Use of Tonic and Raga as Indices for a Query by Example based Music Information Retrieval System

Rajeswari Sridhar*, Hamsini Krishna Kumar, Harini Selvaraj and Abinaya Parthasarathy
Department of Computer Science and Engineering, Anna University, Chennai – 600025, Tamil Nadu, India; rajisridhar@gmail.com, hamsinikk@gmail.com, harinigem@gmail.com, abinayapartha@gmail.com

Abstract

Background and Objectives: A Music Information Retrieval Systems deal with retrieving music from a corpus based on user’s input. Query by Example (QBE) and Query By Humming (QBH) are the two content base music information retrieval systems. Methods/Statistical Analysis: In this paper, a Query By Example (QBE) based Music Information Retrieval system is proposed, where the characteristics of Carnatic music are used as indices for identifying similar matches to an input song. The system begins by introducing a novel focused crawler mechanism that explores the World Wide Web in a methodical manner to harvest music for the database required. The crawled music is then downloaded, and a local repository of songs is created. Carnatic music features like Tonic and Raga are extracted from the crawled songs and these songs are indexed into the database. Tonic identification is a challenging problem and in this work, a new algorithm for estimation of tonic is designed. Using the Tonic, the Raga of the song is identified and these features are used in a modified, optimized version of the Multi-key hashing technique to index the songs and improve the speed of retrieval. During a user query, which is by example, the same features are extracted, compared with the features of the songs indexed in the database, based on the search option provided by the user, and the best matching songs are retrieved, by proposing a new ranking algorithm based on Raga and Tonic similarity. Findings: The proposed system recognized the Tonic of the song with accuracy while the Raga of the song is recognized if the input music piece is free of Gamaka. Application/Improvements: Music information retrieval system for entertainment, therapy and helps in content based retrieval of music and the system could be improved to handle Gamakas in the input music piece.

Keywords: Focused Crawler, Music Information Retrieval, Multi-Key Hashing, Query By Example, Raga, Tonic

1. Introduction

Multimedia content on the World Wide Web (WWW) has increased tremendously over the years. Crawling the www has become a challenging task, with the www being such a vast information resource. Traditional crawling methods consume a lot of resources and bandwidth, and this requires modification of the existing approaches to suit the multimedia application proposed. The concept of ‘Focused Crawling’ considers only those pages that are related to the topic of search, thereby saving a lot of time and resources required for expanding irrelevant sites. Information Retrieval (IR) is the process of obtaining the information required from a rich tagged database of information resources. Music Information Retrieval (MIR) is an extension of IR where the information required is based on the musical characteristics of the input audio. In a typical MIR system, the metadata of the music file is tagged along with the music piece. Hence, the MIR is performed similar to text based retrieval. An alternative method for MIR is to extract musical features from the input music. Hence, to efficiently retrieve music files, the input queries can be in the form of example files or humming. In this paper, an MIR System based on Query By Example (QBE) is proposed.

In the process of retrieval, the musical characteristics of the input are used. Raga is one of the main characteris-
tic features of songs in the context of Indian music\(^3\). This concept of Raga is absent in Western music. Hence, the MIR for western music typically relies on features like pitch and melody, for similarity match\(^3\).

A raga in the context of Indian music, be it the South Indian Carnatic or the North Indian Hindustani style, is characterized by the specific arrangement of notes\(^4\). There are seven notes characteristic of Indian Classical music, namely, S, R, G, M, P, D, N. Arohanam is generally described as the ascending arrangement of notes and Avarohanam is the descending arrangement of notes. A raga defined in Carnatic music differs from its Hindustani counterpart, in the style of singing, arrangement of notes, time-scheme for notes, etc. In Carnatic music, a parent-child relation exists. The parent ragas, known as Melakartha Ragas, contain all the seven swaras and are 72 in number\(^5\). Child ragas or Janya Ragas, of each Melakartha Raga, are a derivation of the Melakartha raga, and typically contain fewer than seven swaras (a subset of the swaras in the Melakartha Raga). The frequencies of all the swaras in a raga have a fixed relationship with the frequency of the middle octave ‘S’ known as the Shadja or tonic\(^6\). Hence, prior to identifying the raga of the song, it is necessary to estimate the tonic frequency. In this work, the Raga and Tonic features of Carnatic music are used to index the music files, and also as features for retrieval later.

This paper is structured as follows: Section 2 provides a glimpse of the related work in the area of web crawling, raga and tonic identification and indexing techniques used in MIR. Section 2 explains the architecture of the proposed system, with details about the contributions to each of the modules. Section 3 describes the dataset used along with the constraints imposed. Section 4 discusses the results obtained and the evaluation of these results. Section 5 summarizes the conclusions drawn, and gives ideas on how the project can be expanded.

The system proposed in this work, requires features such as Raga and Tonic for indexing, a crawler for harvesting music files and an efficient indexing algorithm. A literature survey was performed of the existing work on these topics.

