los anuncios que ocurren durante el programa, ni las canciones. Por si esas limitaciones no bastaran, los metadatos pueden ser fácilmente alterados, el hecho de que no se hace un análisis verdadero de la señal de audio, provoca que se pierda confiabilidad.

1.2.4. Marcas de agua digitales

Poner una marca de agua consiste en incrustar información adicional a la ya contenida en el objeto al que se incrustara. Tomemos por ejemplo los noticiarios televisivos, en estos aparece en una esquina, el logo de la cadena que lo produjo y emitió, esto ya es en si un ejemplo de marca de agua, en el caso mencionado, es relativamente fácil deshacerse de la marca de agua, lo que hacen cuando realmente quieren que no sea fácil remover de la marca de agua (por ejemplo para un reportaje muy exclusivo), ponen una matriz degradada del logo sobre toda la imagen, de esa forma aunque corten una parte de la imagen la marca persistirá. Estas son marcas de agua perceptibles fácilmente y aplicadas a imágenes. Ese mismo concepto es usado de manera habitual en audio, en ocasiones a lo largo de una canción destinada a ser transmitida por radio, se incrusta algún mensaje que se repite varias veces a lo largo de la canción, este mensaje, usualmente contiene información sobre la canción (nombre de la canción, quién canta, disco al que pertenece), la empresa distribuidora del material, y la estación autorizada a transmitirlo. Y aunque en ambos casos dicha información sirve a su propósito resulta a veces molesto poder percibir la marca de agua, además de que la hace mas vulnerable a ataques, se podría pensar en el caso donde se coloca la matriz de logos, simplemente tapar los logos con el mismo color de estos, en el caso de la transmisión, de igual forma se puede afectar la integridad de la marca de agua editando la canción en los intervalos donde aparece. Lo mencionado anteriormente resulta muy básico y fácil de atacar, de tal manera que se pudiera pensar que no es tan útil una marca de agua. En realidad el proceso mas útil resulta un tanto mas complejo, de este punto en adelante nos vamos a referir mas específicamente al proceso de incrustar marcas de agua en señales de audio que es lo que nos resulta de mayor interés. A diferencia de la encripción o compresión de audio, una marca de agua no debe de depender de algún formato. Para poder viajar junto con el contenido al que protege, la marca de agua debe de ser portada

por el contenido en si. Incrustar una marca de agua es una modificación activa a la forma de onda de la señal. Debido a esto, el audio se convierte en portador de un mensaje, sin importar el medio en el que este vive, como se menciona en [Metois99] En el contexto del procesamiento de audio estándar y los sistemas de transmisión el audio puede pasar través de varios escenarios que degradan la señal. Una marca de agua debe de persistir a tales manipulaciones, por lo tanto es imperativo que la tecnología de marcado no confíe solamente en porciones del espectro auditivo que resulten perceptiblemente menos relevantes. Esto debe de hacerse de una manera que no degrade el valor del contenido de la señal y el proceso de incrustación no debe de dejar degradaciones perceptibles. Una marca de agua en audio enfrenta el desafío de abrir un canal inaudible y confiable de datos dentro de la parte mas relevante del espectro auditivo [Metois99] Al lograr el objetivo de enviar información incrustada en una marca de agua se abre la posibilidad de recibir mensajes, que podríamos considerar similares a los metadatos, sin la necesidad de un canal adicional para estos, ni infraestructura diferente a la ya existente. El concepto básico de incrustar una marca de agua en audio fue propuesto en [Neubauer98, Haitsma00] El proceso se podría explicar de

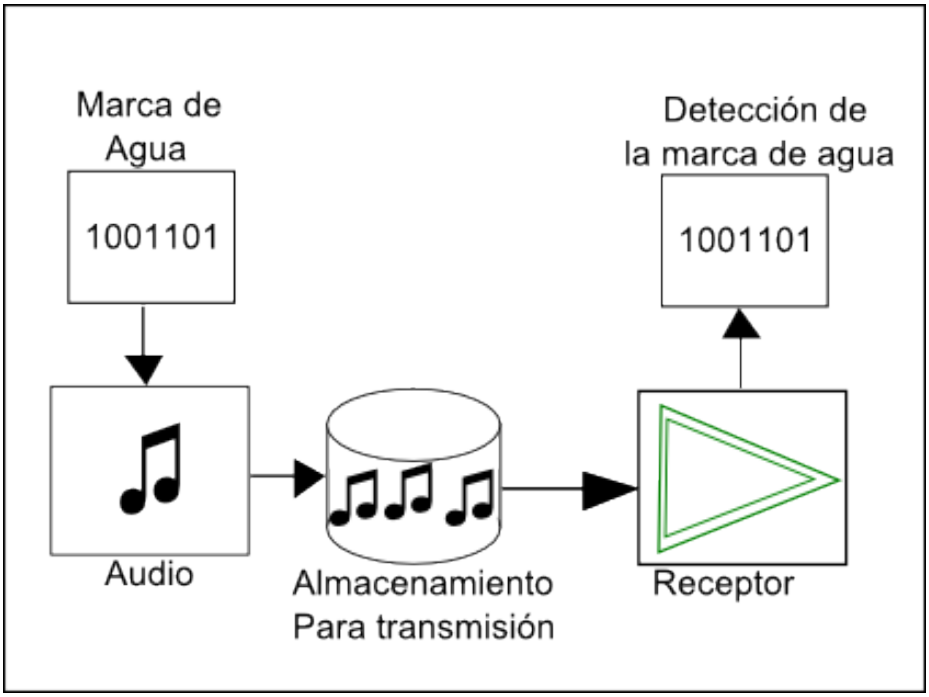

Figura 1.1: Concepto básico de incrustación de marca de agua en audio

esta manera y se ilustra en la Figura 1.1, la parte encargada de incrustar la marca, lee la señal portadora (el audio a transmitir) y el mensaje que se desea incrustar para poder calcular la marca de agua. El contenido marcado es almacenado para ser transmitido y utilizado de manera como se haría con la señal sin la marca. La señal puede ser reproducida por cualquier reproductor de audio estándar, se puede agregar una etapa de detección de la marca de agua para poder recuperar el mensaje incrustado. La incrustación de la marca de agua requiere que se tenga acceso al contenido antes de que este sea distribuido, aunque tiene la ventaja de poder incrustar identificadores específicos en instancias de la misma señal,es decir, dos señales de audio que son perceptualmente idénticas pueden tener diferentes marcas de agua, esto es particularmente útil para detectar usos no autorizados. La marca de agua como ya se mencionó, puede incluir la información que se desea conocer respecto a la señal, o bien un identificador que permita recuperar la información deseada de una base de datos [Hellmuth01].

Aunque existen formas de marcado de agua efectivas y resistentes a degradaciones, para poder implementar de manera efectiva un esquema de monitoreo basado en marcas de agua, además de tener acceso previo al audio a transmitir, se debe de tener cuidado en la forma en que se manejan las transiciones de contenidos, esto debido a que la confiabilidad de detección se deteriora en los limites de los contenidos (el limite entre un contenido marcado y otro contenido que pudiera ser o no marcado)[Nakamura02].

1.2.5. Firmas de audio

El modelado de una firma de audio se hace mediante la extracción de características únicas, propias de una señal. Basado en estas características, una señal puede ser identificada y a diferencia de las marcas de agua no se requiere acceso previo al audio y el contenido de la señal no se altera durante el calculo de la firma. En un esquema de monitoreo de audio es necesario extraer una muestra del audio que se esta transmitiendo, acceder a una base de datos de firmas precalculadas, para saber si el audio muestreado, coincide con el audio que nos interesa monitorear, de ser así, debe ser reportada la ocurrencia.

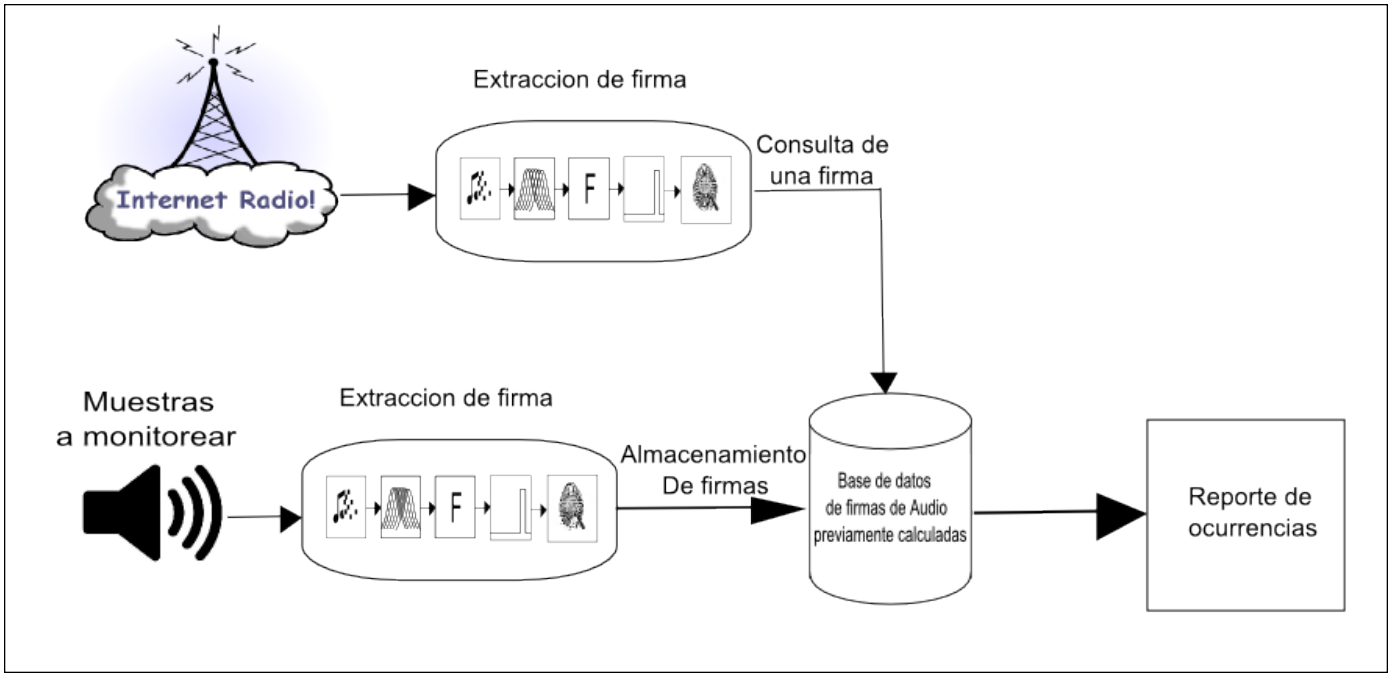

Figura 1.2: Figura que ilustra un esquema general de monitoreo de audio por internet

Observe la Figura 1.2 para monitorear se debe de tener una base de datos con firmas determinadas previamente de los contenidos que deseamos monitorear, enseguida y ya como parte del proceso de monitoreo, se debe de tener una fuente de audio, en este caso una transmisión por internet, a segmentos de esa transmisión que ya hayan sido transmitidos, se les debe de extraer la firma, estos segmentos de firma, deberán ser comparados con la base de datos, y en caso de que exista una firma que se parezca lo suficiente, se debe de reportar una coincidencia. En los artículos [Camarena09, Nieuwenhuizen10, Hellmuth01] se presenta el uso de firmas de audio para el monitoreo fuera de linea de transmisiones de radio, en algunos casos, como en [Nieuwenhuizen10], se busca el audio sin degradación alguna mientras que en [Camarena09, Hellmuth01] existe algún tipo de degradación.

1.2.6. Sumario de métodos de detección

Podemos resumir los puntos fuertes y las debilidades de cada uno de los posibles métodos de monitoreo mencionados:

Testigo humano

- **Pros**
  - Fácil de llevar a cabo
  - No se requiere hardware adicional ni especial
- **Contras**
  - Muy caro para monitoreo 24 horas y 7 días a la semana
  - Suceptible a errores por diferentes factores inherentes al testigo humano
  - El costo aumenta mucho para varios canales

Informe de transmisión

- **Pros**
  - Su implementación resulta trivial
  - Su costo resultaría muy bajo
- **Contras**
  - No se realiza un verdadero análisis de contenidos
  - Resulta muy poco confiable

Metadatos adjuntos

- **Pros**
  - Su implementación resulta trivial
  - Resulta muy barato
- **Contras**
  - No se hace un verdadero análisis de contenidos
  - Resulta poco confiable
  - No todas las transmisiones están preparadas para transmitir metadatos

Marcas de agua

- **Pros**
  - Se hace un verdadero análisis de contenidos
  - Provee una prueba confiable de que el audio ha sido transmitido
  - Se puede identificar una instancia en particular de audio
  - Se puede escalar para monitorear varios canales
  - No es estrictamente necesario el acceso a una base de datos
- **Contras**
  - Se requiere acceso previó al audio a transmitir para insertar la marca de agua
  - Produce una alteración en la señal lo cual puede comprometer la calidad del audio

Firmas de audio

- **Pros**
  - Se hace un verdadero análisis de contenidos
  - Provee una prueba confiable de que el audio ha sido transmitido
  - Se puede escalar para monitorear varios canales
  - No altera el contenido de la señal
  - No se requiere acceso previó al audio a transmitir
- **Contras**
  - Se requiere acceso a una base de datos de firmas pre-calculadas
  - No se pueden identificar instancias particulares de audio (copias especificas)

La lista de fortalezas y debilidades nos permite pensar que las marcas de agua, y las firmas de audio son los mejores enfoques. Esto resulta verdadero en el contexto actual

de las transmisiones, siendo fundamental una desventaja de las marcas de agua, el hecho de requerir el acceso previo al audio que se transmitirá, hace que las firmas de audio resulten mas atractivas. Esto no significa que las marcas de agua no sean un enfoque competente. En esta tesis nos enfocaremos en el uso de firmas de audio, y en el Capitulo 2 discutiremos sobre estas.

1.2.7. Fuentes de audio

Hasta este momento solo se ha hablado de los diferentes enfoques usados para monitorear audio, sin ponerle mucha importancia a la fuente de este. También se mencionó la existencia de servicios orientados a la transmisión multimedia por internet. Dichos servicios se valen de protocolos para establecer las comunicaciones entre quien transmite el audio (servudir) y quién recibe el audio (cliente). Entre los protocolos para llevar a cabo esta tarea, destacamos el protocolo de transmisiones en tiempo real (RTSP por sus siglas en inglés), el protocolo de mensajes en tiempo real (RTMP) y un un software de servicios que además integra modificaciones al protocolo de transferencia de hipertexto (HTTP) para usarlo como medio de transporte llamado Shoutcast. Las especificaciones de los protocolos RTSP y RTMP están disponibles al público. La diferencia central entre ambos protocolos, es que RTSP fue concebido como un protocolo para controlar las transmisiones en sistemas de entretenimiento mientras que RTMP era un protocolo propietario (hasta hace relativamente poco que se liberaron las especificaciones de este) diseñado para controlar las transmisiones de internet entre un reproductor “flash” y un servidor. Encontrar información acerca de como funciona shoutcast es un poco mas difícil, una explicación breve se encuentra en [Jay06] este pequeño borrador hecho basado en ingeniería inversa contiene la suficiente información para poder implementar un cliente básico que funcione con el servicio que provee un servidor shoutcast, el interés de utilizar este servicio por encima de los otros dos, se debe a la facilidad de encontrar emisoras de radio debido a que en el “directorio shoutcast” están recopiladas y organizadas mas de 50,000 [Nullsoft99] estaciones de radio que usan este servicio, entre estas podemos encontrar estaciones de radio que transmiten su programación habitual de manera integra por este medio.

1.3. Objetivos de la Tesis

1.3.1. Objetivo general

El objetivo general de esta tesis es llevar a cabo monitoreo de audio en línea, utilizando firmas de audio con algunas firmas que han demostrado ser efectivas para el monitoreo fuera de linea como las presentadas en [Camarena09, Nieuwenhuizen10, Hellmuth01]. Se han de analizar los puntos fuertes de cada una, y se decidirá en base a ello cual será adaptada para dicha tarea. Hay que señalar que en el monitoreo en línea se necesita además una fuente de audio, para ello habrá que buscar la forma de obtener una transmisión de audio en vivo, que proporcione un flujo de transmisión.

1.3.2. Objetivos particulares

- Implementar un cliente de funcionalidad básica para el servicio Shoutcast desde el cual podamos proveer el audio en tiempo real para nuestro sistema.

- Implementar un sistema que permita extraer una firma de audio de manera continua al audio obtenido con el cliente del punto anterior.

- Recopilar una base de contenidos (anuncios) sujetos a las degradaciones del canal para poder buscarlos en emisiones subsecuentes.

- Optimizar algunos parámetros que pueden variarse en las firmas de audio, para poder saber cual se ajusta mejor a las necesidades de este desarrollo particular.

- Implementar esquemas que permitan recuperar candidatos posibles de entre la base de contenidos de manera eficiente, entre estos, búsquedas aproximadas.

- Evaluar el desempeño del sistema en medida de lo posible con sistemas de monitoreo que permiten el monitoreo fuera de linea.

1.4. Descripción de Capítulos

En el Capítulo 2 se habla a detalle del proceso de extracción de las firmas de audio, parte central para la solución del problema planteado, se profundiza en la explicación sobre la firma particular elegida y se da el por qué a dicha elección.

En el Capítulo 3 se habla de algunos métodos a utilizar para la organización de las firmas de audio obtenidas, de tal forma que se lleve a cabo una búsqueda mas eficiente en vez de una búsqueda secuencial de cada firma en la base de datos.

En el Capítulo 4 se explica como se unen las partes para formar el sistema, se realizan experimentos para determinar la efectividad del sistema y se presentan los resultados obtenidos.

En el Capítulo 5 se presentan las conclusiones a las que se llegan desprendidas de los resultados del Capítulo 4, y se exponen diferentes ideas para trabajo futuro.

1.5. Conclusiones

En este capítulo se dió una breve introducción al problema a resolver, se mencionan algunos de los beneficios obtenidos con su solución, y se plantean algunos enfoques utilizados para resolverlo, además de los alcances que se esperan del desarrollo.

De los enfoques mencionados anteriormente, existen dos que resultan baratos y sencillos de implementar, que son el recibir un informe de la transmisión, y el leer los metadatos. En ambos casos estamos sujetos a creer en información que pudiera resultar no ser confiable, y en ninguno se hace un verdadero análisis de lo que se transmite, esto ultimo es quizá lo que mas pesa para concluir que no son adecuados para nuestro propósito. En tanto usar un testigo humano, marcas de agua y firmas de audio si llevan a cabo un análisis de la señal y su contenido, y sus resultados pueden ser muy confiables, siendo en el caso del testigo humano el menos confiable, ya que un humano es propenso al cansancio, sin mencionar que si deseáramos monitorear mas canales, el costo aumentaría mucho. Las marcas de agua son un enfoque plausible, escalable y confiable, los problemas que se pueden presentar al llevar a cabo el monitoreo con marcas de agua han sido resueltos satisfactoriamente, y posee características deseables. Pero existe un problema, se requiere acceso al audio que va a ser

transmitido, para poder llevar a cabo el proceso de marcado, este detalle hace que pese a ser un enfoque inteligente,hace que no sea viable en el contexto actual de las transmisiones. Lo que nos lleva a concluir que lo que mejor se adapta a las necesidades actuales son las firmas de audio, por lo que en el siguiente capitulo, mencionaremos algunas propuestas de firmas de audio exitosas.

