Brute Force Algorithm for Finding The Key of a Song in Music

Ridho Akbarisanto / 13511005
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13511005@std.stei. itb.ac.id

***Abstract*—Finding the music key of a song can be done easily for some people. But, for some other people it's difficult to find the right key. To find the key we can use many methods. One of the method is to check all the possibilities of the music key. This method is often called brute force. This method can be done by checking the standard music chord for each key and compare to the song chords. The brute force algorithm can be really effective for this problem because the constraint is small.**

***Index Terms*—brute force, music key, music chord, song.**

I. INTRODUCTION

Music is an art form whose medium is sound and silence. Its common elements are pitch (which governs melody and harmony), rhythm (and its associated concepts tempo, meter, and articulation), dynamics, and the sonic qualities of timbre and texture.

Music nowadays is often represented as song. A song can be played by its harmonic melody, and by playing the chords. We can play the chords in many music instruments, such as guitar, keyboard, etc.

In this era, we can find the chords of a song freely in the internet or by asking our friend who knows music. Every song can be played in a music key. Sometimes, when we play a song with our guitar, our friend suddenly asks us, "in which key are you playing the song?" and sometimes we can answer this question, and sometimes we can't.

If we examine carefully, we can get the pattern in the music and if we use this pattern we can get the key of the song. Because of the small constraint, the brute force algorithm will be enough to solve this problem by finding the right key of the music.

II. THEORIES

2.1. Music Theory

2.1.1. Sound

Music theory describes how sounds, which travel in waves, are notated, and how what is sounded, or played, is perceived by listeners. The study of how humans interpret sound is called psychoacoustics, while the conitive aspects of how perceived sounds are interpreted into musical structures is studied in music cognition. In music, sound waves are usually measured not by length (or wavelength) or period, but by frequency.

Every object has a resonant frequency which is determined by the object's composition. The different frequencies at which the sound producers of many instruments vibrate are given by the harmonic series. The resonators of musical instruments are designed to exploit these frequencies. Different instruments have different timbres due to variation in the size and shape of the instrument as well as the choice of materials from which the parts of the instrument are constructed.

A note is generally perceived as a sound on a single pitch. Notes have a regular wave beat on the eardrum that humans (and perhaps other animals as well) find pleasing. This may be in part due to the fact that from the moment the hearing function becomes available to an unborn child, there is the regular rhythm of the mother's heartbeat.

Often the fundamental aspects of sound and music are described as pitch, duration, intensity, and timbre.

2.1.1. Pitch

Sounds can be classified into pitches, according to their frequencies or their relative distance from a reference pitch. Tuning is the process of assigning pitches to notes. The difference in pitch between two notes is called an interval. Notes, in turn, can be arranged into different scales and modes. The most common scales are major, harmonic minor, melodic minor, and pentatonic.

The Key of a piece of music determines what frequency each note is played at. A piece in the key of D major will put all the notes two semitones higher than a piece in the key of C major. Changing the key can change the feel of the piece of music dramatically, as it changes the relation of the pitches of the composition to the pitch range of the instruments on which the piece is being performed, often effecting timbre as well as having other more technical implications for the performers. However, key changes may also go unrecognized to the audience, as changing the key does not (by definition) change the relation of the pitches of the composition to each other, and so different keys can in many cases be considered

equivalent and a matter of choice on the part of performers (this is especially true for popular and folk musics).

2.1.2. Rhythm

Rhythm is the arrangement of sounds in time. Meter animates time in regular pulse groupings, called measures (or bars in British English). The time signature specifies how many beats are in a measure, and which kind of written note is counted and felt as a single beat. Through increased stress and attack (and subtle variations in duration), particular tones may be accented. There are conventions in most musical traditions for a regular and hierarchical accentuation of beats to reinforce the meter. Syncopated rhythms are rhythms that accent parts of the beat not already stressed by counting. Playing simultaneous rhythms in more than one time signature is called polyrhythm.