A thorough analysis of the existing work on focused crawling was carried out, to get a clear idea of the robust and efficient techniques that can be adopted to crawl the www for music files. Soumen Chakrabarti et al.\(^1\) paved the way for the development of a focused crawler, which expands only those links that are relevant to the predefined topics, and discards the irrelevant links. In this system, the importance of a page given as input to the crawler is determined by the classifier, which makes relevance judgments to seek out only those pages that are relevant to a category of topics. A distiller identifies the hubs that are potential resource pages, pointing to many relevant links known as authorities. The classifier is built using a probabilistic model and the Bayes rule, while the distiller is implemented, using the Hypertext Induced Topic Search (HITS) algorithm proposed by Kleinberg\(^7\). Taylan D, et al.\(^8\) introduced an improvisation in focused crawling where the relevant pages, which can be reached through links from irrelevant pages, are also taken into account while crawling. Instead of discarding the irrelevant pages, a tree level is chosen and the links in the irrelevant page are expanded, until this specified tree level is reached. To decide the relevance, words likely to occur in pages relevant to the pre-defined topic along with their weights, are maintained in a topic words weights table. Pages are scored using the similarity between their content and the topic words weights table. They propose a URL optimization method which inserts only those links whose score is above a certain threshold value into the URL Frontier. Qu Cheng, et al.\(^9\) brought in the prediction of page relevance into the focused crawler mechanism, using information from pages which are already crawled. The architecture of the system they propose involves the construction of a topic feature vector, for pages crawled using the topic constructor. The topic relevance of a page is a function of the number of links that point to the page from the chosen seed URLs, the number of links pointing to the seed URLs from the page and the similarity of the feature vector of the page to the topic feature vector. If this value is above a certain threshold value, the page is considered to be relevant. A Link forecaster module extracts the URLs from the parsed pages, and predicts the topic relevance using the information describing the hyperlinks in the current page.

So far, focused crawlers have been implemented only for text based retrieval. In this system, we have tried to modify focused crawling by making the relevance decision based on the presence and absence of music files. We have tried to make use of Meta words in the website source to determine the possibility of the presence of music files, for faster classification of the site as relevant or irrelevant. The idea of using a classifier and a distiller in our system has been adapted from the work of SoumenChakrabarti\(^1\). The classifier in our system makes use of a meta-word scoring table, similar to the topic words weights table\(^8\), to
make decisions on site relevance, and the distiller adopts the URL optimization technique.

Understanding the fundamental terms and concepts related to Carnatic music was essential for this work. T M Krishna et al. define the abstract concept of raga and its various components swara, scale, gamaka and phrases - in their entirety within the aesthetics of Carnatic music. Some of the existing methods for identifying Tonic and Raga, the features that are to be used as indices in this system, were analyzed. Rajeswari Sridhar et al. discuss the technique of using mutation to find the tonic of a song. The Sa Pa Sa signals, which characterize different Shruts or scales in Carnatic music are stored in a local database. The input signal is mutated at three positions - the beginning, the middle and the end - with all the Sa Pa Sa files and features such as the Mel Frequency Cepstral Coefficients (MFCC), the flux and the centroid are extracted for both the input song as well as the mutated signals. The Euclidean distance is computed between the features of the mutated signals and the features of the input song. The Sa Pa Sa file for which this distance value is the minimum for all three mutated signals, gives them the frequency of the Shadja which is the frequency of the tonic of the input song. Ashwin Bellur et al. introduce the technique for detecting the tonic pitch value by processing the pitch histograms of pieces in Indian classic music. The group delay function is used to amplify certain traits of the pitch histogram so as to enhance accuracy in tonic determination. In this work, three different strategies are used to detect the tonic. The concert method exploits the fact that the tonic is constant over pieces in a concert, whereas the template matching method and segmented histogram methods use the following properties: 1. The tonic is always present in the background, 2. Some notes are less inflected and dominant, to detect the tonic of individual pieces.

Rajeswari Sridhar et al. discuss an algorithm for raga identification of Carnatic songs. A fitness function is used to identify the Tala of the input song. The beginning and the end of a Tala cycle can be found by detecting the onset and offset of the vocal signal, and each cycle is then sub divided into the individual Tala segments. The swaras of each segment are determined using the dominant frequencies and using these swaras, the Arohanam and Avarohanam of the song’s raga are determined. Subsequently, the Arohanam and Avarohanam obtained are compared with those of the ragas in the database, using a string matching algorithm, and the raga of the song is identified. Rajeswari Sridhar et al. suggest an alternative method to find the raga of a song using the signal level features and the raga lakshana characteristics of the song. The tonic is found using the algorithm and the swaras of the songs are determined using the technique described. A raga model stores the Arohanam and Avarohanam, the different raga lakshana characteristics such as Graha, Nyasa etc., the swaras of every raga which can take pitch inflections (gamakas), the characteristic phrases and the typical range of signal level features such as MFCC, Flux, Centroid and Carnatic Interval Cepstral Coefficients (CICC) for each raga. The characteristic phrases in the song are identified using the KMP string matching algorithm. The raga of the song is first determined using the Arohanam and Avarohanam of the ragas in the Raga Model, and then using the signal level features, and finally using the raga lakshana characteristics. The raga which is identified using more than one of the three steps of the Raga identification process is considered to be the raga of the input song. The mutation algorithm has a few restrictions, like the requirement of aalaap in the input song and also requires a long running time. The concert method is applicable for a collection of pieces and not for individual ones. While the template matching method works on constraints, such as the presence and absence of certain swaras, the segmented histogram method relies heavily on the presence of the drone of the mridangam. To overcome these limitations, a modified version of the pitch histogram concept is proposed in this work where we make use of swara probability, using the Kernel Density Estimation (KDE) function that helps in the identification of the tonic frequency as well as in the detection of the other swaras present. This information can directly be used for raga identification as well. The Tala based segmentation yields inaccurate results when the tempo of the input songs varies significantly. In this work, two alternative means of segmentation have been put forth for higher accuracy. The three-pronged approach for raga identification requires a compromise on time complexity as there are a large number of comparisons to be made. Therefore, we restrict the comparisons by comparing just the swaras present in the input song to those present in different ragas.