Capítulo 2

Caracterización y modelado de una señal de audio

2.1. Introducción

Como se menciona en la introducción, la tarea de monitorear estaciones de radio se puede llevar a cabo de diferentes formas, entre las cuales destacamos el uso de extracción de características y modelado, lo que resulta de este proceso se le conoce como "Firma de Audio". En este proceso se busca extraer características que sean únicas de la señal procesada, de tal forma que dichas características no se encuentren en señales que sean diferentes, y además que persistan a pesar de que la señal sea degradada, a esta propiedad se le conoce como "robustez" y es indispensable contar con ella. Esta característica no es la única deseable en una firma de audio. Existen otras propiedades deseables como la facilidad de determinación de la firma, esto se refiere a que el proceso de extracción y modelado no debería ser computacionalmente costoso; facilidad de comparación, que se refiere a que sea eficiente la manera en que se determina que tanto se parece una firma con otra. Las señales, específicamente en este caso de audio, suelen tener una duración de varios segundos. Otra característica deseable de una firma de audio es que se puedan identificar fragmentos mas pequeños de la señal, como componentes de la misma, es decir, que con un pequeño fragmento de la señal, seamos capaces de determinar a que señal pertenece ese fragmento, a

esta propiedad se le conoce como "granularidad". No menos importante, es que la extracción y modelado resulten en una firma compacta, es decir, que el resultado tenga un tamaño varias veces menor a la señal original. Finalmente podemos hablar de escalabilidad, que se refiere a que la firma debe de poder operar entre grandes colecciones de firmas, esta última característica esta condicionada a que la firma sea compacta, que el calculo de su distancia resulte computacionalmente eficiente y una buena técnica de indexado. Existen numerosas propuestas para la extracción de características y obtención de una firma de audio y a continuación mencionaremos algunas.

2.2. Esquemas de Extracción de características para audio

Muchos esquemas de extracción de características comparten mucho con el diagrama mostrado en la Figura 2.1

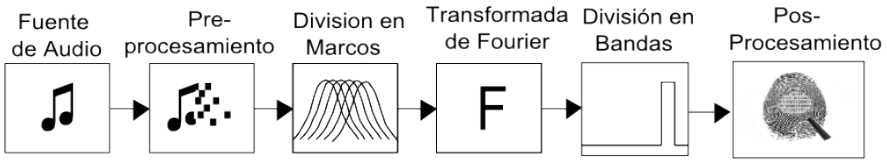

Figura 2.1: Esquema general de un sistema de extracción de características

En la Figura 2.1 se observa una etapa de pre-procesamiento de la señal, generalmente consistente en convertir la señal a sonido monoaural, es decir de un solo canal, sin importar el número de canales de la señal original, esto se logra sumando las muestras de la señal, y dividiendo entre el número de canales de esta, aunque este no el único pre-procesamiento que existe. La siguiente etapa en el extracción y modelado de audio consiste en dividir la señal en partes mas pequeñas a de un tamaño elegir, a estos fragmentos se les conoce como marcos. Adicionalmente es común elegir un tamaño de traslape, menor al tamaño de un marco, este tamaño de traslape indica que tanta información del marco anterior posee el nuevo marco.

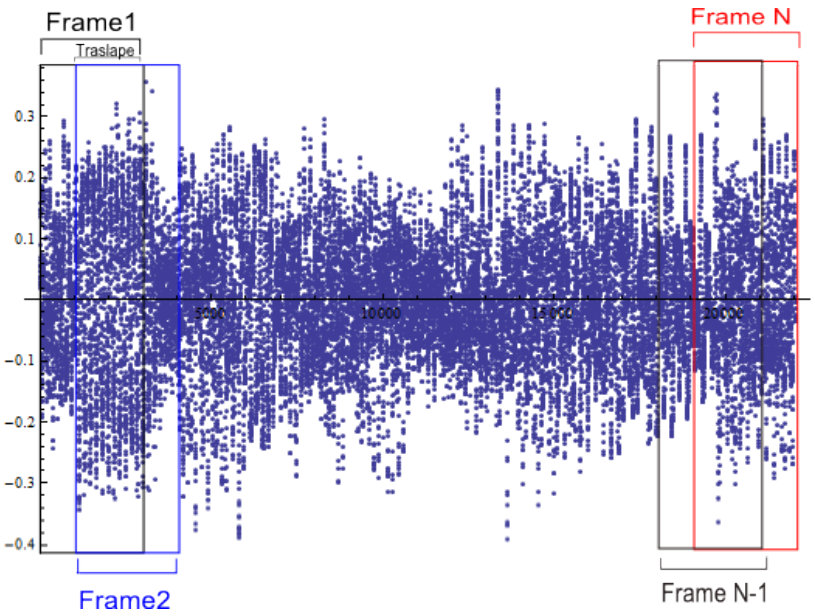

Figura 2.2: Imagen donde se representa el proceso de enmarcado

El efecto de dividir en marcos la señal, ilustrado en la Figura 2.2. Esto se logra aplicando la ventana cuadrada de manera sucesiva, esta ventana está definida por la Ecuación (2.1)

$$w(t) = \begin{cases} 1 \text{ si } t \in [0, T] \\ 0 \ Resto \end{cases} \tag{2.1}$$

T Es el tamaño máximo del marco, la señal truncada $s_h(t) = s(t) * w(t)$, $s(t)$ es la señal original. Es común utilizar una ventana diferente a la ventana cuadrada. Lo siguiente es aplicar la transformada discreta de Fourier, esta nos permite analizar los componentes de frecuencia de las señales. El resultado de aplicar la transformada discreta de Fourier, resulta en una señal del mismo tamaño que la original, pero con valores complejos. Conviene recordar que según el teorema de muestreo de Nyquist-Shanon $F_s \geq 2F_{max}$, esto significa, que la frecuencia de muestreo debe de ser el doble de la frecuencia máxima que deseamos poder representar, tomando esto en cuenta, podemos saber que justamente a la mitad de la señal, se encuentra la frecuencia máxima obtenida, que es de $11025Hz$, si tenemos en cuenta el teorema de Nyquist y recordamos que la frecuencia de muestreo de ese ejemplo es de $22050Hz$. Considerando lo anterior, podemos determinar a que frecuencia pertenece

cada coeficiente devuelto por la transformada de Fourier con la Ecuación (2.2)

$$f_i = (i * F_s)/N \tag{2.2}$$

Donde:

f_i Es la frecuencia a la cual pertenece el coeficiente.

i Es la posición que ocupa el coeficiente en la serie.

F_s Es la frecuencia de muestreo.

N Es la cantidad de coeficientes que hay en la señal.

La Ecuación (2.2) es importante para poder hacer la división en bandas, paso que se explica a continuación.

Autor	Ecuación
Tjomov (1971): (A1)	$z = 6.7senh^{-1}(\frac{f-20}{600})$
Fourcin et al. (1977): (A2)	$z = 6senh^{-1}(\frac{f}{600})$
Schroeder et al. (1979): (A3)	$z = 7senh^{-1}(\frac{f}{650})$
Terhardt (1979): (A4) :	$z = 13.3tan^{-1}(\frac{0.75f}{1000})$
Terhardt (1979): (A5) :	$z = 13.3tan^{-1}(\frac{0.75f}{1000})$
Zwicker y Terhardt (1980) (A6):	$z = 13tan^{-1}(\frac{0.76f}{1000}) + 3.5tan^{t}an{-}1(\frac{f}{7500})^2$
Zwicker y Terhardt (1980) (A7):	$z = 8.7 + 14.2Log_{10}(\frac{f}{1000})$
Traunmüller (1983, 1988, 1990): (A8)	$z = \frac{26.82f}{1960f} - 0.53$
Corrección en baja frecuencia	si $z < 2, z' = z + 0.15(2 - z)$
Corrección en alta frecuencia	si $z < 20.1, z' = z + 0.22(z - 20.1)$

Tabla 2.1: Ecuaciones para aproximar la curva Bark/Hertz
(z son barks, f son hertz, y los ángulos en radianes)

La división en bandas consiste en agrupar las frecuencias de manera que una banda contenga un intervalo de frecuencias determinado. De primera mano podríamos pensar en

agrupar las frecuencias de manera lineal (por ejemplo, decir que una banda es cada 100 hz, o cada 200 hz), sin embargo, una distribución así haría pensar que la señal original tiene las frecuencias en la misma proporción y que todas contribuyen por igual en la señal, eso desde el punto de vista de la señal, desde el punto de vista de quién la escucha, implicaría que el oído es capaz de escuchar por igual todas las frecuencias, lo cual no es cierto, en lugar se utiliza una escala logarítmica, dos escalas comunes son la escala de Mel y la escala de Bark. No existe una formula única para la conversión de hertz a dichas escalas, a en la tabla 2.1 presentamos una compilación de formulas que se han propuesto para conversión de Hertz a Barks, dicha tabla se usa para apróximar la curva mostrada en la Figura 2.1

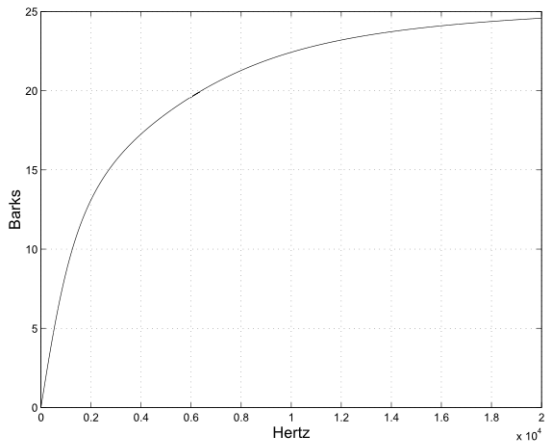

Figura 2.3: Curva Bark/Hertz, imagen de [Camarena06]

En la Figura 2.3 , podemos ver la curva de la relación que existe entre la escala de bark y la frecuencia en hertz, imagen tomada de [Camarena06]. Este es solo un ejemplo de una escala, mencionamos este ejemplo en particular por que esta escala se usa en dos de los tres esquemas que mencionaremos para obtener firmas de audio. El siguiente paso en el proceso de extracción de características y modelado de una firma de audio es propio de cada firma de audio,así que presentaremos algunos de los enfoques mas populares.

2.2.1. Firma de Audio "Altamente Robusta"

La primer propuesta de extracción de firma de audio que mencionaremos fue propuesta por Japp Haitsma en [Haitsma02], la podemos dividir en dos partes, la extracción de características y modelado de la firma y una propuesta de búsqueda para la misma, en la primera parte es en la que nos enfocaremos en este capitulo, hablaremos se la búsqueda en el siguiente capítulo.

Dicha propuesta, sigue el esquema general planteado anteriormente, toma marcos de 370 milisegundos, y hace un traslape de 31/32 de marco, lo cual representa 358.4 milisegundos, previó a la división en bandas, se calcula el valor absoluto de las frecuencias. Una vez calculado, se hace la división en bandas, tomando las frecuencias de 300 a 2000 hz. Se seleccionan 33 bandas, utilizando la escala logarítmica de Bark (no se debe de confundir con las bandas criticas, que son 24). Una vez que se han dividido los marcos en bandas, se procede a calcular la energía de cada banda con la siguiente ecuación:

$$E = \sum_{i=0}^{N} x_i^2 \tag{2.3}$$

Donde:

E: Es la energía del marco.

N: Es el tamaño del marco en muestras.

x_i: Es una muestra del marco.

A la representación compacta de un marco (el marco después de la extracción de características y el modelado) se le llamara "sub firma". Con esa ecuación se calcula la energía de las 33 bandas, sin embargo este no es todo el proceso, lo único que se conserva es el signo de la segunda derivada, primero respecto a la frecuencia, y luego respecto al tiempo (se conserva solo un 0 ó 1), esta binarización se lleva a cabo de la siguiente forma. Si denotamos la banda de energia m, del marco n como $E(n, m)$y el bit m-ésimo de la sub-firma del marco n como $F(n, m)$. Los bit de la sub-firma están definidos con la Ecuación (2.4)

$$F(n,m) = \begin{cases} 1 \text{ si } E(n,m) - E(n,m+1) - (E(n-1,m) - E(n-1,m+1)) > 0 \\ 0 \text{ si } E(n,m) - E(n,m+1) - (E(n-1,m) - E(n-1,m+1)) \leq 0 \end{cases} \quad (2.4)$$

De esta manera tendremos 32 bits por cada marco, a esto es a lo que se le llama sub-firma. Aunque una subfirma no es suficiente para identificar audio en [Haitsma02] se propone el uso de 256 marcos ó sub-firmas para identificar un contenido, que son alrededor de 3 segundos de audio.

Para determinar si dos firmas corresponden al mismo audio, se necesita una medida que determine que tan parecidas son entre ellas, en este caso, y como cada marco son 1's ó 0's se utiliza la distancia de Hamming, que consiste en comparar 2 cadenas binarias y contar el número de bits diferentes entre estas. Si este número de bits diferentes es menor a un umbral determinado entonces se dice que corresponde al mismo audio. Adicionalmente en [Haitsma02], proponen un sistema para organizar las firmas de audio de tal forma que se facilite la búsqueda en bases de datos de gran tamaño que se mencionara en el siguiente capítulo.

2.2.2. Descriptores MPG7

El conjunto de descriptores incluidos en el standard MPEG7 para contenidos audiovisuales incluyen una firma de audio que permite hacer la comparación de dos señales. Esta a su vez utiliza varias instancias de un descriptor definido por el mismo estándar, descritos en [Group01].

El descriptor mencionado en el estándar MPEG7 se basa en la "Planitud espectral", que es el otro descriptor definido en el estándar. El descriptor de planitud espectral describe que tan plano resulta el espectro de frecuencias dado un numero de bandas de frecuencia. El proceso de extracción de esta firma se ilustra en la Figura 2.4

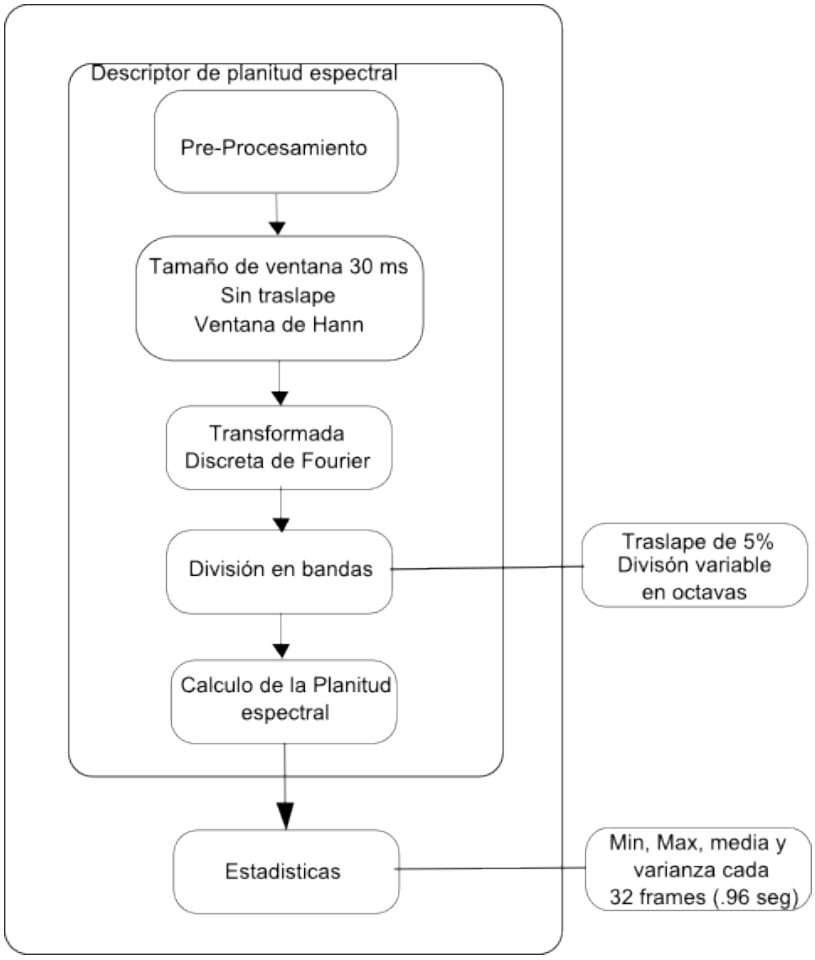

Figura 2.4: Diagrama del proceso de extracción de la firma de mpeg7

La señal después de la transformada discreta de Fourier se divide en bandas en frecuencia y es de resolución configurable, el máximo número de bandas es 24, y la resolución máxima es de 1/4 de octava. El rango de frecuencias que se puede tomar en cuenta también es configurable y el rango mas amplio va de 250 Hz a 16 KHz. Hay un traslape fijo de 5 porciento entre bandas. El tamaño de frame está fijo en 30 milisegundos y no hay traslape entre frames. La ventana de Hann se aplica a cada frame. Cada 32 frames se calcula el mínimo, el máximo, la media y la varianza de la medida de planitud espectral (a la que nos referiremos como SFM de aquí en adelante) por banda, entonces un vector se añade a la firma cada .96 segundos. [Camarena06]

La SFM es una característica relacionada con el aspecto tonal de una señal.La SFM está definida como la relación de la media geométrica y la media aritmética de los coeficientes del espectro de potencias. La SFM para una banda b con ancho de banda n_b se calcula con la siguiente formula:

$$SFM_b = \frac{\left[\Pi_{i=1}^{nb} c\left(i\right)\right]^{\frac{1}{nb}}}{\frac{1}{nb}\Sigma_{i=1}^{nb} c\left(i\right)} \tag{2.5}$$

Donde c es el vector donde los coeficientes del espectro de potencia son almacenados.

La SFM reporta valores entre 0 y 1 . Dichos valores se almacenan en archivos XML. Los mínimos y máximos calculados sirven para delimitar las búsquedas en la colección, mientras que la media y la varianza sirven para determinar que tan cerca esta un individuo x de un individuo y, usando la distancia de Mahalanobis como se muestra en la siguiente ecuación:

$$D\left(x,y\right) = \sqrt{\Sigma_{i=1}^{Bandas} \frac{\left(\mu x_i - \mu y_i\right)^2}{\sigma x_i + \sigma y_i}} \tag{2.6}$$

Esta propuesta de firma de audio resulta robusta, y en [Hellmuth01] se reportó fue usada en un problema similar al que se trata, sin embargo, la siguiente propuesta es la que se ha reportado con mejores resultados en [Camarena06], siendo mas resistente a degradaciones, además de poseer otras características deseables.

2.2.3. MBSES

La firma MBESES es una firma propuesta por José Antonio Camarena y Edgar Chávez en [Camarena06]. Dicha firma se basa en un concepto conocido como “Entropía”. El concepto de entropía apareció por primera vez en termodinámica acuñado por Rodolf Clasius en el año de 1865. Tiempo después, en 1870 Ludwing Boltzman dió un sentido estadístico a la definición de entropía.