In recent years, rhythm and meter have become hot topics among music scholars. Recent work in these areas includes books by Fred Lerdahl and Ray Jackendoff, Jonathan Kramer, Christopher Hasty, William Rothstein, and Joel Lester.

2.1.3. Melody

Melody is the unfolding in musical time of a principle single line. This line can be sounded alone, unaccompanied; or it can be the top (or sometimes an inner) note of a sequence of chords, or sounded against chords as a background by accompanying instruments or voices. Melodic rhythm is usually rooted in the accent patterns of language, and/or the animating rhythms of dance steps and forms.

In much of Western music, melody is often the most identifiable theme. Melodies will often imply certain scales or modes. Counterpoint is the study of combining and layering more or less independent melodies.

2.1.4. Harmony, Consonance, & Dissonance

Harmony can generally be thought of as occurring when two or more pitches are sounded simultaneously, although harmony can be implied when pitches are sounded successively rather than simultaneously (as in arpeggiation). Harmonies involving three or more pitches sounded simultaneously are referred to as chords, though the term is generally used to indicate an organized selection of pitches rather than just any three or more pitches.

Consonance can be roughly defined as harmonies whose tones complement and augment each others' resonance, dissonance as those which create more complex acoustical interactions (called 'beats'). Another manner of thinking about the relationship regards stability; dissonant harmonies are sometimes considered to be unstable and to "want to move" or "resolve" toward consonance. However, this is not to say that dissonance is undesirable. A composition made entirely of consonant harmonies may be pleasing to the ear and yet boring because there are no instabilities to be resolved.

Brief audio (MIDI) musical examples of the interaction and effect of consonance and dissonance upon each other can be found here: "The effect of context on dissonance"' and here: "The role of harmony in music".

Melody is often organized so as to interact with changing harmonies (sometimes called a chord progression) that accompany it, setting up consonance and dissonance.

"Harmony" as used by music theorists can refer to any kind of simultaneity without a value judgment, in contrast with a more common usage of "in harmony" or "harmonious", which in technical language might be described as consonance.

2.1.5. Music Chord

A chord in music is a group of (typically three or more) notes sounded together, as a basis of harmony.

These groups of notes need not actually be played together: arpeggios and broken chords may for many practical and theoretical purposes be understood as chords. Chords and sequences of chords are frequently used in modern Western, West African and Oceanian music. In many other parts of the world music doesn't need chords at all.

The most frequently encountered chords are triads, so called because they consist of three distinct notes, but another notes may be added to give seventh chords, extended chords, or added tone chords. The most common chords are the major and minor triads and then the augmented and diminished triads. The extension, "major", "minor", "augmented", and "diminished" are sometime referred to collectively as chordal "quality". Chords are also commonly classed by their root note so, for instance, the chord C Major may be described as a triad of major quality built upon the note C.

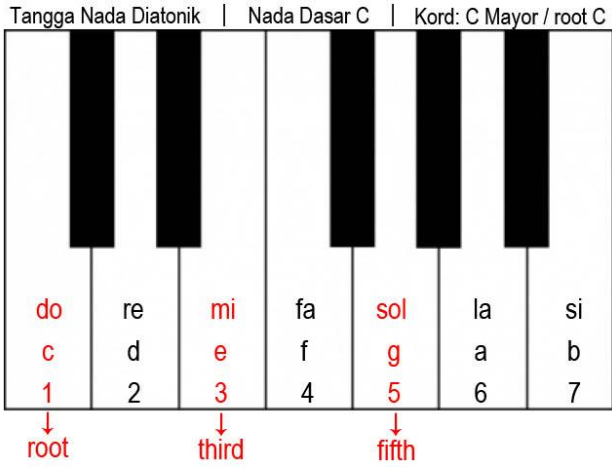
A series of chords is called a chord progression. Although any chord may in principle be followed by any other chord, certain patters of chords have been accepted as establishing key in common practice harmony. In order to describe this, chords are numbered, using Roman numerals upwards from the key-note. Common ways of notating or representing chords in western music other than conventional staff notation include Roman numerals, figured bass (much used in the Baroque era), macro symbols (sometimes used in modern musicology), and various systems of chord charts typically found in the lead sheets used in popular music to lay out the sequence of chords so that the musician may play accompaniment chords or improvise a solo.