The indexing technique is very essential in MIR, as it will reduce the query’s response time. In dual-ternary indexing, proposed by Chang CW et al., the songs are represented in the form of pitch contour, and the pitch content is represented using a two dimensional grid
Use of Tonic and Raga as Indices for a Query by Example based Music Information Retrieval System

and three number notation. Yang XH et al. describe an inverted index approach, to organize and index music collections. According to the musical form structure and repetition rule of musical themes, repeating patterns can be regarded as representations of musical themes. The statistical result indicates that repeating patterns follow Zipf’s law on a given music database. Music melodies are segmented into phrases based on a musical dictionary composed of repeating patterns, and added into an inverted index with the necessary musical information. The experimental results reveal that the proposed indexing method outperforms the compared method greatly, with much less storage space. Rajeswari Sridhar et al. propose the Multi-key hashing technique for indexing, which are a combination of chaining and a variation of re-hashing. A song is generally characterized by a lot of features, and each of these features can be used as a key to retrieve the song in this technique. Features such as melody string, flux, centroid and MFCC, considered are used as keys. On comparison with the dual-ternary indexing technique, it was observed that Multi-key hashing based retrieval had a lesser time complexity. The algorithms were also compared for their precision and recall in which Multi-key hashing had a better recall than modified dual-ternary indexing for the sample data considered. Another disadvantage of dual-ternary indexing is that it uses a fixed segmentation algorithm and is, therefore, unable to handle variable length queries. In this system, we have modified the existing Multi-key hashing technique by introducing an optimization and indexing using features such as Melakartha number, Raga Name, Tonic range and Singer’s Name. Even though this technique has a higher space complexity when compared to the work of Chang C W and Yang X H, it has the advantage of querying for songs based on any feature, without compromising on the response time.

2. Architecture

Figure 1 shows the conceptual model of the system. The color legend indicates the modules that have been contributed to in this work. The algorithms are explained in the following section. This system’s functioning can be partitioned into the offline and online phases.

- In the offline phase, the www is crawled for music files in the domain under consideration, using a Focused Crawler. All the crawled music files are downloaded and stored in a database. The stored songs are first pre-processed and given as input to the Feature Extractor. The musical characteristics of the song such as tonic, raga and signal level features such as MFCC, Flux and Centroid and Meta information such as the singer’s name are extracted from the crawled songs. In order to improve the speed of retrieval during the song search, the downloaded music files are indexed based on the extracted features, using Multi-key Hashing.
- In the online phase, the user selects an input query song. The user can also specify the attribute based on which he expects song similarity - the raga or the singer. The input query song is also pre-processed, and all the features extracted from the crawled songs are extracted from the query song in a similar manner. The features of the input song are then compared with the features of the songs stored in the database, and the songs which are similar to the query song based on the specified attribute, are retrieved, ranked and displayed to the user.

The contribution in this paper to the modules involved is explained in detail below:

2.1 Focused Crawler

The focused crawler is used to explore the WWW in a methodical manner, to harvest music files and create a rich music repository for MIR. The concept of focused crawling has been adapted to make the crawler as efficient as possible, and avoid areas of the web which are irrelevant. Figure 2 describes the working of the designed
focused crawler. There are four main modules: Classifier, Link Expander, Distiller and Similar Sites, whose functions are explained below.

Existing focused crawlers have been implemented by managing the hyperlink exploration process such that the content text of the crawled pages is relevant to the search query. Focused crawlers predict the probability that an unvisited page will be relevant before actually downloading the page, using predictors like anchor text.

The focus in MIR is not to expand pages with text content related to music but to expand web sites which actually contain a considerable number of music files in the domain under consideration by the system (Carnatic and Indian film music). In this case, we found META HTML tags to be a very good predictor. (More often than not, a page with Meta tags ‘down-load’, ‘mp3’ and ‘carnatic’ will give us useful music files). We can also ensure that faster decisions are made about the relevance of a website if we use the meta-words as opposed to the conventional method of using page contents. We had to develop a scoring model which would give appropriate weight to the META tags found while handling issues like spelling differences for proper nouns and spelling errors for common nouns. The scoring model developed, using the sites that are used as the crawler’s seed URLs is described in equation (1):

\[
\text{score} = \frac{A \times \text{Number of occurrences of the meta word in all seed URLs}}{B \times \text{Number of seed URLs} + C \times \text{Number of seed URLs containing the meta word}}
\]

The following example justifies the use of the equation (1) for computing the score of a Meta word.

Case 1: Say we have 10 seed URLs (B), and the word ‘music’ appears 20 times across all URLs (A) and that there are 1-2 occurrences of ‘music’ in each seed URL. (C = 10)

Case 2: Say one of the seed URLs is a website dedicated to a particular singer ‘X’. The word ‘X’ may occur 25 times in the Meta tags of just that page and may not occur in any of the other seed URLs. Here, A=25 B=10 C=1.

Doing just A/B will give us 20/10 = 2 (for ‘music’) and 25/10 = 2.5 (for ‘X’). We end up getting a higher score for ‘X’ just because of duplicate occurrences in one site and a lower score for ‘music’ which is clearly the more important word as it occurs in all of the seed URLs. This is why we multiply by C.

The score of a website is the sum total of the score of all the Meta words in the source of the root page. Websites with a score greater than a threshold, computed as the average of the page scores of the seed URLs, are identified as relevant by the classifier, and are subsequently expanded. Since different sites use different spellings for the same Meta word (especially proper nouns), the Apache Lucene Spell-checker Library (version 3.0.1)[26] is used to suggest the most similarly spelt word in the database to the input Meta word, using the N-gram method and Levenshtein distance.

The link expander expands a site classified as relevant and mines all the music files available within the site. The Meta information required for each song is retrieved from the tags in the music file. To propel the crawler in the right direction, 20 sites similar to each site classified as relevant, returned by an existing API[22], are added to the Frontier Queue.

Distiller logic is adapted to identify whether or not each relevant site is a potential hub, pointing to other relevant websites. A good hub is one which has outlink to many relevant websites. Outlinks from good hubs are added to the frontier queue, and are subsequently expanded.