No fue hasta 1949, en [Shannon49] que Claude Shanon ligo el concepto de entropía de Boltzman con información. El contenido de información de un mensaje, desde el punto de vista de Shanon, es proporcional a cuanto te sorprende cuando lo lees. Esto quiere decir, que un mensaje tendrá mayor información cuanto menor sea su probabilidad de ocurrir. La investigación de Shanon pretendía medir que tanta información podía ser enviada y recuperada cuando esta llegase a su destino, tomando en cuenta las posibles degradaciones que esta sufre en el camino. Pongamos como ejemplo el lenguaje, palabras como "que", "el", "a", que resulta muy probable su aparición aportan poca información al contenido del mensaje, mientras que palabras "niño", "corre" aportan una mayor cantidad de información a la vez que su probabilidad de aparecer es menor. De un texto podríamos borrar algunas de las palabras que son mas frecuentes, y afectaría muy poco su comprensión. El concepto de información es demasiado amplio para englobarlo en una sola definición, sin embargo el concepto de entropía tiene propiedades que cumplen con la noción intuitiva de lo que debería ser una medida de la información [Cover91] .

Entropía

La entropía resulta idónea para medir la información, retomemos el ejemplo del lenguaje, si calculamos la entropía del mensaje cuando faltan palabras de aquellas que resultan mas frecuentes, resulta en una variación mas pequeña respecto al mensaje completo, que si borrásemos palabras menos frecuentes. La entropía se calcula usando la formula de Boltzman.

Sea Y una variable discreta aleatoria con alfabeto β y una función de distribución de probabilidad $p(y) = PrY = y, y \in \beta$ podemos definir entonces la entropía H(Y) con la Ecuación (2.7)

$$H(Y) = -\sum_{y \in \beta} p(y) \log p(y) \tag{2.7}$$

La Ecuación 2.7 Es la utilizada para calcular la entropía del alfabeto β utilizando la probabilidad que tienen los símbolos que lo componen de ocurrir.

Para calcular la versión continua de la entropía, llamada "entropía diferencial" se utiliza la ecuación (2.8)

$$H(Y) = -\int_{-\infty}^{\infty} p(y) \log p(y) \tag{2.8}$$

La base del logaritmo es 2 cuando se trata de la entropía de Shanon, y esta se expresa en bits. Por ejemplo, la entropía de una moneda al aire es de 1 bit. Se usa por convención $0 \log(0) = 0$. [Cover91]. Si en lugar de usar logaritmo base 2 se utiliza base exponencial, entonces la entropía sera medida en "nats", la base del logaritmo no importa, en tanto se especifique, siempre y cuando tomemos en cuenta que la medida de entropía estará expresada en unidades diferentes, por ejemplo, la entropía calculada con la base exponencial se expresa en nats. La entropía en una señal es entonces una medida de que tan impredecible es esta , si la señal es fija su entropía sería 0, y siempre sabríamos que valor toma en cualquier momento, en el caso contrario, si se tiene una distribución de probabilidad uniforme, la entropía es máxima [Camarena06].La entropía a resultado una cuantificación interesante y útil del contenido de información en compresión, por ejemplo, se utiliza como cota inferior para determinar la cantidad mínima de bits requeridos para codificar mensajes [Huffman52], otras aplicaciones del concepto de entropía de Shanon en computación como lo indica la revisión realizada en[Camarena06] incluyen el uso de entropía cruzada para la identificación de imágenes, para determinar la velocidad de cuadros deseable para el análisis de voz y para determinar el tipo de compresión preferida para una señal de audio.

Hasta ahora solo se ha mencionado un poco acerca de que significa la entropía y como se calcula, pero no se ha planteado como usarla como característica perceptual para construir firmas de audio. El primer enfoque usado en [Camarena06] fue usar firma de entropía denominada como TES (Time-Domain Entropy Signature). La gran ventaja de esta firma, era que al ser calculada en el dominio del tiempo, resultaba sumamente económico su calculo, pues no requería la aplicación de DFT. Para calcular esta firma, se construye un histograma con una cantidad de muestras de la señal de audio, dicho histograma es utilizado en la formula 2.7 para poder calcular un valor de entropía, el histograma se actualiza sacando una muestra y actualizando con otra, y se calcula de nueva cuenta la entropía.Ademas de

la facilidad del calculo, resulta en una firma de audio compacta, pero no resulta robusta a degradaciones por ruido o ecualización como lo reporta [Camarena06]. En la siguiente sección, hablaremos de como se extrae la firma basada entropía espectral multi-banda.

Determinación de la firma MBSES

La firma de entropía espectral mutli-banda busca modelar el contenido de información de una señal tal como lo percibe el oído humano, para lograr esto, se auxilia de la escala de Bark. Dicha escala define 25 bandas criticas, cada una de ancho de banda de un Bark exactamente. Se desprecia la ultima banda que corresponde a las frecuencias de 15.5 a 20 KHZ. Si la entropía de los coeficientes espectrales correspondientes a una banda critica k es calculada para cada frame de la señal de audio, se obtiene una secuencia de valores de entropía. Podemos denotar esta secuencia como $SE_k(t)$ para $t = 0, 1, 2..., N-1$ donde N es el número de frames en la señal de audio [Camarena06].

Para cada frame entonces, se obtienen 24 valores de entropía (uno por cada banda). La secuencia de vectores correspondientes a un fragmento corto de audio de unos pocos segundos da como resultado una matriz de 24 filas y un número de columnas dependiente de la duración del fragmento. Esta matriz puede ser mostrada como una imagen donde el eje horizontal represente el tiempo, el eje vertical represente la frecuencia, y los niveles de gris representen los niveles de entropía. Se llaman a estas imagenes "Entropigramas". En la figura 2.5 se muestra un entropigrama de 5 segundos. [Camarena06]

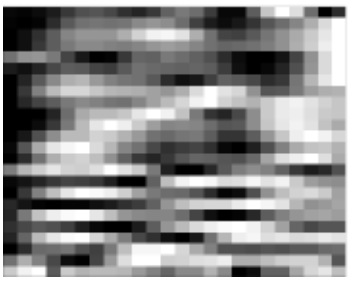

Figura 2.5: Entropigrama,imagen tomada de [Camarena06]

El esquema utilizado para el calculo de la firma MBSES se muestra en la Figura 2.6

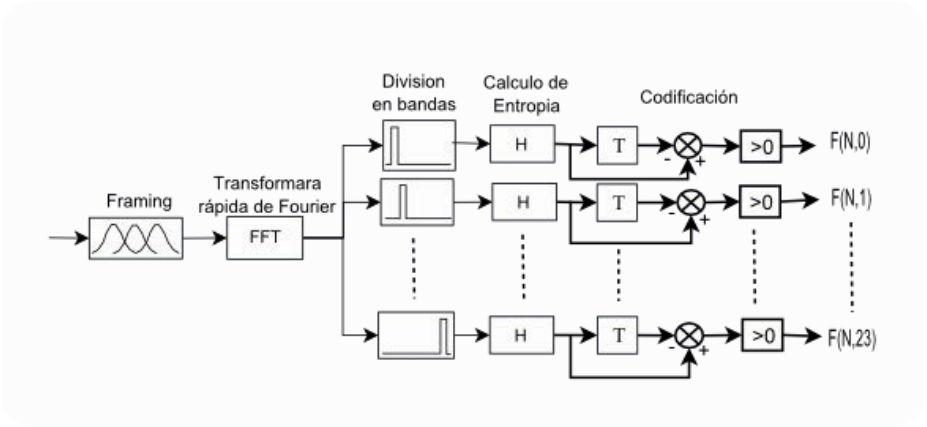

Figura 2.6: Determinación de MBSES, imagen tomada de [Camarena06]

La Figura 2.6 resume la forma de extracción de la firma MBSES, la cual, de acuerdo a [Camarena06] podemos explicar en los siguientes pasos:

1. Las señales estéreo primero deben ser convertidas a monoaural, esto se hace promediando los dos canales

2. La señal se procesa en marcos de 370 ms, este tiempo asegura un soporte de tiempo adecuado para la entropía de acuerdo a los experimentos

3. Los marcos se traslapan un cincuenta por ciento, haciendo que se tengan 5.4 marcos por segundo en la extracción, resultando una firma compacta

4. A cada frame, se le aplica la ventana de Hann y se calcula la transformada discreta de Fourier

5. La entropía de Shanon es calculada para cada una de las primeras 24 bandas criticas de Bark, descartando la ultima banda

Para poder utilizar histogramas se requiere una gran cantidad de datos para poder hacer una estimación confiable de la función de densidad de probabilidad. Sin embargo hay muy pocos coeficientes espectrales agrupados en las primeras bandas criticas del espectro. [Camarena06]

Con una cantidad tan limitada de coeficientes espectrales, se opta por usar un método paramétrico para estimar la FDP. La parte real e imaginaria de la transformada discreta de Fourier pueden modelarse como variables aleatorias Gausianas independientes. Para el calculo de esta firma entonces, se asumirá que los coeficientes espectrales de cada banda critica también siguen una distribución Gausiana. Aunque como reporta [Camarena06], experimentalmente, esta suposición parece no tan alejada de la realidad.

Cuando una señal se apega a una distribución Gausiana con media cero y varianza σ^2, se cumple la ecuación 2.9

$$p(y) = \frac{e^{-y^2/2\sigma^2}}{\sqrt{2\pi\sigma}} \tag{2.9}$$

En [Camarena06] se demuestra la ecuación (2.10) :

$$H = \log(2\pi e) + \frac{1}{2}\log(\sigma_{xx}\sigma_{yy} - \sigma_{xy}^2) \tag{2.10}$$

donde σ_{xx} y σ_{yy} son las varianzas de la parte real e imaginaria respectivamente, y $\sigma_{xy} = \sigma_{yx}$ es la covarianza entre la parte real y la parte imaginaria del espectro en su forma rectangular de tal forma que $\sigma_{xy}\sigma_{yx} = \sigma_{xy}^2$.

Finalmente se sigue el paso de codificación. Esté paso consiste en en guardar simplemente de cada banda, un indicador de si la entropía espectral aumenta o no en el frame actual, en la Ecuación (2.11) se muestra como el bit correspondiente a la banda b y frame n de la firma, es determinado usando los valores de entropía de los frames n y *n-1*.

$$F(n,m) = \begin{cases} 1 \text{ si } [h_b(n) - h_b(n-1)] > 0 \\ 0 \text{ si } Deotraforma \end{cases} \tag{2.11}$$

De esta forma, obtenemos 24 bits por frame (con excepción del primero y el ultimo marco), para hacer la comparación entonces entre dos firmas, se utiliza la distancia de Hamming.

2.3. Conclusiones

En este capitulo se analizaron las propuestas de firma de audio de [Haitsma02, Group01, Camarena06], las dos primeras frecuentemente citadas en la literatura. Los requisitos que nos hemos propuesto para el sistema es utilizar la firma que mejor se comporte respecto a lo que se introdujo en el capitulo era requerido y/o deseado en una firma.

El enfoque planteado en [Haitsma02] procesa 86.2 frames por segundo, de este proceso resultan 86 "sub-firmas" compuestas de 32 bits cada una. Para comparar dos firmas se utiliza la distancia de Hamming, esta distancia resulta muy eficiente pues puede implementarse usando la instrucción xor a nivel de bits. Además cuenta con una forma de organizar múltiples firmas de tal forma que las comparaciones entre individuos sean eficientes (mejorando su escalabilidad), esta esquema de organización será discutido en el siguiente capitulo. Esta firma fue comparada con la firma presentada en [Wang03] en [Nieuwenhuizen10] para fines de monitoreo, donde la propuesta de [Wang03] fue la que presento un mejor comportamiento en audio que según el propio articulo es "Virtualmente libre de ruido y sin comprimir".

Los descriptores MPEG7 procesan 32 frames cada .96 segundos, dando como resultado 32 números entre 0 y 1 que se comparan usando la distancia de Mahalanobis mostrada por la ecuación (2.6), los descriptores cuentan con información estadística para evitar comparaciones innecesarias. Este enfoque resulta muy robusto y también ha sido utilizado con éxito para fines de monitoreo en [Hellmuth01]. El uso mas destacado de esta firma es dentro del programa "Soundhound" como firma utilizada para el reconocimiento de canciones. En [Andrew10, Kevin11, CEO11], se hace una comparación a nivel usuario de Soundhound con otro popular programa de reconocimiento conocido como "Shazam", este ultimo hace uso de la firma propuesta en [Wang03] para llevar a cabo la identificación. Las pruebas a nivel de usuario que realizaron esos 3 diferentes sitios de análisis llegan a conclusiones similares, declarando que mientras que ambas aplicaciones obtienen buenos resultados, Shazam siempre requiere una mayor duración de la muestra a buscar, además de requerir mas tiempo para entregar resultados una vez administrada la muestra de audio, además de ello, Shazam también fue la firma que presentó mayor cantidad de "Falsos negativos", entendidos como

el reportar que no pudo localizar la canción, cuando esta se encontraba en la base de datos, lo cual para nuestro propósito hace que resulten mas interesantes los descriptores MPEG7 que [Wang03], la cual a su vez resultaba mas adecuada que el enfoque de [Haitsma02], sin embargo falta revisar los puntos fuertes de la ultima firma discutida.

El enfoque [Camarena06] procesa 5.4 frames por segundo, lo cual resulta en 5.4 "sub-firmas" de 24 bits cada una, estas se comparan utilizando la distancia de Hamming. Este ultimo enfoque fue además utilizado en [Camarena09] para monitoreo fuera de linea con algunas modificaciones en el tamaño de frame y el traslape, en esa ocasión se utilizó audio severamente degradado, y aún así se obtuvieron resultados satisfactorios. Esta ultima firma resulta ser la mas compacta de los 3 enfoques analizados, adicionalmente al usar la distancia de Hamming hace que resulten muy eficientes las comparaciones. Además como reporta [Camarena06] esta ultima fue la que mejor se comporto ante degradaciones. En base a esto, MBSES será la firma que utilicemos para la construcción del sistema. Aunque hay que considerar que en este punto, se han utilizado para la comparación de canciones, para llevar a cabo la tarea de monitoreo de estaciones, modificaremos algunos de los parámetros de la firma, esto debido a la naturaleza de los anuncios que son en su mayoría voz y suelen tener menor contenido en frecuencia puesto que las emisoras suelen transmitir con frecuencias de muestreo menores, y en base a experimentación se decidirá que parámetros son los que mejor se adaptan a dicha tarea, de momento se descarta la posibilidad de usar [Wang03], debido principalmente a la reportada baja granularidad que posee.

La firma en si misma cumple con las características deseables en una firma, sin embargo no se cuenta con un indice, que podría aumentar aun mas la escalabilidad de dicha firma. El siguiente capitulo analizaremos como podemos organizar las firmas para evitar búsquedas exhaustivas y mejorar el tiempo de las búsquedas, entre estas mencionaremos enfoques de búsquedas por proximidad que esperamos poder aplicar a la firma elegida.

Capítulo 3

Indexado y recuperación de elementos de una base de datos

3.1. Introducción

Una vez caracterizadas las señales de audio que queremos identificar y modeladas como firmas de audio, estas han de guardarse y organizarse de alguna manera, para cuando se requiera hacer una consulta, a la que llamaremos q de aquí en adelante , y determinar si el contenido es suficientemente similar a alguna de las firmas ya conocidas, y en base a ello poder decir que se trata del mismo audio, a esto se le conoce como búsqueda por proximidad. No se debe pretender una búsqueda exacta, ya que en aplicaciones reales, sera muy difícil que el proceso de extracción de características y modelado nos produzca individuos idénticos a las muestras de la base de datos (salvo que procesemos exactamente el mismo individuo).

La forma mas sencilla de buscar es una comparación secuencial, es decir, comparar q contra todos los elementos en la base de datos mediante el cómputo de una función que determina que tan similar es un individuo a otro, llamada distancia. La mayor ventaja de este enfoque, es que de existir en la base de datos un individuo lo suficientemente similar a q, como para considerar que q se encuentra en la base de datos, este enfoque lo encontrara, sin importar el tamaño de la base de datos, ni el orden de esta. Sus desventajas son principalmente que será computacionalmente costoso, a medida que la base de datos crezca y

a medida que el calculo de la función de distancia sea mas caro en términos de tiempo que tiene que dedicar el procesador a esta. Por tanto este enfoque solo a de aplicarse cuando se tiene una base de datos pequeña, y el calculo de la distancia no resulte costoso.

La idea detrás de organizar la caracterización de las señales de audio, es buscar a q lo mas eficiente posible, en este capitulo se tratan algunas maneras para organizar la base de datos, orientadas a llevar a cabo este objetivo.

3.2. Espacios métricos

Antes de introducir los esquemas que permitirán una recuperación mas eficiente de *q*, es necesario introducir el concepto de "espacios métricos", a continuación se presenta una breve introducción a los espacios métricos, si el lector desea conocer mas acerca de la basta teoría de los espacios métricos, se recomienda revisar el libro [Kahmsi01] Un espacio se distingue de un conjunto al poseer atributos no poseídos por una mera colección de objetos, y un subespacio es un subconjunto de un espacio que se asume ha heredado dichos atributos. Un espacio métrico sugiere que dados dos puntos en el espacio, debe de haber un número real que mida la distancia entre ellos, a este número también se le conoce como "métrica" . Para discutir lo que es una "métrica", iniciemos con un par *(M,d)*, donde M es un conjunto y $d : M \times M \to \mathbb{R} \geq 0$ y *d:* es un mapeo del producto cartesiano $M \times M$ a números reales no negativos $\mathbb{R} \geq 0$. Si *d(x,y)* se interpreta como la distancia entre dos puntos $x,y \in M$, se puede asumir que d satisface las siguientes restricciones para cada $x,y \in M$: [Kahmsi01]

I. $d(x, y) = 0 \Leftrightarrow x = y$

II. $d(x, y) = d(y, x)$

Un par *(M,d)* que satisface las dos condiciones anteriores, se le conoce como espacio "Semimetrico". Los espacios semimétricos son una subclase de los espacios métricos, el espacio semimétrico se puede considerar como espacio métrico, si además de las dos condiciones anteriores satisface la siguiente:

III. (Desigualdad triangular) Para cada punto $x, y, z \in M$:

$d(x,y) \leq d(x,z)+d(z,y)$

La distancia en un espacio métrico puede ser de dos tipos, dependiendo de las características de los valores regresados, discretas y continuas[Zezula05]. Una distancia discreta, regresa valores predeterminados en un rango pequeño, en este rango el número mas pequeño indica que el objeto de la consulta es el mismo al que se compara, y la distancia mas grande implica que el objeto de la consulta es lo mas diferente que puede ser al objeto con el que se compara (es común elegir 0 y 1 respectivamente, pero no es indispensable), mientras que una distancia continua puede tomar valores infinitos. Un ejemplo de distancia discreta es la distancia de edición entre dos cadenas, mientras que un ejemplo de distancia continua es la distancia euclidiana.

El espacio métrico mas natural que existe es, la recta de los reales $\mathbb{R}$ con el valor absoluto como métrica o distancia $d(x,y) = |x - y|$ para $x, y \in \mathbb{R}$. [Kahmsi01]

La teoría de los espacios métricos es amplia y se podría escribir mucho al respecto, sin embargo, con esta breve introducción es suficiente para que el lector pueda comprender lo relacionado con espacios métricos que se tratara en esta tesis, de este punto en adelante, a la métrica simplemente se le llamara distancia buscando homogeneizar el termino con el calculo de distancias de las firmas de audio, que a final de cuentas es la métrica en un espacio de firmas.