![Focused crawler architecture](image)

Figure 2. Focused crawler architecture.

2.2 Pre-Processor

A pre-processor is used to pre-process the input query song as well as the songs in the database before feature extraction. If the given input query song or the songs crawled are not in a format supported by the system, they have to be converted to the domain specific format. The song then needs to be fed into a signal separation block where the input song should be separated into voice and music signals. The Feature Extractor requires only the
vocal portions of the song. For accurate feature extraction, the gamakas or pitch inflections in the voice signal need to be removed or handled to the best possible extent. The pre-processed song signal, thus obtained, is divided into several segments using either fixed segmentation or edge detection, and these segments are given as input to the Feature Extractor. Using fixed segmentation, the song is divided into smaller segments of the same length. The duration of each segment does not exceed 0.25 seconds when we use this method, so as to ensure that just one note is being sung in each particular segment. Using edge detection, a filter is created to detect the edges or the points of frequency change in the song under consideration. Figure 3 shows the edges detected for a given input song. Here, each segment is of variable length and there is no restriction on the duration of each segment.

Figure 3. Segmentation using Edge detection.

2.3 Feature Extractor

The Feature Extractor is used to describe short segments of recordings in a robust way. After pre-processing, both the query song as well as the crawled songs in the database is passed to this module for feature extraction. A feature set - the tonic or the chosen frequency of the Shadja, the raga, the singer’s name and signal level features, such as MFCC, flux and centroid is chosen to uniquely identify each song.

2.4 Tonic Identifier

For tonic determination, the dominant frequency of the signal, in each segment of the pre-processed input song, is estimated using YIN. In another work, the use of pitch histograms yielded fairly accurate tonic frequency values. However, there were several constraints on the input. In this work, the idea of using a pitch histogram has been modified to include probability. Since the KDE function is a non-parametric method to estimate the probability density function of a random variable, it has been used in this system for tonic identification. Here, the random variable for the probability density function takes the values of the dominant frequencies present in the segmented input song. Kernel Density Estimation fits the need of the system, due to its simplicity and its ability to process a large set of input values. Local maxima or peaks in this function graph correspond to the frequencies of interest as they have a high probability of being present in the input song. As the first step, local maxima with a very low probability are eliminated. The remaining frequency candidates are narrowed down to one, using the properties of the Shadja. Every song (usually sung over three octaves) necessarily contains two frequencies of the Shadja - the Shadja of the middle octave and that of the higher octave which are typically available at a ratio of 1:2. So, the frequencies that do not have a visible peak at twice the frequency are eliminated. After the first two steps, frequency pairs are further narrowed to one using the property that the rendition of most songs dominates the middle octave. Figure 4 shows the graph for the KDE function for an input song in the raga Abhogi. The peak which represents the tonic and the last circled peak correspond to the two frequencies of the Shadja. The algorithm used to identify the frequency of the tonic is summarized below:

The algorithm works based on two assumptions:

- The Shadja of the middle octave occurs more frequently than notes with frequency higher than the Shadja of the higher octave.
- The Shadja of the higher octave occurs more frequently than notes with frequency lower than the Shadja of the middle octave.

Figure 4. Tonic identification using KDE.
However, this is typically true for any song in the domain considered (Carnatic music and Indian film music).

**FindTonic(song)**

begin
segments <- segment(song;segmentation_method)
for i<- 1 : length(segments)
begin
    data(i) <- yin(segments(i))
end
y <- kde(data)
peaks <- findpeaks(y)
count<-1
for i<- 1 : length(peaks)
begin
    for j <- i+1 : length(peaks)
    begin
        if((peaks(i):xvalue*2) ≈ peaks(j).xvalue)
        begin
            candidate_tonic(count) <- [peaks(i):xvalue;peaks(j):xvalue]
            break
        end
    end
    count<- count+1
end
[tonic;tonic_pair]<- find_first_extreme_peaks(candidate_tonic)
end

**find_first_extreme_peaks(freq)**

begin
i<- 1
j<- length(freq)
while(1)
    if(freq(i;1) x 2 = freq(j;2))
    break
end
    if(peak_height(freq(i;1))>peak_height((freq(j;2)))
    j <- j-1
else
    i<- i+1
end
end
return freq(i;1);

The accurate determination of the frequency of the middle octave Shadja or the tonic is key for the success of the MIR system proposed. We analyzed the performance of edge detection vs block based segmentation to be sure that the method we use gives us as accurate a tonic value as can be obtained. Due to pitch inflections or gamakas being prevalent in Carnatic music, we found that the segments obtained were too small, when edge-detection was used. This resulted in YIN algorithm yielding a lot of inaccurate values for dominant frequency in each segment. Since the KDE graph was plotted on the basis of these values, tonic identification was also inaccurate.

When we used fixed segmentation, the sizes of the segment were big enough for YIN to yield fairly accurate values although the start and end points of each segment didn't have any musical significance. We experimented with fixed segmentation with equal segments of 0.1s and 0.25s to determine which would yield better results as discussed in Section 4.