3.3. Estructuras de datos métricas

Una "Estructura de datos métrica" es una manera de organizar los puntos de un espacio métrico, de tal manera que la estructura cuente con un mecanismo de búsqueda que permita discriminar puntos del espacio métrico, evitando la comparación con estos, esta discriminación generalmente se logra aprovechando la condición que deben cumplir dichos espacios, conocida como "desigualdad triangular", al evitar comparar una consulta q, contra todos los puntos de un espacio métrico, se logra buscar q de manera mas eficiente.

Un concepto que se manejara recurrentemente en las próximas páginas es el de "pivotes". Un pivote representa una referencia en este caso, dentro de un espacio métrico, que pudiera ser un elemento del conjunto de puntos en el espacio métrico, o un punto artificial, que sirve para ser comparado contra otros elementos. El comparar con estos pivotes,

nos permite ahora ubicar a nuestro conjunto no solo respecto a su posición en el espacio, si no respecto a su posición con los pivotes. Conviene también definir lo que es "Ancho de anillo" el ancho de anillo es un rango de distancia alrededor de un pivote. A este rango le llamaremos w, debe de cumplir lo siguiente $d_{min} \leq w \geq d_{max}$, donde d_{min} representa la distancia mínima posible en el espacio métrico (que es 0, al comparar un punto contra el mismo), y d_{max} es la distancia máxima que puede haber entre dos puntos en el espacio. Un anillo cuyo ancho $w = d_{max} - d_{min}$, representará a todos los puntos en el espacio métrico. Como se muestra en la Figura 3.1

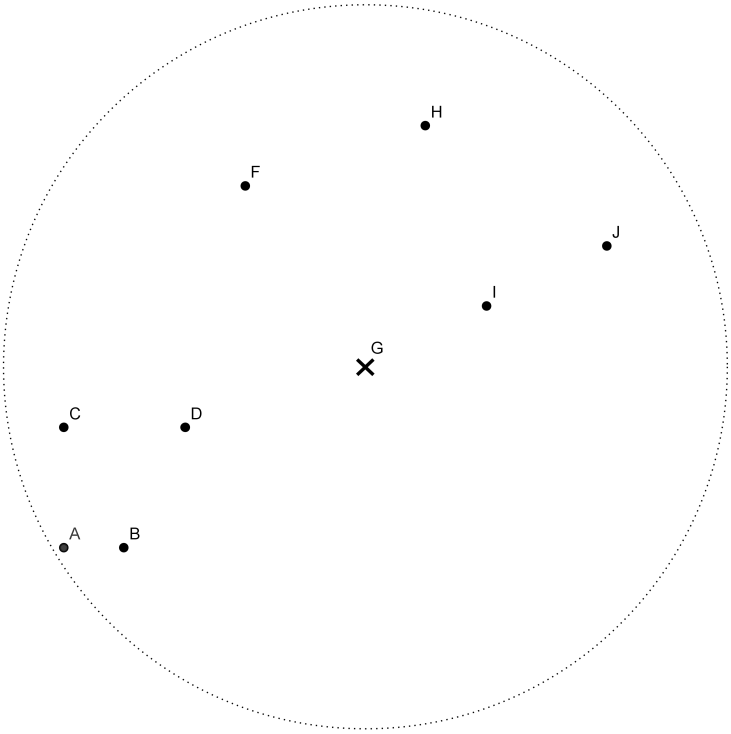

Figura 3.1: Figura que muestra un anillo respecto al punto G con ancho $w=d_{max}$, en la imagen, se puso el anillo alrededor de todos los puntos lo mas ajustado, pero un anillo con $w=d_{max}$, abarcaría mas espacio.

Redistribuir los puntos alrededor de un solo pivote con un solo anillo, no permite hacer búsquedas eficientes, por lo tanto esto se hace con varios anillos, definiendo un ancho de anillo mas pequeño, así se tiene un número de anillos N tal que $\lceil N = d_{max}/w \rceil$. Además el numero de pivotes es igualmente importante, y este ha de escogerse en base a la cantidad

de elementos en el espacio métrico, un número pequeño de pivotes no permitiría descartar una gran cantidad de elementos, mientras que un número exagerado de pivotes, haría que el ahorro en comparaciones fuera menor. Si utilizáramos un ancho de anillo menor para el ejemplo de la Figura 3.1, de tal forma que pudiéramos tener al menos 3 anillos por pivote, y ademas usáramos como pivotes los puntos D e I de nuestro ejemplo, tendríamos como resultado la Figura 3.2

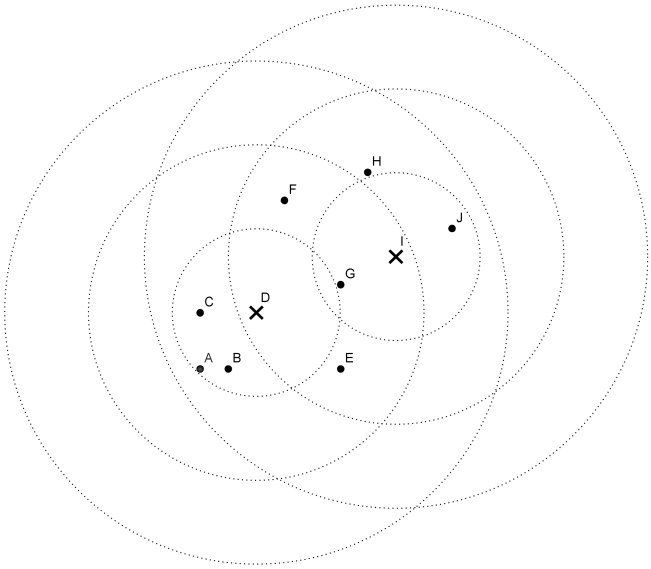

Figura 3.2: En esta figura se muestran 3 anillos usando a los puntos D e I como pivotes

Usando la ubicación de los puntos respecto a los pivotes y sus anillos, en vez del espacio, se obtiene la distribución mostrada en la Figura 3.3

Para entender como es que de esta manera evitaremos comparaciones, consideremos una consulta q, ilustrada en la Figura 3.4

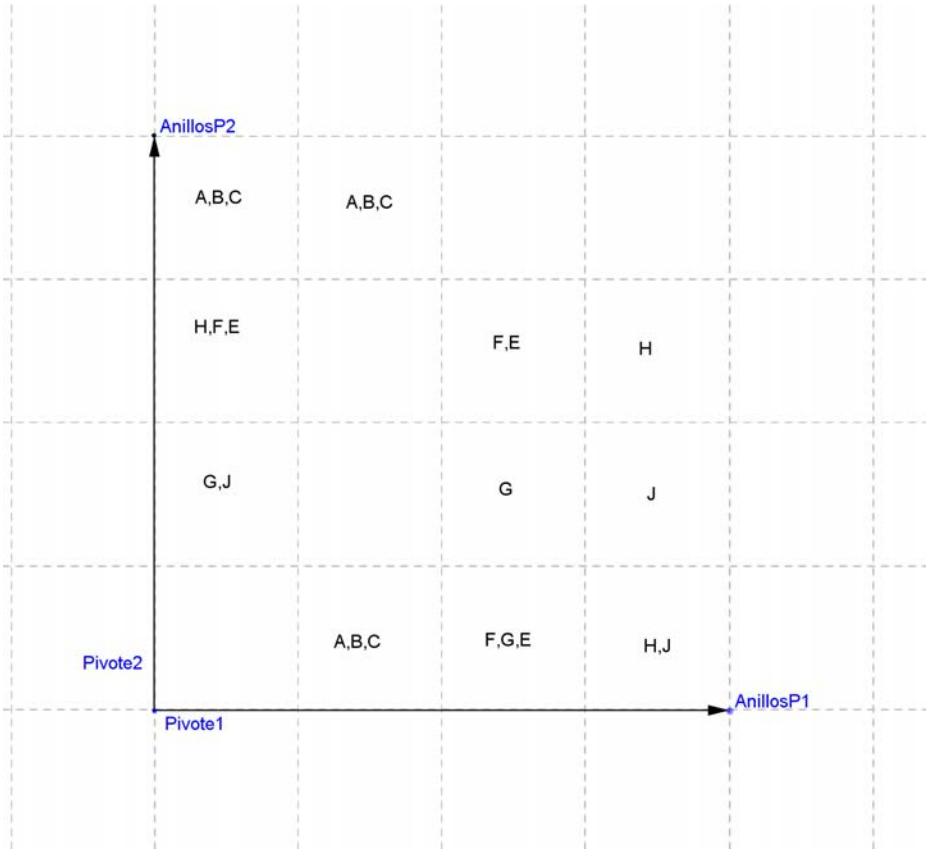

Figura 3.3: Podemos observar como se distribuyen los puntos respecto a los pivotes en los anillos de estos

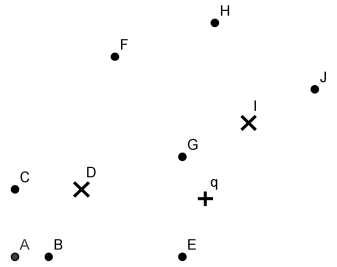

Figura 3.4: Podemos observar al elemento q

En la Figura 3.4 , se puede observar que el punto q esta muy cercano a los puntos G y E, nosotros somos capaces de observarlo, pero la computadora no puede hacer esta determinación visual, pero si comparamos solamente contra los pivotes D e I, podemos ubicar el anillo en cada uno al que q pertenece

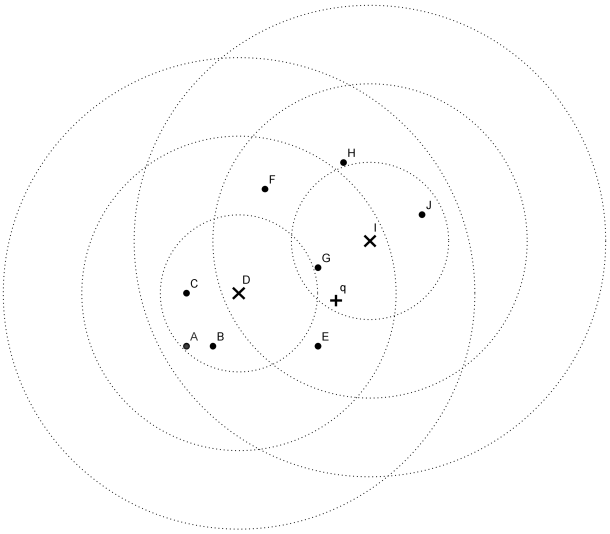

Figura 3.5: Aquí ya con los anillos dibujados, podemos ubicar a los elementos mas cercanos a q

En la Figura 3.5 , se observa entonces, que el punto q, está en el segundo anillo respecto a D, y en el primer anillo respecto a I, en esta combinación, solo se encuentra el punto G, y se podría considerar además el pivote I (si es parte del conjunto de datos y no es un punto artificial), la comparación con I ya realizó (puesto que es uno de los pivotes), solo falta comparar con G, y determinar cual de estos dos puntos es el mas cercano a q.

En las siguientes páginas, explicaremos dos estructuras de datos que explotan estos conceptos descritos anteriormente, dichas estructuras son los "Árboles B-K", y los "Arreglos de consulta fija"

3.3.1. Árbol B-K

La primera de las estructuras de datos métrica que se presenta apareció por primera vez en [Burkhard73]. Se le conoce como árbol Burkhard-Keller. Este árbol asume una distancia discreta y se construye recursivamente de la siguiente forma: De un conjunto de

puntos S, un objeto arbitrario $p \in S$ es elegido como el nodo raíz del árbol. Para cada distancia $i \geq 0$, subconjuntos $S_i = \{o \in S, d(o,p) = i\}$ son definidos como los grupos de todos los objetos a distancia i del nodo p. Un hijo del nodo p es construido por cada conjunto no vacío S_i. Todos los nodos hijos pueden ser repartidos recursivamente hasta que ya no sea posible crear un nuevo nodo hijo [Zezula05]. Este proceso se repite, eligiendo a uno de los individuos de cada conjunto S_i como sub-árbol y repartiendo de forma recursiva cada conjunto. El proceso termina cuando ya no hay elementos que puedan ser repartidos (cuando los conjuntos S_i contienen un solo elemento). Los elementos elegidos como raíces de los sub-árboles, y la misma raíz son llamados pivotes.

Para realizar una consulta por proximidad (o por rango) en este índice, se hace de la siguiente manera. La búsqueda $R(q,r)$, inicia con el pivote raíz del árbol y lo compara con su objeto p con la consulta q. Si p satisface la consulta, eso es si $d(p,q) \leq r$, el objeto p es devuelto como la solución. Subsecuentemente, el algoritmo entra a todos los nodos hijos o_i tal que

$$max\{d(q,p) - r, 0\} \leq i \leq d(q,p) + r \tag{3.1}$$

Y procede de manera recursiva hacia abajo. como resultado de la Ecuación 3.1 se cortan algunas de las ramas del árbol [Zezula05], evitando así realizar comparaciones con algunos pivotes. Dentro de las desventajas de este método, se encuentran que el árbol resultante no necesariamente esta balanceado, además de que dependiendo del tamaño de anillo y la distribución de los datos, se podrá degenerar al grado de ser una lista ligada.

Ejemplo de construcción del Árbol B-K

Para ejemplificar como se construye este árbol, utilizaremos el conjunto de puntos en el plano cartesiano de la Figura 3.1, los cuales están etiquetados como **A, B, C, D, E, F, G, H, I, J**, dichos puntos no representan nada en particular, solamente serán usados para ilustrar como se construye el árbol, las distancias entre ellos las damos a continuación en la Tabla 3.1

d(x,y)										
x \ y	A	B	C	D	E	F	G	H	I	J
A	0	1	2	2.82	5	6.7	5.8	9.21	8.06	10.2
B	1	0	2.23	2.23	4	6.32	5	8.6	7.2	9.4
C	2	2.23	0	2	5.3	5	5.09	7.8	7.2	9.4
D	2.82	2.23	2	0	3.6	4.1	3.1	6.4	5.3	7.6
E	5	4	5.3	3.6	0	6.3	3	7.07	4.47	6.4
F	6.7	6.32	5	4.1	6.3	0	3.6	3.1	4.4	6.0
G	5.8	5	5.09	3.1	3	3.6	0	4.1	2.2	4.4
H	9.2	8.6	7.8	6.4	7.07	3.1	4.1	0	3.1	3.6
I	8.06	7.2	7.2	5.3	4.4	4.4	2.2	3.1	0	2.2
J	10.2	9.4	9.4	7.6	6.4	6.0	4.4	3.6	2.2	0

Tabla 3.1: Tabla con las distancias de los puntos

Ahora, la distancia entre estos puntos es la distancia euclidiana, está es considerada como una distancia continua, sin embargo, podemos acotar el espacio para este ejemplo y considerar la distancia como discreta, por razones de conveniencia a este ejemplo, se dirá que la distancia máxima entre 2 puntos es de 12. Empecemos con la construcción, elegimos arbitrariamente un punto, en este caso el punto A como raíz, si establecemos el ancho de anillo W=3, entonces tendremos 4 particiones $S_i = S_3$, S_6, S_9 y S_{12}, en dichas particiones tendremos los siguientes elementos $S_3 = \{B, C, D\}, S_6 = \{E, G, I\}, S_9 = \{F\}, S_{12} = \{H, J\}$ El conjunto S_9 tiene un solo elemento, este elemento sera una hoja del árbol, ya que al ser un solo elemento, este ya no se repartirá. En esta iteración, el árbol se muestra en la Figura 3.6:

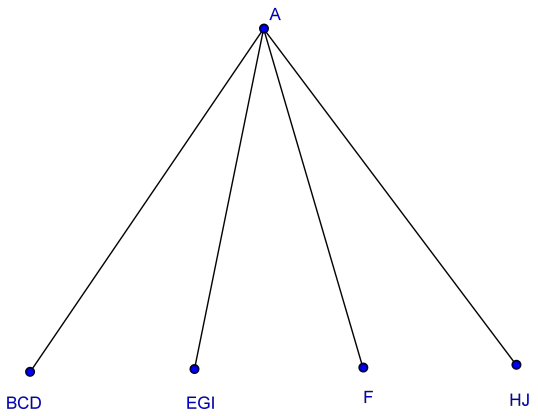

Figura 3.6: Primera iteración del árbol B-K

Enseguida se procede a repartir los conjuntos que tienen mas de un elemento, de el conjunto S_3 si tenemos a B como raíz del sub-árbol, podemos tener $S_{3,3} = \{C, D\}$, en tanto los otros conjuntos resultan vacíos, en S_6 si elegimos a E como raíz también del sub-árbol tenemos $S_{6,6} = \{G, I\}$ y $S_{6,6,6} = \{I\}$, el conjunto S_9 ya cuenta con un solo elemento y ya no se puede repartir, mientras que si en S_{12} elegimos a H como raíz del siguiente sub-árbol, entonces $S_{12,6} = \{J\}$. Se procede de esa forma hasta que todos los conjuntos tengan un solo elemento, el resultado final con el ejemplo presentado se muestra en la Figura 3.7.

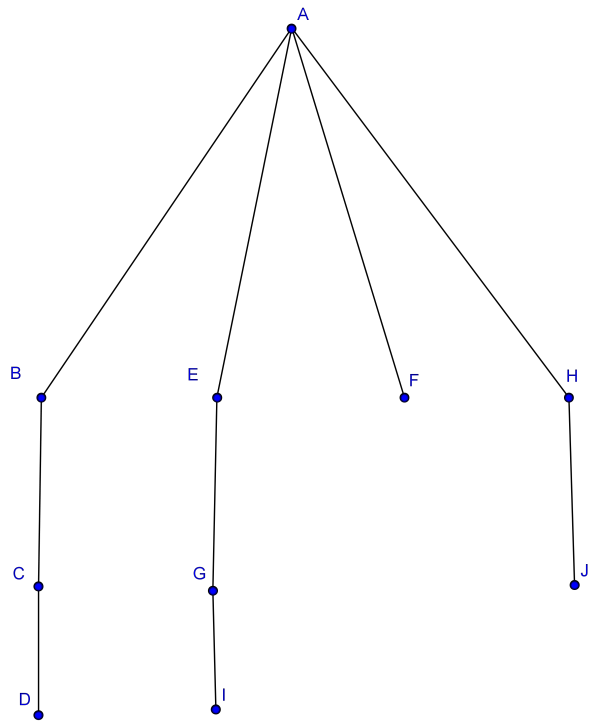

Figura 3.7: Árbol Burkhard-Keller completo para el ejemplo

Para ejemplificar como se lleva una búsqueda, retomemos el ejemplo de la Figura 3.4. Primero necesitamos establecer el rango, para este caso sera $r \leq 2$. Primeramente obtenemos $d(q, A) = 5.9$, no satisface nuestra restricción de rango, pero al compararlo, esto ubica el punto q en S_6, en donde habría que comparar con el pivote E, $d(q, E) = 4.4$, lo cual todavía no satisface el rango , pero permite ubicar a q en $S_{6,6}$, en donde la siguiente comparación corresponde a G, $d(q, G) = 1.2$. Este ultimo satisface la restricción de rango, así que devolvemos al punto G, aún necesitamos hacer la comparación con I, esto es $d(q, I) = 1.9$, este ultimo no cumple la restricción y además ya no hay mas elementos que comparar. En solo 3 comparaciones pudimos encontrar al punto que cumplió con la restricción de rango (y hay una comparación adicional después de haberlo encontrado), sin embargo, en otros casos hay que ampliar el rango.Hay que notar que en este caso se restringió la distancia euclidiana, pero hay que recordar que no es una distancia discreta y la estructura asume distancias discretas.