2.5 Raga Identifier

The tonic identified in the previous module gives us the frequency of the Shadja'S' belonging to the middle octave. The peaks visible in the KDE are located at the frequencies of the swaras that are present. In Carnatic music, there are three types each of R (Rishabha), G (Gandhara), D (Dhaivatha) and N (Nishadha) and two types of M (Madhyama). P (Panchama) and S (Shadja) are invariant and do not take Gamaka. The height of each peak is proportional to the probability of the corresponding frequency present in the input song. We define 'Confidence Factor’ as the value taken by the KDE function for each frequency and this is used to model the fact, that higher the peak height, higher is the probability of the frequency being present. The ratio of each frequency with high Confidence Factor to that of the tonic frequency is determined, and is mapped to a swara. Table 1 gives the ratios of the frequency of each of the swaras with that of 'S'. In Figure 4, the peak representing the tonic frequency and the circled peaks, represent the middle octave frequencies of the swaras present in the raga Abhogi (the raga of the input song), namely, S, R2, G2, M1, D2, S. In each of the R, G, M, D and N families, if more than one member has high Confidence Factor, the member with the highest Confidence Factor is declared present.
Use of Tonic and Raga as Indices for a Query by Example based Music Information Retrieval System

Table 1. Swaras and their ratios with ‘S’

<table>
<thead>
<tr>
<th>Swara</th>
<th>Ratio</th>
<th>Swara</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>P</td>
<td>R1</td>
<td>D1</td>
</tr>
<tr>
<td>R1</td>
<td>D1</td>
<td>R2</td>
<td>D2</td>
</tr>
<tr>
<td>R2</td>
<td>D2</td>
<td>R3</td>
<td>D3</td>
</tr>
<tr>
<td>G1</td>
<td>N1</td>
<td>G2</td>
<td>N2</td>
</tr>
<tr>
<td>G2</td>
<td>N2</td>
<td>G3</td>
<td>N3</td>
</tr>
<tr>
<td>M1</td>
<td>N3</td>
<td>M2</td>
<td>SA</td>
</tr>
<tr>
<td>M2</td>
<td>SA</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In Carnatic music, there are twelve semitones (smallest musical intervals) in an octave: S, R1, R2=G1, R3= G2, G3, M1, M2, P, D1, D2=N1, D3=N2, N3.

A Melakartha raga must necessarily have S and P, one of the M’s, one each of the R’s and G’s, and one each of the D’s and N’s. Also, R must necessarily precede G and D must proceed N. This gives 72 Melakartha ragas (2 x 6 x 6)\(^5\). A Raga Model containing these 72 Melakartha Ragas along with the swaras present in each raga is maintained\(^2\). The swaras detected in the input song are compared with the swaras present in each of the ragas in the database, and the raga with the most number of matches is determined to be the Raga of the input song. If the raga of the input song is a Janya raga (lesser than 7 swaras), the raga detected is expected to be the parent Melakartha raga. The Melakartha number of the raga identified is also returned for indexing purposes. If the Raga Model is expanded to include Janya Ragas, the Melakartha number will help us identify the parent Melakartha raga.

2.6 Signal Level Features and Meta Information

The signal level features such as the MFCC, centroid and flux are extracted from the input query song as well as the songs in the database\(^4\). The name of the singer is the Meta information required for indexing, and this information is obtained from the Meta tag of the input query song or from the Meta tag of the crawled files.

2.7 Indexing

The technique of Multi-key Hashing is justified for multimedia indexing and retrieval since a multimedia piece is characterized by more than one media and each of the media can be a key\(^6\). In our system, we have considered this algorithm for indexing songs. A Carnatic music song is characterized by various characteristics, among which we consider four features to be used as keys - the Melakartha number, raga name, tonic range and singer’s name. As the first step, an index is computed using a predefined hash function with the first key, the Melakartha number. All songs sung in the Melakartha Raga, as well as songs sung in any of its Janya Ragas hash to this location. In case of collision, the song is chained at the first index, and a second index is computed using the second key - the Raga Name and the same hash function. In this case, just the songs sung in the particular Raga can be found at the index computed. Another collision prompts chaining, followed by the use of the third key. The tonic range, to which the singer’s tonic belongs, is determined and an index is computed. Two tonic ranges, one from 120 Hz to 145 Hz and one from 180 Hz to 240 Hz are used. The limits chosen for the two ranges effectively helps to group the songs into two - songs sung by male singers and those sung by female singers. The final key used for indexing in case of collision, is the singer’s name. All the crawled songs are indexed in this manner and stored. The order in which the keys are chosen is determined, based on their ability to limit the possible candidates for a match, their uniqueness and their robustness. An optimization was attempted on the Multi-key hashing technique, to enhance the retrieval speed. In case of collisions at all indices to which the song hashes, it is necessary to traverse any one of these chains to find the exact match. The chain with the shortest length among the chains the song hashes to is kept track of during retrieval, to minimize the number of misses during retrieval.

2.7.1 Retrieval

During retrieval, the first element at all the indices computed, using the four features is checked in the same order, used during storage. If an exact match is not found, the first chain, namely, the chain at the index computed using the Melakartha number of the song as key, is traversed. As the number of songs stored increases, the number of songs that map on to this index may also increase. This is because, songs that are sung in the Melakartha Raga as well as songs that are sung in one of its Janya ragas, hash on to this index. In such a scenario, one of the subsequent chains may be of a shorter length, and traversing it will help reduce the number of misses. In our system,
the Multi-key hashing indexing technique is optimized to keep track of the chain length at each index. In case the exact match is not found by checking the first elements at the indices computed using all four keys, the chain with minimum length is traversed, considerably reducing the number of misses.

2.8 Similarity Matcher

As the focus of our system lies in finding similar music, in addition to retrieving the exact song, there is a need to define similarity in Carnatic music. This module identifies ragas similar to the raga of the input song, and helps retrieve songs sung in the similar ragas, in addition to songs sung in the raga of the input song. A logical grouping of ragas based on similarity is achieved by adopting the concept of Chakras. Melakartha ragas are divided into 12 Chakras.

All the ragas in a particular Chakra have the same member of the Ri and Ga family, and differ only in Da and Ni. The Chakra of the input raga is first identified, and the remaining five ragas which are present in the same Chakra are considered to be similar. This is done when the user wishes to find similar songs based on the Raga. If similar songs are to be retrieved in comparison with the singer of the input song, all songs hashed into the index computed, using the singer's name of the input song are retrieved.

2.9 Ranker

The retrieved songs are ranked based on two metrics. The first metric quantifies the raga similarity based on the number of swaras that are present or absent in the song retrieved, compared to the song retrieved to the input song. It is obtained by proposing a new relationship which is defined by equation (2).

\[ \text{Raga similarity} = 2 \times n(A \cap B) - n((A \cup B) - (A \cap B)) \quad (2) \]

where, \( A \) is the set of swaras present in the input song and \( B \) is the set of swaras present in the song retrieved and \( n(X) \) indicates the number in \( X \).

The number of swaras that are common between the input song and the retrieved is an important factor that tells us about the similarity in raga between the two. The first term denotes this and is multiplied by a weight of 2 to give it more importance.

The similarity in raga between the input song and the retrieved song becomes less if:

- There are swaras present in the input song that are absent in the retrieved song.
- There are swaras present in the retrieved song that are absent in the input song.

The second term \( n((A \cup B) - (A \cap B)) \) denotes this. The songs which have same value for the similarity metric, are further ranked based on the tonic distance, i.e., the difference between the input song's tonic frequency and the retrieved song's tonic frequency. This helps in ranking songs sung in similar voices, higher than the rest, when the raga similarity score is equal.

3. Dataset and Constraints

The dataset consists of 846 songs. Out of these, 461 songs are songs downloaded, using the newly designed focused web crawler from the WWW. Some examples of the seed URLs considered are www.ragasurabhi.com, www.no1tamilongs.com, etc. The remaining 385 songs are synthesized, using pureswaras with no gamaka (or) pitch inflection for testing and evaluation purposes. The features of the songs like tonic, Raga name, Melakartha Number, singer's name and signal level features such as MFCC, flux and centroid are extracted and stored in the database. The set-up of this method involves a Query By Example (QBE) as against a Query By Humming (QBH) system.

Some of the constraints imposed for correct functioning of the focused crawler are:

- The presence of Meta words in the site source is a must.
- The Meta words should describe the site correctly.
- Similar sites are found only for the sites present in the database used by the API.
- The number of calls that the API allows is limited.
- The link expander assumes that all music files present in the website classified as relevant are within the domain under consideration.

For accurate tonic and raga identification, the following constraints exist:

- The input file should be in the Waveform Audio File Format.
- This module assumes that most of the song is sung in the middle octave, as is usually the case.
Use of Tonic and Raga as Indices for a Query by Example based Music Information Retrieval System

Typically, an input query song of a length of two minutes or more satisfies this constraint.

- The presence of musical instruments in the background could lead to incorrect tonic identification.
- The input song should be a solo piece i.e. performed by one singer.
- The input song should not be a ragamalika which means a mixture of ragas.
- The presence of gamakas or pitch inflections in a swara increases the likelihood of the adjacent swaras being shown as present. The algorithm works best for songs with a flat rendition of notes without inflections (as in the case of instruments).
- Information about the raga of the input song should be present in the database used. If not, the most similar raga present is used.
- In case of Janya Ragas, the name of the parent Melakartha Raga is shown as output.

For singer identification, the name of the singer provided in the Meta tag is assumed to be correct. During retrieval, ragas in the same chakra as that of the input song are assumed to be similar to it. If the two features used for ranking, i.e., the tonic and the swaras present are the same for two or more songs, the songs are ranked arbitrarily.

4. Results and Analysis

Keeping in mind the constraints explained in the previous section, the system was tested with the input data set, and the analysis of each module proposed, is discussed in the following section, based on the results obtained.

4.1 Analysis of Focused Crawler

The focused crawler is designed to crawl the WWW in such a manner, that it considers only the websites that are related to Music. The use of Meta words in the site source for classification helped a great deal in ensuring, that the focused crawler meets the above requirements, but this method also had its own demerits.

The analysis of the focused crawler was done by manually tagging the first 1000 sites crawled as:

- Sites accepted correctly: Sites crawled, which contain music files belonging to the domain considered.
- Sites accepted incorrectly: Sites crawled, which do not contain music files, or contain music files outside the domain considered.
- Sites rejected correctly: Sites rejected, which do not contain music files, or contain music files outside the domain considered.
- Sites rejected incorrectly: Sites rejected, which contain music files belonging to the domain considered.

The crawler retrieves thousands of songs if we allow it to execute fully and takes a long time to terminate but we use only a small subset of the songs retrieved as our dataset to analyze our algorithms.

<table>
<thead>
<tr>
<th>Measure</th>
<th>Equation</th>
<th>Value obtained</th>
</tr>
</thead>
<tbody>
<tr>
<td>Harvest Ratio</td>
<td>( \frac{\text{Number of sites crawled}}{\text{Time}} )</td>
<td>141.1 sites / hr</td>
</tr>
<tr>
<td>Relevance Ratio</td>
<td>( \frac{\text{Number of sites accepted correctly}}{\text{Total number of accepted sites}} )</td>
<td>0.551</td>
</tr>
<tr>
<td>Goodness Ratio</td>
<td>( \frac{\text{Number of sites accepted correctly}}{\text{(Number of sites accepted correctly + Number of sites rejected incorrectly)}} )</td>
<td>0.623</td>
</tr>
</tbody>
</table>

Figure 5. Correctness of focused crawler decisions.

Table 2. Focused Crawler evaluation metrics
sites crawled. In this work, we introduce three metrics for the purpose of evaluating the modified focused crawler.