3.3.2. Arreglo de consulta fija

La siguiente estructura a discutir, fue planteada en [Chávez01]. Está estructura comparte mucho en común con una modificación del árbol B-K conocido como "Árbol de consulta de altura fija", en dicho árbol las consultas se realizan siempre comparando contra los mismos pivotes.Los arreglos de consulta fija no organizan los elementos en una estructura de árbol, sino en arreglos. Para construir el arreglo de consulta fija, se debe primeramente elegir un número de pivotes. En [Chávez01] se plantea el uso de $\Theta \log(N)$, donde N es el número de elementos dentro del conjunto. Enseguida, los elementos en el arreglo se tienen que ordenar de manera lexicográfica respecto a su distancia del primer pivote, en caso de existir un empate entre dos elementos, este se romperá utilizando al siguiente pivote y así sucesivamente hasta que todos los elementos hayan sido ordenados. En la Figura 3.8 se muestra el resultado de como se presentarían los puntos de la figura 3.1 usado como pivotes los puntos B, E y H

	B	A	D	C	E	G	F	I	H	J
B	0	1	2.23	2.23	4	5	6.7	7.2	8.6	9.4
E	4	5	3.6	5.3	0	3	6.3	4.47	7.07	6.4
H	8.6	9.2	6.4	7.8	7	4.1	3.1	3.1	0	3.6

Figura 3.8: Arreglo de consulta fija para los puntos de la figura 3.1

Para realizar una búsqueda por rango, se procede de manera similar como con el árbol B-K, retomemos nuestro punto q del ejemplo del árbol. Primero se compara q con el primer pivote para ubicar que rango le corresponde, $d(q, B) = 5$, si hacemos una analogía con las ramas en el árbol B-K, podemos entonces ubicar a q en un rango de 3 a 6 respecto a B, esto nos permite descartar a los pivotes B, A, D, C, F, I, H, J. Dejando solo 2 posibles puntos, siendo uno de ellos G, que es el punto que cumple con el requisito, podríamos entonces detenernos y realizar una comparación secuencial entre los puntos restantes.Sin embargo si deseamos continuar, entonces podemos obtener $d(q, E) = 2$, de nueva cuenta podemos ver que el rango correspondiente, sería de 0 a 3, podemos notar que tanto E y

G se encuentran en este rango, no permitiendo descartar ninguno de los dos. Finalmente $d(q, H) = 5$, esto ubica a q en el rango de H de 3 a 6, el único punto que queda en este rango es el punto G, pues los otros en ese rango ya se encontraban descartados por los pivotes anteriores. El equivalente a las visitas a los nodos del árbol se simulan mediante el algoritmo de búsqueda binaria [Chávez01]. En este ejemplo con visitar el equivalente a un solo anillo obtuvimos buenos resultados, sin embargo dependiendo de la distribución de los datos y de los pivotes elegidos, podría necesitarse visitar anillos adicionales.

3.4. Indice invertido

El siguiente enfoque es el que utilizan para organizar las firmas en [Haitsma02], en ese articulo le llaman LUT (Look up table, o tabla de búsqueda), se trata de un índice que comparte mucho en común con una estructura utilizada para la búsqueda de fragmentos de texto en archivos, conocida como "índice invertido" [Justin06], así que también nos referiremos a este como índice invertido.

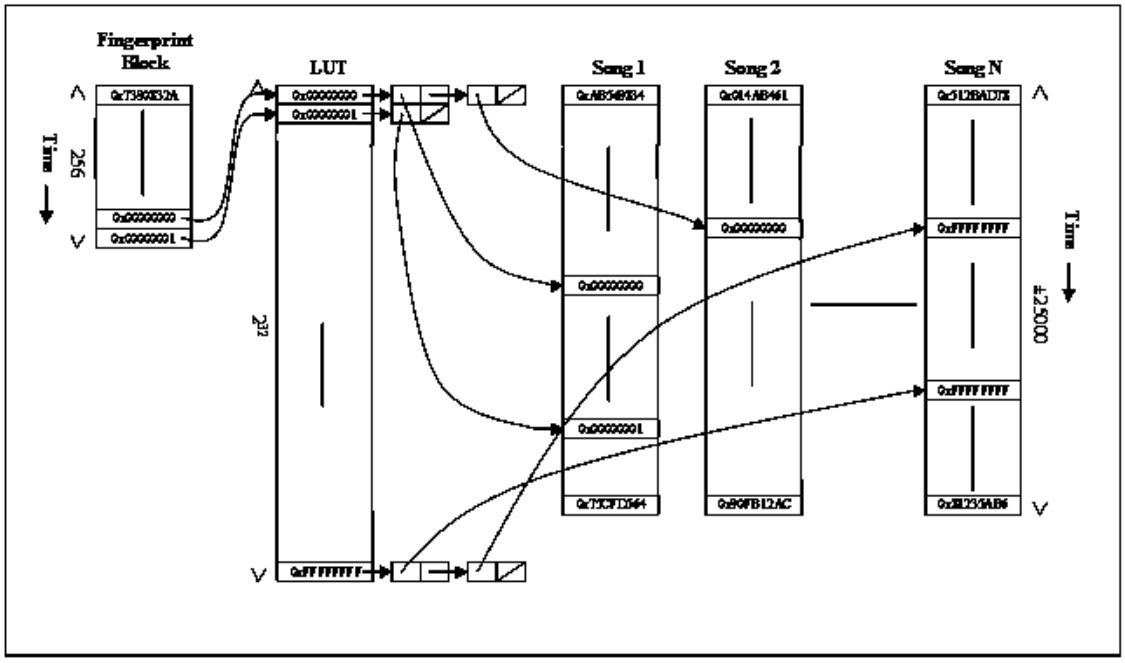

Figura 3.9: Ilustración del índice invertido, imagen tomada de [Haitsma02]

En la figura 3.9 se muestra de manera ilustrativa como es el índice y como se realiza una consulta con este. Para construir el índice, primero recordemos como se representa un frame de la firma presentada [Haitsma02]. Los frames en esta firma son cadenas de bits. Esto significa que cada frame (o "sub-firma") se puede representar utilizando el valor entero que codifican los bits. Teniendo en cuenta esta posibilidad, cada frame se puede representar como una cadena de enteros decimales, estando compuestos por 32 bits cada frame, si consideramos solo enteros decimales positivos, cada frame puede tomar un valor entre 0 y $2^{32}-1$. Si tenemos una tabla con los valores precisamente de este intervalo, podemos guardar en que posición de la firma ocurre un frame en la entrada de la tabla correspondiente al entero codificado por los bits del frame. Esto se lleva a cabo con cada frame de la firma, y se repite para todas las firmas. Hay que tener en cuenta que en la firma MBSES solo se tienen frames de máximo 24 bits, lo cual implica una tabla de mucho menor tamaño.

Las búsquedas en este índice no son exactamente por proximidad, si no por fragmentos pequeños de un archivo mas grande. Cuando se tiene una consulta q, un frame se codifica como un entero decimal, con este entero decimal, podemos consultar la tabla creada y determinar en que posición de las firmas de la base es donde ocurre ese frame. Al tener la posición, podemos ubicar la consulta en las posiciones donde podría ocurrir en cada una de las firmas de la colección, eventualmente si la consulta corresponde a una de las firmas de la base, esta se podrá ubicar y la medida de distancia nos permitirá saber que ha ocurrido. Podemos notar que si existen cambios en los bits de los frames de la consulta, estos no codificaran con el mismo entero que fue codificado cuando se creo la tabla, haciendo que la consulta no se pueda ubicar en la posición idónea y por lo tanto no se puedan reportar si la consulta ocurre. La primera forma para manejar esta situación, es simplemente considerar que eventualmente existirá un frame en la consulta que codifique exactamente igual que como fue codificado en el archivo originalmente indexado, esto permite encontrar la ocurrencia si es que existe, independientemente de que la extracción de la firma no produzca resultados exactamente iguales, esto va a depender de que tan robusta resulta la firma. La otra forma de tratar este problema, es consultar las entradas de la tabla cambiando algunos bits. Esta última solución no resulta tan alentadora, pues el espacio de búsqueda podría crecer demasiado.

3.4.1. Ejemplo

Vamos a ejemplificar como se construye el índice y como es que se hace una consulta. Vamos a representar números de solo 4 bits, y lo haremos en su forma decimal y no en su forma binaria, por ejemplo, el número binario 0000 codifica como 0 en su forma decimal. El tener 4 bits da $2^4 = 16$ entradas para la tabla. Entonces supongamos que vamos a indexar archivos con los siguientes frames, $A1 = \{15, 7, 15, 0, 0, 15\}$, $A2 = \{15, 9, 4, 1, 15, 12\}$, $A3 = \{5, 6, 9, 10, 6, 10\}$, $A4 = \{3, 10, 11, 8, 13, 15\}$, en la imagen 3.10 se muestra como resultaría el indice.

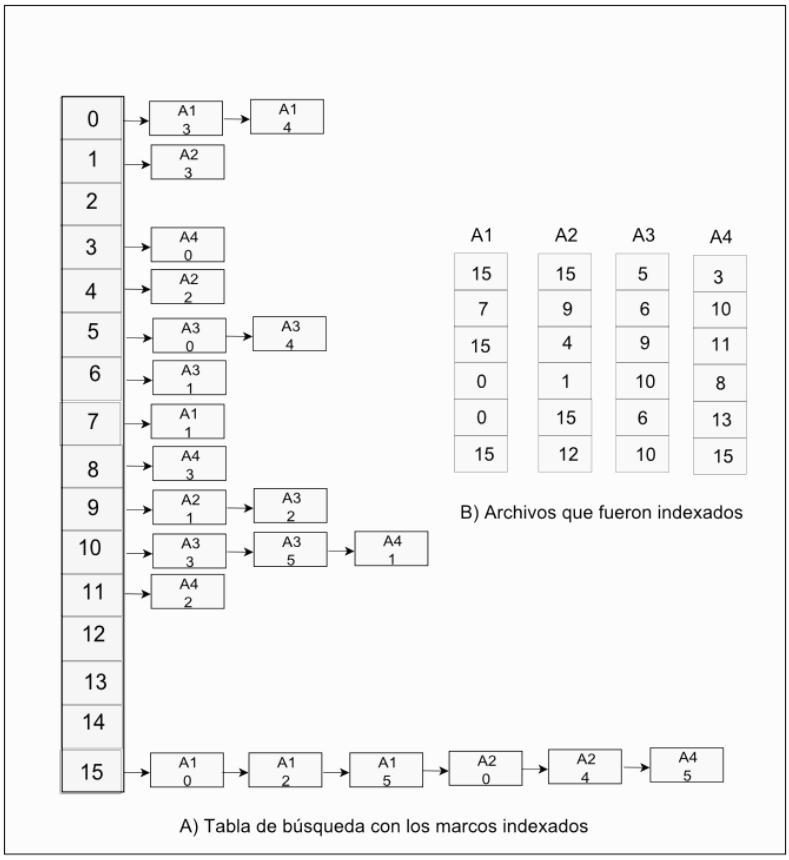

Figura 3.10: Imagen de un índice invertido

Ahora, si suponemos que necesitamos hacer una consulta $q = \{10, 10, 8\}$ podemos notar de inmediato que el tamaño de la consulta resulta menor al tamaño de los archivos

indexados, esto no representa un problema en este índice. El primer frame de la consulta nos indica que en el archivo $A3$ en su posición 3 ocurre ese frame. Enseguida procedemos a revisar, y podemos notar que por debajo de ese frame ya no hay nada mas en el archivo $A3$, el siguiente archivo en donde ocurre ese mismo frame es en el archivo $A4$. Podemos notar que esta entrada ocurre en la posición 1 de dicho archivo, con lo cual podemos tomar las posiciones $1, 2$ y 3 para nuestra consulta. Al hacer la comparación con la distancia de Hamming, podemos darnos cuenta que nuestra consulta solo difiera en 1 bit con el contenido de $A4$, esto si vemos el segundo frame de nuestra consulta es 10, este número codifica en binario como 1010, mientras que el número indexado en esa misma posición es 11, el cual codifica como 1011, el resto de los números son iguales. Si tenemos como tolerancia 1 bit de diferencia en este caso, entonces podemos reportar la ocurrencia. Se debe notar que este es un ejemplo muy sencillo, y podría ocurrir en una aplicación mas complicada que cambie uno o varios bits en todos los frames de la firma, y sin embargo las firmas sean lo suficientemente parecidas como para considerar que es del mismo audio. En caso de que esto ocurra, la búsqueda tal como está planteada no funcionaría.

3.5. Conclusiones

En este capitulo se discutieron algunas estructuras de datos orientadas a mejorar las búsquedas, los cuales pretendemos aplicar al problema de monitoreo en tiempo real de estaciones de radio, mas específicamente, con ellos pensamos ordenar la colección de firmas de audio de los comerciales utilizando los enfoques vistos en este capítulo.

El primer enfoque, el árbol B-K, es un enfoque que ha inspirado muchas variaciones, como los árboles de consulta y los árboles de consulta de altura fija. A pesar de que en el ejemplo mostrado en la sección correspondiente al árbol B-K se llegaba a resultados extraordinarios, no siempre resulta así, y hay ocasiones en que es necesario visitar ramas adicionales del árbol, en el peor de los casos, se puede obtener el equivalente a una búsqueda secuencial. Es por eso que no damos por hecho que este enfoque resultara y no sera necesario explorar otros. Además de que su complejidad de construcción es en términos de computo de distancias $O(n \log n)$.

El siguiente enfoque, los arreglos de consulta fija, fueron elegidos por la simplicidad que implica para en los lenguajes de programación el manejo de arreglos. Estos arreglos igualmente pueden evitar la necesidad de búsquedas lineales, al mismo tiempo de no requerir usar una representación en forma de árbol, de igual forma, la construcción puede resultar tan eficiente como lo sea el algoritmo de ordenamiento elegido para su construcción.

Ambas estructuras son basadas en pivotes, diseñadas para distancias discretas, si se aplican a casos continuos, se pueden degenerar y provocar de acuerdo a [Zezula05] un "timo lineal", esta observación además aplica para otras estructuras basadas en pivotes. La distancia de Hamming es una distancia discreta, sin embargo esto no garantiza buenos resultados, tampoco hablan de un criterio para la selección de pivotes, situación que podría impactar directamente en el desempeño de cualquier índice basado en pivotes. Pueden ocurrir casos en donde sea necesario recorrer todos los elementos de las estructuras para obtener los resultados deseados, en este caso, la solución resulta un poco peor que la búsqueda lineal debido a los "Computos laterales", que son aquellos que se llevan a cabo para la construcción del indice, en vista de esta posibilidad, se plantea adicionalmente otra posibilidad de solución. No solo el criterio de elección de pivotes tiene un impacto en el desempeño de las estructuras, si no también la distribución de los puntos en el espacio métrico. Sin embargo, de las dos estructuras planteadas, los arreglos de consulta fija, son aquellos que pueden potencialmente tener un mejor desempeño, esto además de por la facilidad de el manejo de arreglos, por ser una representación en un arreglo de un árbol balanceado.

La ultima de las posibles soluciones presentadas, es una instancia del indice invertido. Este mas que permitir una búsqueda por proximidad, permite hacer una búsqueda en base a un fragmento pequeño. La razón por la que se espera funcione es por la confianza que se tiene en la robustez de la firma, que permita que existan marcos cuya extracción sea idéntica en posición a los de la muestra, haciendo que podamos ubicar específicamente la parte de los anuncios donde ocurre la consulta. Hay que notar que podemos aplicar esta solución a [Camarena06] por que en esta firma, los marcos de igual manera están codificados como cadenas de bits.

Aunque existe una gran cantidad de posibles soluciones interesantes tales como el índice basado en permutantes [Figueroa07] que ha reportado tener incluso buenos resultados

en espacios semi-métricos, o "los k vecinos mas cercanos sucintos´´ [Tellez11], y enfoques mas parecidos al índice invertido como el hash sensible a localidad, el trabajo se debe acotar, y estos últimos se dejaran para comparaciones futuras, en este trabajo, se presentaran los resultados de indexar las firmas utilizando las 3 estructuras planteadas en este capítulo.

En el próximo capitulo se habla del sistema desarrollado, y de las posibles opciones que se tienen en este, en vista de que ya se hablaron de las partes principales, como son la determinación de una firma de audio que resulta con las cualidades deseadas, y algunos esquemas orientados a mejorar la escalabilidad de la misma.

Capítulo 4

Implementación

4.1. Introducción

Se ha implementado un sistema utilizando el lenguaje de programación java. Además de las herramientas que provee dicho lenguaje, provistas tanto como por el creador del lenguaje como por terceros, su portabilidad y relativa facilidad de establecer un ambiente gráfico hicieron que se decidiera implementar el sistema en dicho lenguaje. El sistema se ha desarrollado y probado en una computadora portátil,que tiene un procesador de 4 núcleos de 1.7 GHZ y 3 GB de memoria de acceso aleatorio.

En este capitulo se mencionan detalles acerca del desarrollo del sistema, de como se adquiere el audio transmitido por internet, y como se implementaron las opciones tanto como de caracterización y modelado del audio, y de indexado que se comentaron en capítulos anteriores.

4.2. Captura del Audio transmitido por Internet

Como se menciono al inicio, se eligió el protocolo de transmisión shoutcast. En este protocolo, como en la mayoría de los protocolos de transmisión por internet, se utiliza una "versión modificada" del protocolo de transferencia de hiper texto (http). Para poder capturar audio con este protocolo, necesitamos la dirección web de la emisora. La dirección suele tener la siguiente forma `http://radio.alphamx.net:8080`. Enseguida del

nombre de dominio, se encuentran los puertos a los que se se tiene que conectar el cliente. El tipo de puerto generalmente es UDP. Podemos conectarnos al servidor utilizando las herramientas que java provee en su paquete `java.net.URL()` como lo es específicamente `URL.openConnection()`. Podemos generar usando dichas herramientas, un flujo de entrada. Al realizar la conexión, obtendremos para la conexión con la URL usada en el ejemplo lo siguiente:

```
ICY 200 OK
icy-notice1:<BR>This stream requires <a href="http://www.winamp.com/">Winamp</a><BR>
icy-notice2:SHOUTcast Distributed Network Audio Server/Linux v1.9.8<BR>
icy-name:RMX - Gdl.
icy-genre:Rock
icy-url:http://www.rmx.com.mx
content-type:audio/mpeg
icy-pub:1
icy-br:56
```

En caso de que el servidor se encuentre operando, recibiremos el mensaje "ICY 200 OK", en caso de que el servicio no se encuentre disponible, recibiremos el mensaje "ICY 401 service unavailable". El resto de los encabezados son auto-explicativos y varían de servidor a servidor. La combinación de caracteres retorno de carro y linea final al aparecer dos veces seguidas (representados como dos saltos de linea en el ejemplo de conexión) son el indicador de que iniciara la transmisión de audio, generalmente se encuentra en formato mp3. En ese punto es cuando debemos decodificar el audio si este se encuentra en mp3. La decodificación la llevamos a cabo usando la librería *javazoom*.