The sites accepted correctly by the focused crawler yielded a considerable number of songs and a high proportion of the files belonged to the domain under consideration (Indian Music). However, a major drawback here is the loss of relevant websites, due to the poor programming practice of avoiding the Meta tag in the source codes. A number of sites, that are in fact relevant, are rejected by the focused crawler, due to the lack of Meta words contributing to the average goodness ratio. The average relevance ratio can be attributed to sites, with just music-related text content, such as song reviews, and sites that provide song lyrics.

### 4.2 Analysis of the Tonic Identification Algorithm

The tonic is the frequency of the middle octave note “S” or “Shadja”. The algorithm proposed, identifies the frequency of this note in the musical piece given as input. The tonic varies depending on the characteristics of the singer’s voice and the scale or Shruti chosen by the singer to render the song. (It remains more or less constant for a chosen singer as they usually tend to sing in the same Shruti.) To evaluate the accuracy of the tonic frequency determined by the algorithm, we manually isolated the middle octave “Shadja” in several musical pieces rendered by the singer (who sang the 461 music pieces retrieved from the web) and determined its frequency using Audacity. The tonic frequency determined was approximately 194 Hz. This was compared with the mean tonic frequency determined by the algorithm using the three segmentation methods.

Tonic values for the dataset were determined using all the three segmentation methods, i.e., fixed segmentation of 0.1 seconds, and 0.25 seconds, and edge detection. Table 3 shows the various evaluation parameters of tonic accuracy, and the values obtained for each of the parameters. The deviation from the mean tonic value was used to determine the most effective segmentation method. The lower the standard deviation, the more effective is the segmentation method.

Table 3 shows that the mean tonic using a fixed segmentation of 0.25 seconds is marginally closer to the tonic of the singer as compared to the mean tonic using a fixed segmentation of 0.1 seconds. However, the standard deviation is the least, when the fixed segmentation of 0.1 seconds is used for segmenting the input songs. Figure 6 shows the variation in tonic for randomly selected songs from the dataset, using the three segmentation methods. An almost perfectly straight line is observed for a fixed segmentation of 0.1 seconds.

<table>
<thead>
<tr>
<th>Evaluation Parameter</th>
<th>Segmentation Method</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean Tonic</td>
<td>Edge Detection</td>
<td>186.0219</td>
</tr>
<tr>
<td></td>
<td>0.1 sec fixed</td>
<td>191.3317</td>
</tr>
<tr>
<td></td>
<td>0.25 sec fixed</td>
<td>192.1165</td>
</tr>
<tr>
<td>Standard deviation from Mean tonic</td>
<td>Edge Detection</td>
<td>53.99275</td>
</tr>
<tr>
<td></td>
<td>0.1 sec fixed</td>
<td>23.62453</td>
</tr>
<tr>
<td></td>
<td>0.25 sec fixed</td>
<td>30.00482</td>
</tr>
</tbody>
</table>

This method of determining tonic and tonic accuracy is scalable across all Indian music. However, it would require some prior preprocessing such as removal of gamakas and separation of voice and music signals and this is outside the scope of our project (as is proved by analyzing 385 synthesized pieces using the voices of 4 different singers). The number of sites on the web dedicated to Carnatic music is small. Therefore, we analyzed 461 pieces by one singer due to the fact that this was the subset of songs retrieved by the crawler that conformed to the input constraints of the MIR system (such as absence of musical instruments and absence of gamakas).

![Figure 6. Variation in tonic using the segmentation methods.](image-url)
ber of songs for which the tonic value was inaccurate, to evaluate the three segmentation methods. The value was found to be low for the 0.1 seconds segmentation. Therefore, a fixed segmentation of 0.1 seconds is chosen as the method of segmentation, for the determination of the Tonic and other frequency components in the input.

Table 4 gives us a thorough comparison between the algorithm proposed for tonic identification and an existing algorithm that uses the technique of mutation.11

Table 4. Comparison between KDE based and Mutation based tonic identification

<table>
<thead>
<tr>
<th>KDE based</th>
<th>Mutation based</th>
</tr>
</thead>
<tbody>
<tr>
<td>No constraints on the structure of the input song or the portion to be used for analysis</td>
<td>Requires aaalaap (gradual exposition of a Raga in slow tempo) in the beginning.</td>
</tr>
<tr>
<td>Determines the frequency value of the Shadja accurately</td>
<td>Compares against the 17 predefined Shadja frequencies and estimates the closest.</td>
</tr>
<tr>
<td>Computes just the Kernel Density Estimation for the set of dominant frequencies in the segmented input song. Average running time ( \approx 2 ) minutes</td>
<td>Computes three features (MFCC, Centroid and flux) for the input song, which are to be compared with three mutated signals for all 17 possible shadjas. Hence time consuming. Average running time ( \approx 45 ) minutes</td>
</tr>
<tr>
<td>Is heavily affected by presence of gamakas or pitch inflections</td>
<td>Not affected by gamakas</td>
</tr>
</tbody>
</table>

4.3 Analysis of Raga Identification Algorithm

We developed a metric to evaluate the performance of the raga identifier, by taking into consideration the distance between the swaras determined and the swaras present. Table 5 illustrates the evaluation metric used for the Raga Identification algorithm. The total score is calculated using our devised equation,

\[
\frac{\text{Index of Closest Swara Determined} - \text{Index of Actual Swara}}{\text{Length of Actual Song}}
\]

Equation 3 gives us a normalized value between 0 and 1, which depicts the accuracy of the swaras found when compared to those swaras actually present. Table 6 gives us the values for the mean raga accuracy score calculated, using each of the three segmentation methods, and Figure 7 shows the standard deviation of the raga accuracy score from the mean. The mean value of the Raga Accuracy Score for the 385 songs synthesized without gamakas was as high as 0.99464, and the standard deviation from the mean value was just 0.03128. This justifies the reasoning that raga accuracy is lost due to the presence of gamakas, and that handling gamakas will help in correct raga identification for unsynthesized songs.