4.3. Determinación de la Firma MBSES

Ya con el flujo de audio decodificado, podemos proceder a calcular la firma. El proceso de determinación de esta es prácticamente idéntico para archivos (las muestras a monitorear), y para transmisiones. La diferencia estriba en que en los archivos se cuenta con todo el audio, y la extracción termina cuando es procesado todo el archivo. En el caso de una transmisión, no resultaría conveniente un criterio de parada relacionado con la duración del audio, ya que en dicho caso, la duración esta mas bien sujeta a las posibilidades de transmisión de las estaciones, esto provoca que sea mas conveniente controlar la extracción

con una variable booleana, dicha variable controla el proceso de monitoreo. Nos permitimos repetir una figura del Capitulo 2 en la Figura 4.1 para describir el proceso de extracción.

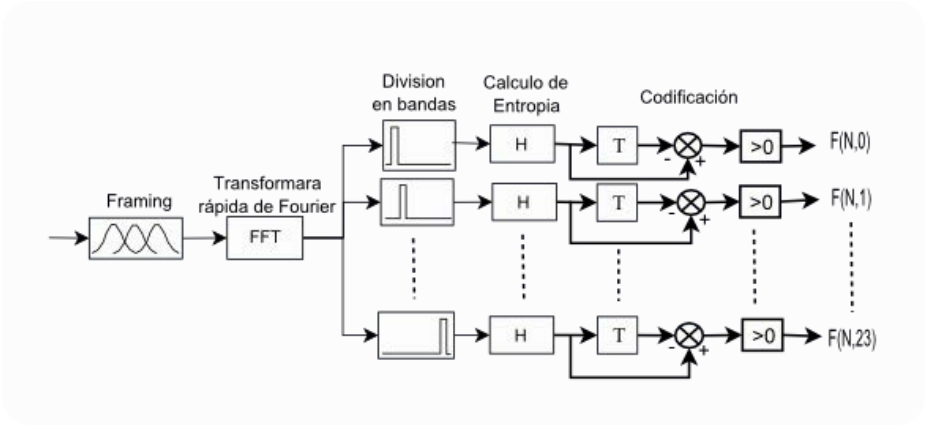

Figura 4.1: Determinación de MBSES, imagen tomada de [Camarena06]

Una vez que tenemos el audio (sin importar la fuente) y haber realizado el pre-proceso (en nuestro caso, el unico pre-procesamiento aplicado, es promediar las muestras para reducir el audio a mono-aural), se sigue el proceso que describe la Figura 4.1, este es el siguiente, el audio se divide en segmentos de duración corta, a los que llamamos marcos, el tamaño de marco es un valor que debe de ser proporcionado por el usuario. Adicionalmente, el siguiente marco que se va a procesar, debe de tener tantos elementos del marco anterior como se indique en otro valor variable, que se conoce como traslape. A cada segmento se le extrae la transformada discreta de Fourier, para esto implementamos la transformada tal como aparece en [Press92]. Enseguida dividimos los coeficientes espectrales que resultan de la transformada en bandas criticas de Bark. al final de esto, se calcula la entropía, utilizando la Ecuación 2.10 mostrada en el Capitulo 2. La duración del audio no modifica la extracción, debido a que la influencia mas directa en cada marco de audio procesado (marco), ese el marco anterior. Tanto para extraer archivos muestra como al realizar extracciones de transmisiones, se escribe un archivo con los bits de la firma codificados en forma de 3 bytes de 8 bits cada uno, en caso de que no se haya hecho uso de las 24 bandas criticas de la escala de Bark, esto implicará que haya tantos 0's como bandas despreciadas en algunos de los bytes. En el caso de nuestra implementación, si estamos monitoreando una transmisión

de 22,000 Hz, significa que tenemos frecuencias de hasta 11,000 hz, que comprenden 22 de las 24 bandas criticas, en este caso, los dos últimos bits de derecha a izquierda de nuestro ultimo byte, siempre estarán en cero, esto debe de ser tomado en cuenta al momento de calcular la distancia. Adicionalmente, en el caso de la transmisión, se debe elegir un tamaño de consulta para llenar un buffer, como se ilustra en la figura 4.2

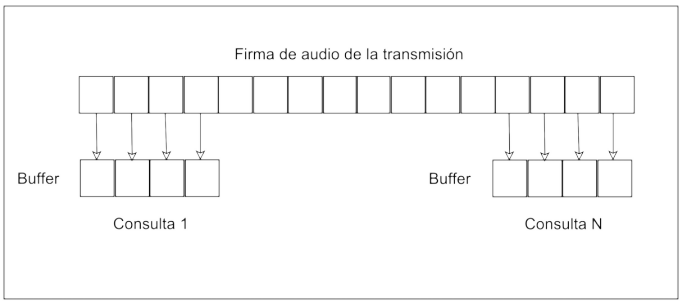

Figura 4.2: Ilustración del buffer a través de la transmisión

El buffer será utilizado para realizar consultas a nuestra base de firmas pre-calculadas en busca de ocurrencias. Este buffer se actualizará cada vez que llegue un nuevo marco, sacando el marco mas antiguo, y metiendo en la ultima posición de este al mas nuevo, generando una nueva consulta, este proceso se ilustra en la Figura 4.2. Los reportes del monitoreo se guardan en archivos de formato csv (comma separated values, o valores separados por coma en español). Para realizar las consultas, se puede realizar de una forma secuencial, o usando índices, esto se abordará mas adelante en esta sección.

4.3.1. Calculo de la distancia

Para calcular la distancia entre dos firmas, se emplea la distancia de Hamming , se compara byte a byte de cada "sub-firma", es decir, si tenemos las siguientes "Sub-firmas" 10000000, 00001111, 00011110 y 10000000, 00101000, 00100000. Al obtener la distancia de Haming en dichos marcos, obtendríamos 0 del primer byte del primer marco, con el primer byte del segundo marco, 4 del segundo byte del primer marco, con el segundo byte, del segundo marco, y 5 del tercer byte del primer marco, con el tercer byte del segundo marco. Eso nos da que hay 9 bits de diferencia entre esos 2 marcos. Si consideramos que la diferencia

de 24 bits implicaría una diferencia del 100 %, una diferencia de 9 bits implicaría un 37 %. En el caso de una firma de audio, se debe de tomar el total de bits en la firma como si este fuera 100 % y en base a los bits diferentes, calcular cual es la diferencia entre las firmas. Por ejemplo, si una firma tiene 10 marcos, y usa las 24 bandas de Bark, el total de bits es 240. En este trabajo, se considerara para fines prácticos que solo se pueden comparar segmentos de igual tamaño. Es decir, no vamos a comparar una firma de 240 bits, con una firma de 480 marcos, en todo caso, se buscaría en que parte de la mas grande coincide la mas pequeña. Se mencionó que la distancia de Hamming podía ser calculada de manera eficiente a nivel de bits con la operación xor, además de que nuestros bits están codificados en 3 enteros de 8 bits, si retomamos el ejemplo anterior, tendríamos los siguientes marcos 128, 15, 30 y 128, 40, 32, al aplicar la operación xor obtendríamos 0, 39 y 62, de primera mano. Estos números representan una codificación en forma de enteros, de las posiciones donde ocurren los bits diferentes. Podemos construir una tabla de tal manera, que los números obtenidos por la operación xor sea posiciones en la tabla, y en dichas posiciones, se encuentre la cantidad de bits diferentes que implica el resultado de esa operación xor, la tabla es construida con el código mostrado en la Figura 4.3

```
        llenaTablaUnos(){
        int i,mask,c;
        dist[0]=0;
        dist[1]=dist[2]=dist[4]=dist[8]=1;
        dist[3]=dist[5]=dist[6]=2;
        dist[7]=3;
        dist[255]=8;
        for (i = 9; i < 255; i++) {
            c = 0;
            for (mask = 1; mask <= 128; mask <<= 1) {
                if ((i & mask) != 0) {
                    c++;
                }
            }
            dist[i] = c;
        }

    }
```

Figura 4.3: Código fuente para el llenado de la tabla de distancias

Al consultar el valor numérico que se encuentra en la posición 0 en la tabla, encontraríamos un 0, en la posición 39, encontraríamos un 4, y en la posición 62, encontraríamos un 5, al sumar todos estos valores, obtenemos un 9, que es el mismo valor obtenido al utilizar una representación binaria de los bits.

4.4. Indexado

Teniendo las firmas que nos interesa monitorear podemos proceder a organizarlas de alguna manera, de tal suerte que cuando necesitemos realizar una consulta, no sea necesario revisar todas las firmas. Para llevar esto a cabo, implementamos los índices men-

cionados en el Capitulo 3, a continuación mencionaremos los detalles de la implementación de cada uno de los enfoques.

4.4.1. Árbol B-K

Para poder indexar nuestros archivos de firmas, estos necesitan tener el mismo número de marcos, para lograr esto, decidimos sacar fragmentos de audio del mismo tamaño para todos los anuncios, en el capitulo de resultados se plantean los tamaños de los fragmentos utilizados.

El Algoritmo 1 es el que se utiliza para construir el árbol B-K, como se puede explicar a grandes rasgos, si el árbol no ha sido iniciado, este tendrá valores nulos, en caso de que sea así, el nodo que se este tratando de indexar será elegido como raíz, si la raíz tiene un hijo en ese rango, entonces se repite el proceso con este hijo como si fuera la raíz, hasta encontrar un nodo que no tenga hijo. Si la raíz no tiene un hijo en ese rango, entonces el nodo es colocado como hijo.

En la búsqueda hicimos la siguiente consideración, nuestro conjunto de datos se agrupa en un rango muy especifico, en donde la diferencia entre un individuo y otro diferente, caen entre 47 % y 53 %. Debido a esto, hemos decidido que el árbol tenga solamente 2 hijos, que comprenden el rango de diferencias de 0 % a 50 % de diferencia, y de 51 % a 100 %. También debido a esto, se ha decidido que al realizar las búsquedas, simplemente se utilice una de las ramas de cada nodo exactamente como si se pretendiera insertar el objeto buscado. Otra consideración es que al comparar la consulta con uno de los nodos, si la diferencia es menor o igual a un umbral determinado, entonces devolveremos el nodo, es decir, no buscamos estrictamente el mas cercano, ni todos los mas cercanos, si no el primero que cumpla la condición descrita anteriormente. Estas decisiones se tomaron para evitar comparar en varios nodos. Si no limitáramos la consulta a una sola de las ramas, acabaríamos revisando todos los nodos del árbol, lo cual haría que fuese inútil tener una estructura de datos.

Algoritmo 1 Algoritmo para hacer el árbol B-K

Require: NODOS $Node_n = node_1 \ldots, node_n$

1: $insertado \leftarrow false$
2: $i \leftarrow 0$
3: $bkt \leftarrow null$
4: **while** $i \leq n$ **do**
5: **if** $bkt.raiz = null$ **then**
6: $bkt.raiz \leftarrow node_i$
7: $insertado \leftarrow true$
8: **else**
9: $tmpNode \leftarrow bkt.raiz$
10: **while** !$insertado$ **do**
11: $rama \leftarrow getRama(nodo_i, tmpNode)$
12: **if** $tmpNode.Rama[rama] = null$ **then**
13: $tmpNode.Rama[rama] \leftarrow nodo_i$
14: $insertado \leftarrow true$
15: **else**
16: $tmpNode \leftarrow tmpNode.Rama[rama]$
17: **end if**
18: **end while**
19: **end if**
20: $i \leftarrow i + 1$
21: **end while**

4.4.2. Arreglo de consulta fija

De igual forma que con el árbol B-K, se uniformiza el tamaño de las firmas para poder utilizar el arreglo de consulta fija. Para implementar esta estructura, solamente es necesario ordenar lexicográficamente de menor a mayor los elementos respecto a los pivotes elegidos. El lenguaje de programación java provee herramientas para ordenar arreglos en su clase `Java.Arrays` de cualquier tipo haciendo uso de un algoritmo de ordenamiento eficiente llamado "Merge sort" [Song06]. En nuestro caso, lo único necesario fue implementar una interfaz "comparador" para nuestro nodo. El nodo utilizado es simplemente la información de la firma de audio, y las distancias a los pivotes. En nuestra interfaz comparador, simplemente hacemos una comparación numérica de los pivotes de 2 nodos, atendiendo el siguiente criterio; Al comparar con el primer pivote, el nodo menor es aquel que tenga la distancia mas pequeña respecto al primer pivote. En caso de empate, se utiliza el siguiente pivote como criterio de desempate. Si resulta que ambos nodos se encuentran a la misma distancia respecto a todos los pivotes, entonces se puede decir que ambos tienen el mismo valor y es irrelevante cual va antes.

Las consultas se realizan utilizando búsqueda binaria para determinar a que rango corresponde el nodo, descartando candidatos en base a los pivotes. De nueva cuenta, en la clase `Java.Arrays` se cuenta con el algoritmo de búsqueda binaria, dicha clase también hace uso de la interfaz comparador tal cual se implementó para el ordenamiento.

4.4.3. Indice invertido

El índice invertido que usamos, está inspirado en el índice que aparece en [Haitsma02], explicaremos primero como se construiría el índice tal cual esta planteado, y mas adelante la variación del índice que es la que utilizamos. En lugar de plantear nuestros marcos como un solo entero que tenga tantos bits como bits tiene el marco, nosotros utilizamos 3 bytes de 8 bits máximo cada uno. Para poder utilizar el índice, es necesario que tengamos tantas entradas en la tabla, como enteros se puedan representar con la cantidad de bits utilizada. Por ejemplo, con 22 bits, se pueden representar 4,194,303 enteros, esto implica tener ese número de posiciones en la tabla. Para convertir nuestra representación de 3 enteros de 8

bits cada uno, a una representación de 1 entero de hasta 24 bits (dependiendo del número de bandas criticas utilizadas) hacemos uso de operaciones de corrimiento, como se ilustran en la Figura 4.4

```
int Pos=(255 & Byte1) << 16 | (255 & Byte2 ) << 8 | (255 & Byte3)
```

Figura 4.4: Ejemplo de operaciones de corrimiento para concatenar bytes

El ejemplo de la Figura 4.4 aplica para números de 24 bits, si despreciamos bandas, tendremos que tenerlo en cuenta a la hora de elegir las mascaras y al momento hacer los corrimientos. Una vez que tenemos los marcos en forma de entero, podemos proceder a barrer cada archivo que tenemos en la firma e ir colocando marco a marco en la tabla, tal como ilustra la Figura 3.10.

Consideraciones de implementación

Podemos notar de la Figura 3.10, que en el archivo "A1" el marco 15 ocurre un par de veces en el mismo archivo. Tal situación ilustrada en un pequeño ejemplo, es común, los marcos ocurrirán varias veces por archivo. En muchos casos, no solo será un par de veces, y dependiendo de la cantidad de marcos, podrían llegar a ser cientos de veces. Esto, y la observación, de que sin importar la posición de dichos marcos en el archivo, en la tabla estos marcos siempre serán contiguos. Esto nos lleva a proponer la siguiente modificación al índice, en lugar de insertar una posición en la tabla por cada marco, insertaremos una posición que contenga un arreglo con todas las posiciones en donde ocurre dicho marco. La primer ventaja que tiene este enfoque, es ahorrar algo de memoria. Mientras que insertar marco a marco en la tabla,hace que se inserte información repetida. Nuestro enfoque permite reducir información repetida. En la figura 4.5 se ilustra la modificación propuesta. El marco 15 ya no genera dos entradas en la tabla, si no están agrupadas en una sola.

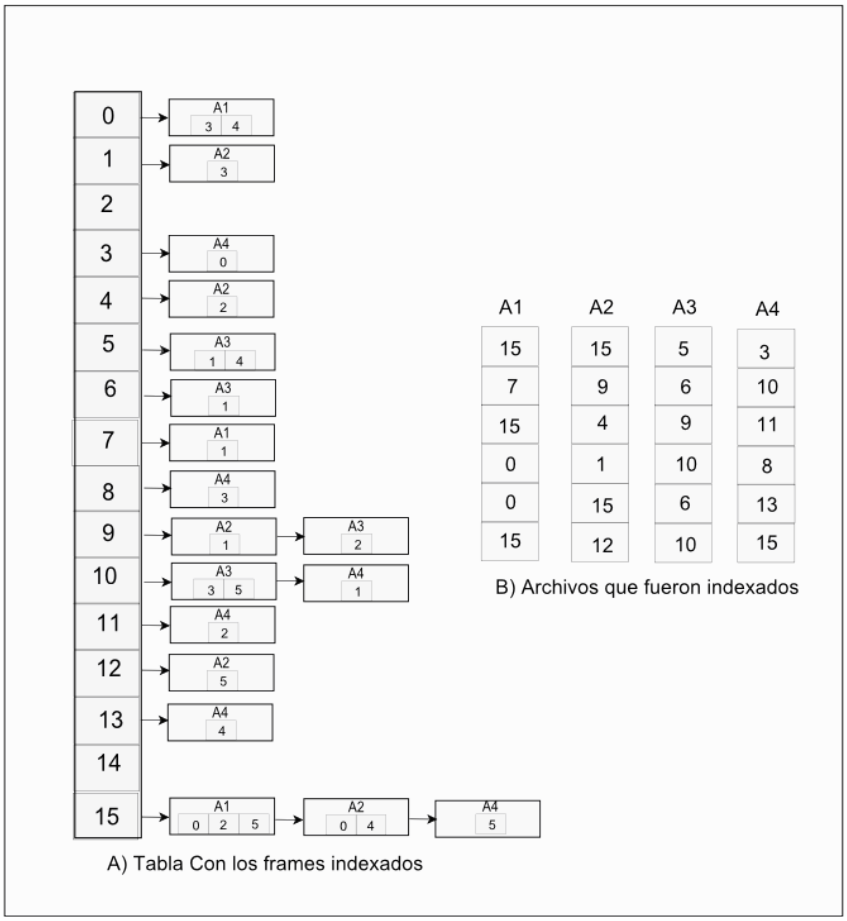

Figura 4.5: Indice invertido con la modificación propuesta

Esto genera que la construcción resulte diferente. Para poder agrupar los marcos y construir el índice de acuerdo a como lo ilustramos en 4.5, tomamos los marcos de la firma, y los guardamos en un arreglo, junto con la posición en donde ocurre cada marco. Ordenamos los marcos en orden ascendente. Esto permitirá que junto a cada marco, tengamos la posición donde ocurre el marco. Procedemos entonces a barrer el archivo de la firma, y guardar las posiciones de los marcos que tengan el mismo valor en un arreglo. Una vez que el marco sea diferente, insertamos ese arreglo en la tabla junto con su referencia a la firma, repetimos este proceso hasta barrer todo el archivo con los marcos de la firma ordenados. Esta forma de construir la tabla nos permite evitar dar varias pasadas a los archivos, lo cual aumenta bastante el tiempo de construcción de esta. El ordenamiento se hace de manera eficiente

con las herramientas que provee Java en su librería `Arrays.sort`

El esquema tradicional de búsqueda en el índice invertido como se plantea en [Haitsma02] consiste en dada una consulta $q = \{15, 9, 9\}$, primero tomamos el primer marco, entonces el marco marcado con el número 15 ocurre en la posición 0 del archivo A1, nuestra consulta es de tamaño 3, esto significa que en A1 hay suficentes marcos para realizar una comparación, la comparación la hacemos con $\{15, 7, 15\}$, si la diferencia no es menor a un umbral t previamente definido, entonces procedemos a consultar si ocurren mas veces ese marco en el archivo, y en efecto, ocurre en su posición 2, eso significa que no hay suficentes marcos restantes en ese archivo para comparar, como ya no existen marcos con ese número en esa entrada (y por ende en ese archivo) pasamos a la siguiente entrada en esa posición de la tabla, ocurre en el archivo A2, nuevamente en la posición 0, permitiendo comparar con A2, específicamente con los marcos $\{15, 9, 4\}$, si de nueva cuenta no es satisfecho este umbral, pasamos a la siguiente entrada. En la posición 15 de la tabla,ya no hay mas entradas, así que avanzamos en nuestra consulta, y obtenemos el marco 9, este marco ocurre en A2, en su posición 1, de igual forma es la posición 1 en nuestra consulta, de tal suerte que la comparación se hace de nueva cuenta con $\{15, 9, 4\}$. Esta forma de llevar a cabo comparaciones se repite hasta que una de las comparaciones satisfaga el umbral, o que ya no se puedan realizar mas comparaciones. Adicionalmente a la modificación en la creación del índice, proponemos un esquema de búsqueda, que permita tomar solo un marco en la posición 0 de cada consulta. Esta modificación funcionó como se esperaba, debido a la gran cantidad de marcos que hay en los archivos (y por ende en la tabla), y permitió hacer búsquedas mucho mas rápidas en el índice como se reporta en el Capitulo de resultados.

4.5. Conclusiones

El sistema esta basado en 3 partes principales de las cuales se mencionaron los detalles en este capitulo y son; la parte de captura, la parte de extracción, y el índice para consultas. La parte de captura simplemente permite recibir el audio en el formato mp3. Este formato es decodificado con ayuda de la librería *javazoom* como se menciono antes. Una vez decodificado el audio, se procesa en frames de un tamaño y traslapes determinados por

el usuario. Cuando se tienen suficientes frames para realizar una consulta, esta se realiza. Tenemos 3 opciones con las que probaremos, el árbol B-K, el arreglo de consulta fija, y el índice invertido. En el siguiente capitulo se explican los experimentos realizados, y los resultados de dichos experimentos.

Capítulo 5

Resultados

5.1. Introducción

El sistema se ha desarrollado para realizar monitoreo tanto en linea como fuera de linea, se han recopilado 83 anuncios en base a grabaciones de programación regular. Estos 83 anuncios han sido sometidos a degradaciones por compresión con pérdida (principalmente debido a la compresión a formato mp3 al momento de guardar el archivo), pero debido a la robustez de la firma de MBSES esto no debe representar ningún problema. Sin embargo, para mostrarque el sistema es confiable de alguna forma, así que también se han grabado 23 horas de transmisiones de una estación de radio en particular. A lo largo de dichas grabaciones ocurrirán en repetidas ocasiones los anuncios del conjunto de prueba. Para establecer en donde ocurren de una forma independiente al sistema desarrollado, se han revisado exhaustivamente las grabaciones,y estos resultados se contrastaran con los obtenidos del sistema, con diferentes traslapes, esto con la finalidad de optimizar este parámetro de la firma. Adicionalmente en [Nieuwenhuizen10] se utilizan fragmentos de 3 segundos de los anuncios para su detección, de igual manera, se presentan los resultados de las pruebas analizando secuencialmente como en el articulo mencionado, adicionalmente se presentan los resultados usando 5 y 7 segundos. El utilizar fragmentos del anuncio para la detección, tiene su fundamento en evitar la superposición que hacen las emisoras de un anuncio con otro, lo que resulta en la degradación del inicio y el final de los anuncios, dejando la parte

central de los anuncios como la menos propensa a dicha alteración. Otra ventaja de usar un fragmento solamente, es que permite que el conjunto de firmas pueda ser indexado con los enfoques del árbol B-K y el arreglo de consulta fija.

5.2. Resultados de Monitoreo fuera de linea

Primeramente vamos a presentar los resultados que se obtuvieron sin utilizar índice alguno.

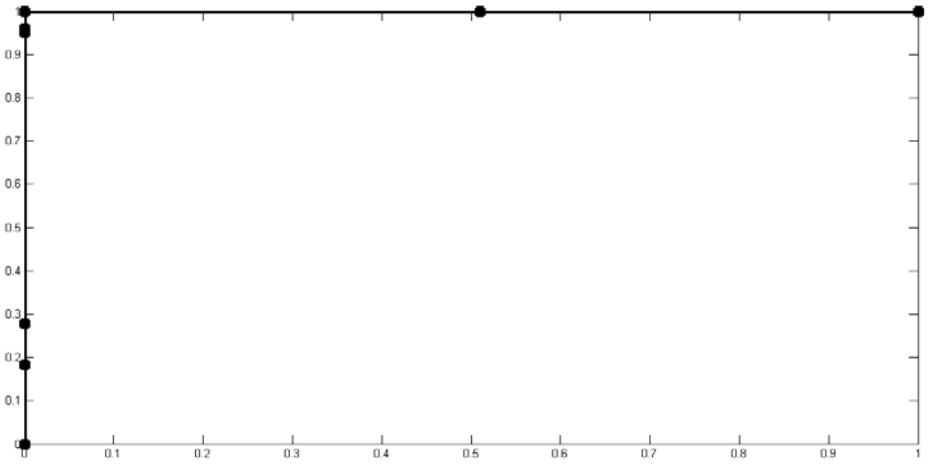

Figura 5.1: Curva ROC para el traslape de 19/20

En la Figura 5.1 podemos observar una curva ROC para el traslape de 19/20, los puntos en la curva representan diferentes umbrales de prueba. Para obtener esta curva iniciamos con un umbral de 0 %, es decir, consideramos que tenía que resultar exactamente la misma firma para poder marcarla como coincidencia, como podemos notar, en este punto no se tuvo ninguna detección. El umbral se fue aumentando en 5 % cada vez, hasta que se llegó al umbral de 50 %, enseguida de ese, se procedió a considerar ocurrencias aún con el 100 % de diferencia, en donde se clasifican correctamente todos los positivos, pero incorrectamente todos los negativos. En la figura no se percibe que en la esquina de comportamiento óptimo se encuentran 3 puntos, correspondientes al 25 %, 30 % y 35 %. Además se encuentran el 40 % y 45 % de igual manera muy cerca, pero ya con algunos falsos positivos (no son perceptibles

por la escala de la figura). Este comportamiento de la curva ROC fue muy similar para los traslapes de 18/20, 17/20, 16/20, 15/20 y 14/20. Que fueron todos los traslapes que utilizamos, de dichas curvas podemos concluir que el umbral que se comporta mejor en todos los casos analizados es 30 %, esto debido a que en el traslape mínimo utilizado,un umbral menor implicaría un falso negativo. De la figura se podría pensar de primera mano que el traslape no tiene importancia en la detección. Esto resulta cierto en términos estrictamente de detección, pero en términos de aplicación la diferencia estriba en el tiempo que se requiere para analizar las 23 horas de audio, siendo un tiempo mucho menor con el traslape mínimo. La información referente al tiempo que toma analizar el audio con los diferentes tamaños de ventana se muestra en la Tabla 5.1

Traslape	Anuncios detectados	Tiempo en minutos
19/20	100 %	338.63
18/20	100 %	90.53
17/20	100 %	47.06
16/20	100 %	23.01
15/20	100 %	15.26
14/20	100 %	11.00

Tabla 5.1: Detecciones logradas usando un umbral de 30 % con una búsqueda secuencial

De la Tabla 5.1 podemos concluir que el menor traslape consigue que sea mas rápido el análisis del audio. Esto se debe a que las firmas resultantes son mas pequeñas, por consiguiente el costo computacional del cálculo de la distancia resulta menor. Si consideramos que el tamaño de frame es 240 milisegundos, el traslape de 19/20 representa procesar un frame cada 12 milisegundos, esto significa que por cada segundo de audio habrá aproximadamente 83 frames, mientras que el traslape de 14/20 hace que se procese un frame cada 72 milisegundos, lo cual hace que por segundo se procesen alrededor de 14 frames. Sin

embargo el que el traslape sea mayor va a permitir reducir la degradación conocida como "Inicio aleatorio". Esta degradación básicamente dice que es imposible que se realice el proceso de extracción de características en el mismo punto exacto en 2 instancias diferentes de audio. La forma en que Jaap Haitsma mitiga el problema en [Haitsma02] es usando un traslape de 31/32.

En este caso estamos tratando con los anuncios completos, cada anuncio tiene una duración diferente, así que no podemos utilizar en este punto el árbol B-K, ni el arreglo de consulta fija . El único indice con el que podemos indexar anuncios completos es el índice invertido (o tabla de búsqueda como le llama [Haitsma02]). Antes de mostrar los resultados del índice invertido, podemos considerar esto, si tenemos una consulta $q = \{1, 3, 4, 2, 6\}$, que es una parte de la transmisión, el índice invertido realizará una consulta por cada frame, intentará buscar cada frame de la consulta en todos los lugares de los archivos en los que pudiera ocurrir. Así se genera una consulta por cada frame. Esto se debe a que en el enfoque original de [Haitsma02] cuando se tiene una consulta, es de un tamaño determinado, y no se tiene mas información que la disponible en la consulta. Sin embargo, en una transmisión, resulta normal que existan mas frames al final de una consulta (para generar una nueva consulta). Dicha situación provocaría que se generaran consultas nuevamente para frames a los que ya se les generó.

Búsqueda consultando todas las posiciones		
Traslape	Anuncios detectados	Tiempo en minutos
19/20	100 %	1436.516
18/20	100 %	198.15
17/20	100 %	38.06
16/20	100 %	16.451
15/20	100 %	8.03
14/20	98 %	4.06

Tabla 5.2: Diferentes traslapes y detecciones logradas usando la tabla de búsqueda

Por ejemplo, en nuestra consulta q anterior, se revisarían 5 posiciones de la tabla de búsqueda. Una vez revisadas, se introducirá un frame nuevo, y saldrá el frame mas viejo. Esto va a provocar que haya nuevas consultas, para frames ya consultados, pero en diferente posición. De la Tabla 5.2, podemos ver como con la resolución mas alta, debido a la gran cantidad de consultas que se generan, se vuelve prácticamente inutilizable, pues resulta casi 4 veces mayor al tiempo que toma realizar las consultas secuenciales. El reducir la resolución a las firmas (reducir el traslape) resulta en menos frames indexados, los frames por segundo son muchos menos tanto en las consultas como en las transmisiones, esto provoca que el computo de distancia resulte menos computacionalmente costoso por cada consulta, por ello en la mínima resolución si existe una mejora en el tiempo, pero podemos además notar que se pierden un par de anuncios.

Si tomamos en cuenta la gran cantidad de frames que se tienen indexados, y la robustez de la firma, podríamos considerar realizar las consultas solamente utilizando el primer frame en cada ocasión, una vez que se actualice la consulta, el siguiente frame sería usado como primer frame, en lugar de realizar las búsquedas como se hace tradicionalmente. Esto evitaría una gran cantidad de consultas permitiendo analizar el audio mucho mas rápidamente y dependiendo del audio tener una taza de recuperación aceptable.

Búsqueda usando solo el primer frame cada ocasión		
Traslape	Anuncios detectados	Tiempo en minutos
19/20	100 %	16.66
18/20	100 %	3.00
17/20	100 %	0.75
16/20	100 %	0.366
15/20	96 %	0.2333
14/20	95 %	0.15

Tabla 5.3: Diferentes traslapes y detecciones logradas utilizando la tabla de búsqueda

De la Tabla 5.3 podemos notar que utilizando solamente el primer frame para realizar la búsqueda a un 4 % de lo que resultaría una búsqueda secuencial con el traslape máximo utilizado y aún teniendo una recuperación del 100 %, en este caso la resolución que mejor funciona es aquella con un traslape de 16/20, ya que con está aún recuperamos todos los anuncios, mientras que tardamos menos de 1 minuto (incluida la construcción del indice) en analizar las 23 horas. Con el traslape mínimo se tarda solo 9 segundos, pero no se detectan 8 de las ocurrencias. Si tenemos en cuenta que la diferencia en tiempo es solo de 13 segundos, y conseguimos 8 detecciones mas, podemos concluir que conviene mas utilizar el traslape de 16/20 para este indice. En la Figura 5.2 (a) podemos ver los resultados de manera gráfica en tiempo con A representando la búsqueda secuencial, B representando la búsqueda tal como se plantea en [Haitsma02] y C utilizando solamente el primer marco de la consulta, en 5.2 (b) podemos ver el porcentaje de detecciones que se logran respectivamente.

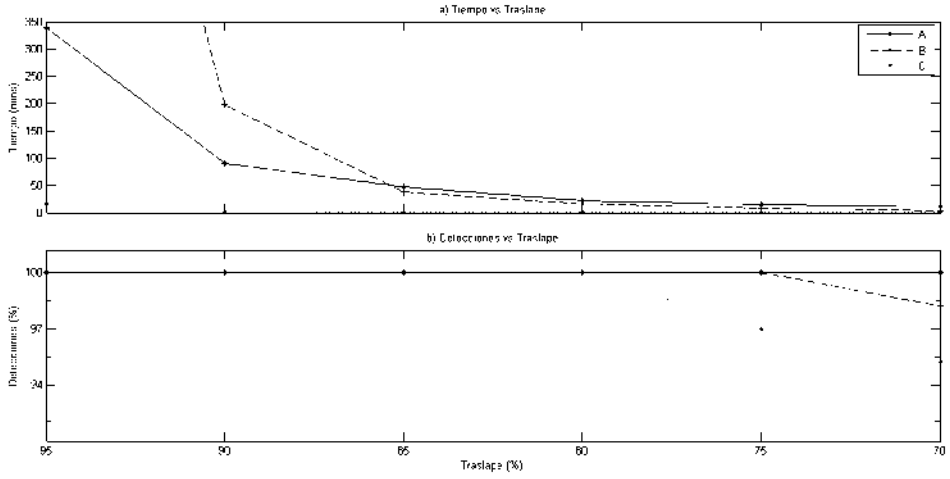

Figura 5.2: Resultados utilizando búsqueda lineal y la tabla de búsqueda

Enseguida se realizaron pruebas uniformizando los tamaños de las firmas con 3, 5 y 7 segundos, esto para poder utilizar los índices de proximidad que hacen uso de los espacios métricos. El reducir el tamaño de los anuncios va a implicar que el cómputo de distancia resulte menos costoso, por tal razón resulta necesario repetir las revisiones secuenciales ahora

buscando los anuncios recortados, esto para saber cuanto tomaría realizar el monitoreo de dicha forma con el costo del cómputo de distancia es menor, ya que las firmas resultantes son de menor tamaño, y compararlo con los resultados obtenidos de usar los índices como son el árbol B-K y el arreglo de consulta fija.

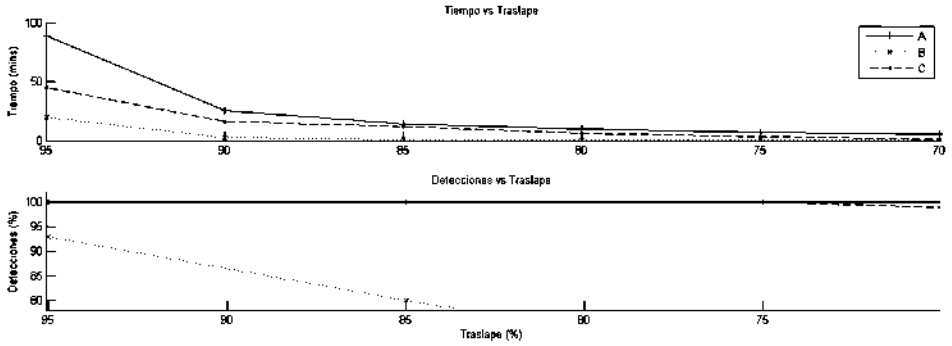

Figura 5.3: Resultados utilizando 3 segundos de audio

De la Figura 5.3, la linea A representa la búsqueda secuencial usando solo 3 segundos de audio. Podemos ver como se reduce el tiempo de consulta respecto a la versión completa de los anuncios, que se encuentran en la Tabla 5.1. Así mismo se observan los resultados obtenidos usando el árbol B-K (linea B) y los resultados con el arreglo de consulta fija. Observamos como en todos los casos el tiempo cae, pero en el caso del árbol B-K las detecciones nunca llegan al 100 %, usando el arreglo de consulta fija, la taza de detecciones comienza a caer después del traslape de 16/20. Sin embargo, en todos los casos tuvimos falsos positivos, situación que no sucede con la versión completa de los anuncios. Además, en el caso del árbol B-K se tiene una cantidad incremental de falsos negativos (partes de la transmisión en donde el sistema clasificó sin coincidencias, que en realidad si tenían). Recordemos en el capítulo anterior describimos con realizaríamos la búsqueda con el árbol B-K para poder ahorrar tiempo, el árbol B-K es una estructura que nos permite una búsqueda por rango, que además explota el concepto de anillos visto en el Capitulo 3. Para llevar a cabo una búsqueda, se define un rango, este rango es lo que define en que anillos de la estructura buscaremos. En determinados casos, puede ocurrir que el o los objetos mas parecidos se encuentren fuera del rango de búsqueda, en cuyo caso, puede ampliarse el

rango para revisar mas anillos. En nuestro caso, al analizar los datos, nos dimos cuenta que la gran mayoría se agrupan en un rango muy especifico, que es 47 % a 53 % de diferencia, el que se agrupen en este rango tan pequeño de distancias, puede ser atribuido a la maldición de la dimensionalidad, o mas recientemente descrita como "El efecto de concentración" en [Pestov12] . Este análisis nos llevó a decidir que para nuestros experimentos, haríamos uso de solo 2 anillos por pivote, de rango de 0 % a 50 % y 50 % a 100 %, al momento de realizar la búsqueda simplemente revisaremos un anillo. Por dicha razón es que no se logra una mayor recuperación. Sin embargo buscar en mas anillos implicaría revisar todos los nodos para nuestras consideraciones particulares. Esto llevaría a no tener ningún ahorro sobre la búsqueda lineal. Respecto a los arreglos de consulta fija (Linea C) el tiempo es menor que una búsqueda secuencial, pero el ahorro no es mucho mayor comparado con la búsqueda secuencial.