Table 5. Raga accuracy score

<table>
<thead>
<tr>
<th>Index of Closest Swara Determined – Index of Actual Swara</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>±1</td>
<td>2</td>
</tr>
<tr>
<td>±2</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 6. Mean Raga Accuracy Score

<table>
<thead>
<tr>
<th>Segmentation Method</th>
<th>Mean Raga Accuracy Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edge Detection (Crawled Songs)</td>
<td>0.82</td>
</tr>
<tr>
<td>0.1 sec Segmentation (Crawled Songs)</td>
<td>0.83</td>
</tr>
<tr>
<td>0.1 sec Segmentation (Synthesized Songs)</td>
<td>0.99</td>
</tr>
<tr>
<td>0.25 sec Segmentation (Crawled Songs)</td>
<td>0.83</td>
</tr>
</tbody>
</table>

The comparable values of the mean score, using the fixed segmentation of 0.1 seconds and 0.25 seconds, and the lower mean value of segmentation using edge detection, further justifies the use of a fixed segmentation of 0.1 seconds for pre-processing and feature extraction. For evaluating the raga identifier, songs for which the tonic could not be determined were omitted, but songs with both correct as well as incorrect tonic were considered. As a result, for a few songs, an incorrect tonic determination percolated to this module, resulting in incorrect raga identification leading to a low similarity score.

The correctness of the Raga also justifies the correctness of the tonic. Only if the frequency of the tonic is accurately determined, can the presence or absence of other notes be determined correctly (as their frequencies have a fixed relationship to the tonic frequency).

Raga accuracy is determined using the Raga Accuracy Score (computed using equation 3).

From Table 6, we can conclude that:
- A Raga accuracy score of around 0.82 implies highly accurate tonic identification for crawled songs (considering gamakas or pitch inflections...
contribute to most of the incorrect Raga identifications).

- A Raga accuracy score of 0.99 implies almost perfect tonic identification for the synthesized songs (no gamakas).

### 4.4 Analysis of Indexing Technique

The worst case space complexity of this algorithm is $O(n)$. After indexing the 846 songs in our dataset, the space complexity was found to be $O(3.542654028 \times n)$.

![Figure 7. Standard deviation of the Raga Accuracy Score.](image)

As hashing is used for storage, the time complexity is $O(1)$. Figure 8 shows the example of a scenario where the optimized Multi-key hashing technique can be used to save the number of misses. When songs belonging to a Melakartha Raga (here, Harikambhoji) and a Janya Raga of the Melakartha Raga (here, Mohanam) are chained, they index on to the same location, using the key Melakartha Number (here, 28) but to different locations, using the key Raga Name. Using the existing approach, if a song is not present as the first element of any of the chains, the first chain is traversed to find it.

Using the optimized approach, the shortest chain is traversed instead of the first.

![Figure 8. Misses during Retrieval of Songs.](image)

Table 7 shows the number of misses before finding an exact match using each approach for the above example.

#### Table 7. Number of Misses

<table>
<thead>
<tr>
<th>Song Name</th>
<th>Raga Name</th>
<th>Number of misses using original approach</th>
<th>Number of misses using optimal approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>Song A</td>
<td>Harikambhoji</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Song B</td>
<td>Mohanam</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Song C</td>
<td>Harikambhoji</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Song D</td>
<td>Mohanam</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>Song E</td>
<td>Harikambhoji</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>Song F</td>
<td>Mohanam</td>
<td>5</td>
<td>2</td>
</tr>
</tbody>
</table>

### 5. Conclusion and Future Work

We have built a Music Information Retrieval system based on QBE, which retrieves songs similar to the input query song, based either on the raga or the singer. The focused crawler used in the project efficiently filters out areas of the web that are not related to music, and do not contain domain-specific music files for downloading. The tonic identification algorithm proposed by us, gives highly accurate values of tonic frequency, and works on considerably fewer constraints on the musical characteristics of the input song, as compared to the existing methods. The time taken for tonic identification is also much lower. Once the tonic is identified, raga identification can be done in an almost constant time, as the same information used for tonic identification is required for raga identification also. The similarity matcher makes use of the grouping of the Melakartha ragas into Chakras to find similar ragas. The Multi-key hashing indexing technique is optimized to keep track of the chain length at each index. By traversing the chain with the minimum length, the number of misses before finding an exact match is considerably reduced.

Further work on the focused crawler could help make it immune to the lack of Meta words in the page source. The textual content of the page can be processed in the absence of Meta words. The relevance of web pages can also be predicted using information from the pages with outlinks to it. An efficient signal separation technique is required to filter the non-voice frequencies present in the input file before using the tonic identification algorithm, for identifying the frequency of the Shadja. The gamaka is defined as the meandering of a swara between the adjacent swaras. The peculiarity of this gamaka is that the
pitch value or frequency of the swarasthana (pitch position) is not specially sounded, but the swara is sung as an oscillation between the notes adjacent to it, before and after the swara. Gamakas are required to be removed so that the rendition of notes in the song is at. However, the presence of gamakas characterizes certain swaras, and their removal could lead to the loss of crucial information. There are multiple types of gamakas, and an analysis of these could help handle a few of the types. However, the full essence of the gamakas cannot be eliminated, and it is required to live with the loss of system efficiency due to this. Characteristics of Carnatic ragas other than the swaras present (such as the raga lakshana characteristics) can be used for identification. An attempt can be made to reduce the space complexity of Multi-key hashing to improve the efficiency as the application of MIR system involves entertainment, therapy, etc.

6. References


27. Min S. The effect of musical activities programs on parenting efficacy and resilience of mothers with preschool children. Indian Journal of Science and Technology. 2015 Apr; 8(s.7):650-6.