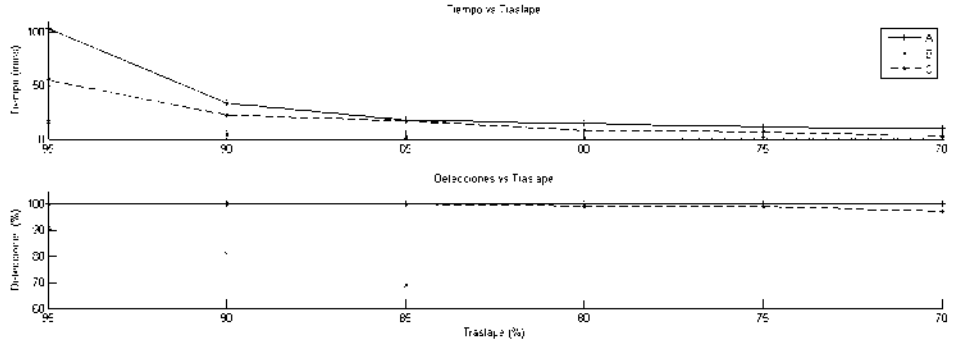

Figura 5.4: Resultados utilizando 5 segundos de audio

En la Figura 5.4 podemos observar los resultados obtenidos de los enfoques lineales (A), uso del árbol B-K (B) y el uso del arreglo de consulta fija (C). El tiempo respecto a usar solo 3 segundos aumenta, lo cual es esperado, sin embargo no tenemos mejores resultados en cuanto a detecciones y en este caso seguimos obteniendo falsos positivos. A continuación se muestran de manera similar los resultados usando 7 segundos.

De nueva cuenta, en la figura 5.5, usando una búsqueda secuencial (A), usando árboles B-k (B) y haciendo uso de los arreglos de consulta fija (C) podemos ver que el tiempo aumento respecto al uso de 5 segundos, en este caso ya no hay falsos positivos sin.

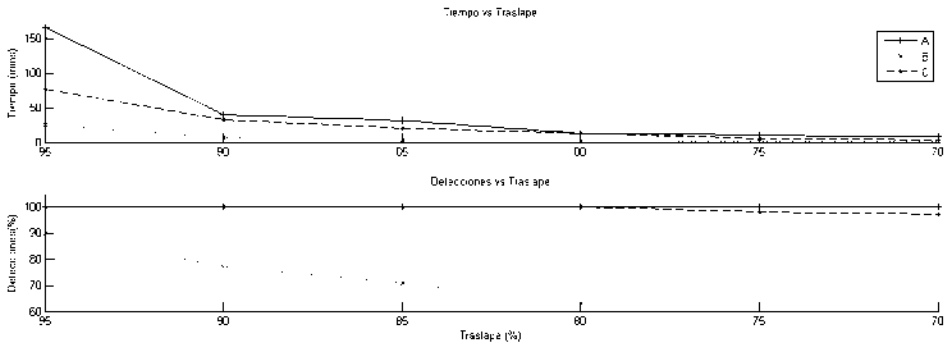

Figura 5.5: Resultados utilizando 7 segundos de audio

El cortar la parte central de los anuncios implica trabajo adicional y los índices que hacen uso de este enfoque no son buenos. Las consideraciones de rango que hacemos tanto como para el árbol B-K como para el arreglo de consulta fija provocan que no se recupere el 100 % de los elementos en la base de datos pero se hacen para tener algún ahorro en las consultas y no acabar revisando toda la base de datos. De los resultados obtenidos, los mas alentadores son aquellos que se obtuvieron haciendo uso del índice invertido mostrados en las Tablas 5.2 y 5.3, ilustrados en la Figura 5.2, dichos resultados nos alientan a llevar a cabo las pruebas de monitoreo en linea utilizando el índice invertido. Además de las ventajas en tiempo de recuperación de dicho índice, este nos permite indexar los anuncios completos, es decir, no necesitamos cortar los anuncios para poder utilizarlo.

5.3. Resultados de Monitoreo en linea

En cuestiones de monitoreo en linea, no tiene caso medir el tiempo que se tarda en revisar la totalidad del audio. En lugar de eso, mediremos el desempeño en términos de que tanto se tarda en realizar una consulta. Para estas pruebas, tomamos 5 horas de nuestras 23. Estas 5 horas representan las que tienen una mayor concentración de spots de audio. Estas 5 horas las retransmitimos y las monitoreamos haciendo uso de las 2 resoluciones que mejor funcionaron, que son 17/20 y 16/20. Además de utilizar el índice invertido, contrastamos contra una búsqueda secuencial.

Los marcos que se procesan al momento de la extracción, son del mismo tamaño,

así que la diferencia en tiempo de cálculo entre un traslape y otro, estriba en la cantidad de marcos que hay que procesar. En nuestras mediciones, pudimos observar que al sistema le toma alrededor de 6.93 milisegundos procesar cada marco. Si tomamos en cuenta que para el traslape de 17/20 se requieren alrededor de 28 marcos para 1 segundo de audio, entonces tenemos que el tiempo necesario para extraer 1 segundo de audio con esta resolución es de 194 milisegundos aproximadamente, mientras que con un traslape de 16/20 se requieren aproximadamente 21 marcos, tenemos que para extraer 1 segundo de audio con esta resolución requerimos 145 milisegundos.

Búsqueda secuencial			
Traslape	consultas realizadas	Anuncios detectados	Promedio por consulta (milisegundos)
17/20	485393	100 %	1.86
16/20	363816	100 %	1.17
Búsqueda tradicional con indice invertido			
Traslape	consultas realizadas	Anuncios detectados	Promedio por consulta (milisegundos)
17/20	485393	100 %	1.003
16/20	363816	100 %	0.555
Búsqueda usando solo 1 marco			
Traslape	consultas realizadas	Anuncios detectados	Promedio por consulta (milisegundos)
17/20	485393	100 %	0.0286
16/20	363816	100 %	0.0200

Tabla 5.4: Diferentes traslapes, detecciones logradas en 5 horas, y tiempo requerido

Los tiempos obtenidos con el índice, resumidos en la Tabla 5.4, sugieren que es posible monitorear varias estaciones de radio a la vez, con una sola computadora, ya que la parte mas costosa computacionalmente del proceso es el cálculo de la firma.

Capítulo 6

Conclusiones

6.1. Conclusiones Generales

El monitoreo automático de audio es un problema que puede ser solucionado de diferentes formas tal y como se menciono en el primer capítulo. En este documento se ha planteado como resolverlo utilizando firmas de audio. Dentro de las diferentes opciones que existen empleando el enfoque de firmas de audio, se ha elegido una opción que ademas de superar el desempeño de otras firmas exitosas, representa una opción que nos permitiría una implementación comercial sin hacer pago de regalías a terceros. Vale la pena señalar que las soluciones actuales que hacen uso de habilidades técnicas, suelen ser muy costosas, y el hecho de que sean soluciones pre-construidas, obliga al usuario final a adaptarse a la forma de trabajar de estas, esta es otra ventaja mas de una implementación como la descrita en este documento, es posible adaptarla a las necesidades del usuario.

Quizá eventualmente se desarrollen mecanismos que permitan a un sistema automático superar la capacidad de entendimiento que posee un humano y se puedan tener las bondades de todos los enfoques de solución presentados en el primer capitulo, pero para el requerimiento actual y con las herramientas exploradas, la solución planteada en este documento es satisfactoria.

La solución propuesta se enfoca en aquellas estaciones que retransmiten su contenido por internet, utilizando específicamente el protocolo shoutcast, dejando afuera a

todas aquellas emisoras que no transmiten por internet, o lo hagan pero no utilicen el protocolo mencionado anteriormente, pero mientras se investigó acerca de los protocolos de transmisión por internet, pudimos notar que es cada vez mas frecuente que las estaciones retransmitan su contenido por dicho medio, y se decanten por este protocolo específicamente. Esto motiva a pensar que eventualmente no sería necesario preocuparse por que existan emisoras que no pongan a disposición sus transmisiones usando la red.

Pudimos también constatar el desempeño de los índices que se probaron, resaltando aquel índice que nos proporciono mejores resultados, resulto sorpresivo que dicho índice sea aquel que se pensaba toleraba menor degradación.

Además una observación que se puede notar, es que mientras un sistema de monitoreo en linea pudiera revisar varias estaciones a la vez, difícilmente se requiere reportar ocurrencias instantáneamente al interesado. Resultando entonces el monitoreo fuera de linea mas eficiente. Esto debido a que un sistema de monitoreo que no requiere hardware especializado, puede ser puesto a funcionar en el momento en que los recursos computacionales disponibles no estén siendo utilizados. Esto permitiría realizar monitoreo con equipo con el que ya se cuenta, siendo que el monitoreo se puede llevar a cabo de manera muy rápida, se pueden revisar varias horas de audio en unos pocos minutos, esto permitiría revisar varios dias de monitoreo en un par de horas. No siendo así con el monitoreo en linea si requiere que el hardware este dedicado a este monitoreo, y siendo en tiempo real, el monitoreo no se puede realizar mas rápido de lo que es capaz de transmitir cada estación. Sin embargo, es posible llevarlo a cabo de ser necesario.

6.2. Conclusiones particulares

1. Pudimos notar que el audio que se transmite por internet no resulta libre de degradaciones, se pensaba al inicio de este trabajo, que el audio no presentaría mayor degradación a la compresión con perdida, y siendo este el caso, la degradación mas preocupante sería aquella conocida como “inicio aleatorio”, esto se pudo constatar no es así. Al llevar a cabo el monitoreo por medio de testigo humano, para poder establecer la confiabilidad del sistema, se notaron distorsiones como ecualización, cambio de

volumen, interrupción de servicio, y en algunos casos ruido de fondo. Esto sin embargo no represento mayor problema debido a la robustez de la firma utilizada.

2. La firma utilizada resulta muy flexible al variarse los parámetros como lo son el tamaño de ventana, el traslape y las bandas criticas que se utilizan, esta flexibilidad nos permitió adaptar la firma a la frecuencia de muestreo a la que sea transmitido el audio a revisar. Además, variar el traslape nos permite reducir el error conocido como "inicio aleatorio", el aumentar o reducir el tamaño de ventana nos permite abarcar mayor cantidad de audio. Hay que señalar que estas variaciones no resultan gratuitas, el aumentar mucho el tamaño de ventana, resulta en una transformada de Fourier mas costosa computacionalmente hablando, el traslape afecta directamente al número de marcos que se procesan, impactando en el tamaño final de la firma, mientras que el uso de bandas criticas esta relacionado con las frecuencias que se toman en cuenta, el despreciar muchas bandas podría finalmente ocasionar que no se cuente con suficiente información para distinguir las diferentes instancias de audio. En audios de corta duración, es conveniente aumentar el traslape para tener la mayor cantidad de marcos posible. El variar esos parámetros va a depender de la aplicación que se le piense dar.

   Algo que fue detectado al analizar las distancias entre las firmas y una característica muy deseable es que cada firma se encuentra muy lejos de las firmas que no son instancias del mismo audio. Esto permite poner umbrales de detección relativamente elevados, sin riesgo a que estos impliquen falsos positivos.

3. Los índices para espacios métricos, como lo son el árbol B-K y el arreglo de consulta fija, mientras que resultan exitosos en muchas aplicaciones, cuando se inicio este trabajo, se creía que el mayor reproche que se haría de estos, serían las medidas tomadas para hacer que el problema se ajustara a un espacio métrico (monitorear fragmentos de igual tamaño sin importar la duración original del anuncio), en este caso pudimos experimentalmente notar que las diferencias se agrupan en rangos muy específicos y pequeños de distancias. Esta situación provoca que sea difícil elegir los rangos que cubrirán los anillos, y hace que los datos se concentren en unos pocos anillos. Esto hace que si se intentan llevar a cabo búsquedas eficientes, se pierdan muchos de los

mas similares, mientras que al abarcar mas anillos, se descartan muy pocas opciones, lo que hace pensar que no vale la pena respecto a una comparación lineal.

4. Se creía también que la instancia del índice invertido implementada tendría resultados aceptables, sin embargo tuvo mejores resultados de los que se esperaban, aún con traslapes mayores que los que se pensó funcionaría, como ya se ha mencionado en gran medida por lo robusta de la firma MBSES. Permitiendo constatar que esta combinación de índice y firma resultan ideales para audio cuyo grado de degradación sea moderado

6.3. Trabajos Futuros

1. Aunque se abordo un protocolo de transmisión por internet ampliamente utilizado, que además tiene la ventaja de transmitir en formatos que de igual manera son ampliamente utilizados, se debe de buscar la posibilidad de monitorear mas protocolos y mas formatos de audio, otros protocolos atractivos para este uso es el RTMP y el RTSP, debido a que existen algunas emisoras que utilizan dichos protocolos, pese a no ser tan populares como shoutcast.

2. Se ha hablado de como en [Camarena06] se supera el desempeño de [Haitsma02] y [Group01], y de como por deducciones a base de experiencias de usuarios hemos establecido nuestra creencia de que de igual forma superaría el enfoque de [Wang03], está creencia estaría mejor respaldada con resultados experimentales, esto hace que surja la necesidad de implementar este último y realizar una comparación de desempeño.

3. Las partes computacionalmente mas costosas del proceso de extracción consisten en la transformada discreta de Fourier, y las copias de datos de un arreglo a otro, mientras que las partes mas costosas del indexado son los cómputos de distancia. Estos procesos resultan altamente paralelizables, este detalle, y el reciente crecimiento de plataformas de computo paralelo que resultan accesibles en costo, como lo pudieran ser las unidades de procesamiento gráfico (GPU). Resulta entonces atractiva la idea de implementar

tanto la extracción de firmas como el proceso de indexado de manera que puedan ser procesadas de forma paralela.

4. Los índices basados en espacios métricos utilizados tuvieron un pobre desempeño para esta aplicación, sin embargo, se podría mejorar este desempeño si se utilizan mejores pivotes. No habiendo ningún criterio de selección de pivotes óptimos, resultaría interesante llevar a cabo un análisis de los datos que permita saber que elementos del espacio métrico resultarían en buenos pivotes, o bien como diseñar "pivotes artificiales" que permitan mejorar el desempeño de dichos índices.

5. Existen además índices como los "K-vecinos mas cercanos" [Tellez11] o los permutantes [Figueroa07] que pudieran tener un buen desempeño al ser utilizados con esta firma, resultaría entonces necesario probar dichos índices.

6. Pese al buen desempeño obtenido por la instancia del índice invertido,no es propiamente una consulta por similitud, resulta evidente que habrá casos en donde no pueda recuperar los individuos mas parecidos, por que se encuentren cambiados los bits de la consulta (por degradación severa) respecto al archivo original. Esto nos hace pensar que una forma para solucionar este problema, y aún contar con la rapidez de este índice, son los enfoques como el hash sensible a localidad, estos permitirán entonces que si se lleve a cabo una consulta por similitud.

Referencias

[Andrew10] Andrew, K. Shazam vs. soundhound: Which song identifier is best? http://androinica.com/2010/11/shazam-vs-soundhound-which-song-identifier-is-best-droid-vs-droid/ 2010.

[Burkhard73] Burkhard, A. W. y Robert, K. M. Some approaches to best-batch file searching. 1973.

[Camarena06] Camarena, A. y Chavez, E. A robust entropy-based audio-fingerprint. *En IEEE International Conference on Multimedia and Expo (ICME2006)*, págs. 1729–1732. July 2006.

[Camarena09] Camarena, A., Chávez, E., y Tellez, E. S. Robust radio broadcast monitoring using a multi-band spectral entropy signature. *En* E. Bayro-Corrochano y J.-O. Eklundh, eds., *CIARP*, tomo 5856 de *Lecture Notes in Computer Science*, págs. 587–594. Springer, 2009. ISBN 978-3-642-10267-7.
URL http://dblp.uni-trier.de/db/conf/ciarp/ciarp2009.html#Camarena-IbarrolaCT09

[CEO11] CEO, S. Apps reconocimiento de música: Soundhound vs shazam. http://somosiphone.com/app-reconocimiento-de-musica-soundhound-vs-shazam, 2011.

[Chávez01] Chávez, E., Marroquín, J. L., y Navarro, G. Fixed queries array: A

fast and economical data structure for proximity searching. *Multimedia Tools Appl.*, 14(2):113–135, 2001.

[Cover91] Cover, T. M. y Joy, T. A. Entropy,relative entropy and mutual information. 1991.

[Figueroa07] Figueroa, K. M. *Indexación efectiva de espacios métricos usando permutantes*. Tesis Doctoral, 2007.

[Group01] Group, M. A. *Text of ISO/IEC Final Draft International Standar 15938-4 Information Technology - Multimedia Content Description Interface - Part 4: Audio*. July 2001.

[Haitsma00] Haitsma, J., van der Veen, M., Kalker, T., y Bruekers, F. Audio watermarking for monitoring and copy protection. *En MULTIMEDIA '00: Proceedings of the 2000 ACM workshops on Multimedia*, págs. 119–122. ACM, New York, NY, USA, 2000. ISBN 1-58113-311-1. doi: http://doi.acm.org/10.1145/357744.357895.

[Haitsma02] Haitsma, J. y Kalker, T. A highly robust audio fingerprinting system. *En International Symposium on Music Information Retrieval ISMIR*. 2002.

[Hellmuth01] Hellmuth, O., Allamanche, E., Cremer, M., Kastner, T., Neubauer, C., Schmidt, S., y Siebenhaar, F. Content-based broadcast monitoring using mpeg-7 audio fingerprints. *En International Symposium on Music Information Retrieval ISMIR*. 2001.

[Huffman52] Huffman, D. A. A method for construction of minimum-redundancy codes. *Proceedings IRE*, 40(9):1098–1101, 1952.

[Jay06] Jay. The shoutcast streaming standard. 2006.
URL `http://forums.radiotoolbox.com/`

[Justin06] Justin, Z. y Alistair, M. Inverted files for text search engines. 2006.

[Kahmsi01] Kahmsi, M. A. y Kirk, W. A. *An Introduction to metric spaces and fixed point theory.* Wiley, 2001.

[Kevin11] Kevin, P. Shazam vs. soundhound: Battle of the mobile song id services. `http://lifehacker.com/5757214/shazam-vs-soundhound-battle-of-the-mobile-song-id-services`, 2011.

[Metois99] Metois, E. Audio watermarking and applications. 1999.

[Nakamura02] Nakamura, T., Tachibana, R., y Kobayashi, S. Automatic music monitoring and boundary detection for broadcast using audio watermarking. *En SPIE*, págs. 170–180. 2002. doi:10.1117/12.465274.

[Neubauer98] Neubauer, C. y Herre, J. Digital watermarking and its influence on audio quality. 1998.

[Nieuwenhuizen10] Nieuwenhuizen, H. A. W. C. V. y Grobler, L. M. The study and implementation of shazam's audio fingerprinting algorithm for advertisment identification. 2010.

[Nullsoft99] Nullsoft. Shout cast directory. `www.shoutcast.com`, 1999.

[Pestov12] Pestov, V. Indexability, concentration, and vc theory. *J. Discrete Algorithms*, 13:2–18, 2012.

[Press92] Press, W. H., Teukolsky, S. A., Vetterling, W. T., y Flannery, B. P. *Numerical Recipes in C: The Art of Scientific Computing. Second Edition.* Cambridge University Press, 1992.

[Shannon49] Shannon, C. y Weaver, W. *The Mathematical Theory of Communication.* University of Illinois Press, 1949.

[Song06] Song, Q. Merge sort algoritm. 2006.

[Tellez11] Tellez, E. S., Chávez, E., y Navarro, G. Succinct nearest neighbor search. *En SISAP*, págs. 33–40. 2011.

[Wang03] Wang, A. An industrial strength audio search algorithm. *En ISMIR*. 2003.

[Zezula05] Zezula, P. y Giuseppe, A. *Similarity search, the metric space approach*. Springer, 2005.