The English word *chord* derives from Middle English *cord*, a shortening to *accord* in the original sense of "agreement" and later "harmonious sound". A sequence chords is known as a chord progression or harmonic progression.

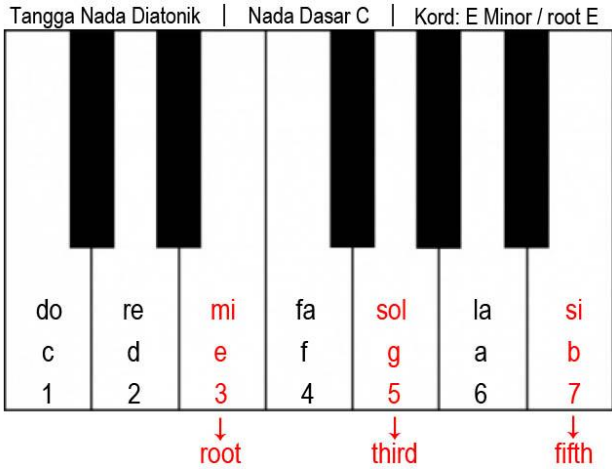
In most genres of popular music, including jazz, pop, and rock, a chord name and the corresponding symbol are typically composed of one or more of the following parts:

1. The root note (e.g. C).
2. The chord quality (e.g. major, maj, or M).
3. The number of an interval (e.g. seventh, or 7), or less often its full name or symbol (e.g. major

seventh, maj7, or M7).
4. The altered fifth (e.g. sharp five, or #5).
5. An additional interval number (e.g. add 13 or add13), in added tone chords.



Picture 2.1 C Chord in C Key



Picture 2.2 E Chord in C Key

For instance, the name C augmented seventh, and the corresponding symbol Caug7, or C+7, are both composed of parts 1, 2, and 3.

None of these parts, except for the root, directly refer to the notes forming the chord, but to the intervals they form with respect to the root. For instance, Caug7 is formed by the notes C-E-G#-Bb. However, its name and symbol refer only to the root note C, the augmented (fifth) interval from C to G#, and the (minor) seventh interval from C to Bb. The interval from C to E (a major third) sets the chord quality (major). A set of decoding rules is applied to deduce the missing information.

Chord qualities are related with the qualities of the component intervals that define the chord. The main chord qualities are:

- Major, and minor.
- Augmented, diminished, and half-diminished.
- Dominant.

Some of the symbol used for chord quality are similar to those used for interval quality:

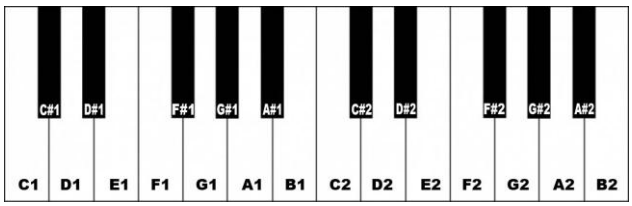
- M, or min for minor.
- M, maj, or no symbol.
- aug for augmented.
- dim for diminished.

2.1.6. Music Key

In music theory, the key of a piece usually refers to the tonic note and chord, which gives a subjective sense of arrival and rest. Other notes and chords in the piece create varying degrees of tension, resolved when the tonic note and/or chord returns. The key may be major or minor, although major is assumed in a phase like "this piece is in C." Popular songs are usually in a key, and so is classical music during the common practice period, about 1650-1900.

In music, the key identifies the tonal center or home base of a song. This tonal center is a note that the whole song revolves around. Every note in that song will be corresponded to the home base note. For example, if a song is in the key C, then every note in the song will be corresponded towards C.

There are 12 possible keys any particular song can be played in. 12 is the number of notes that can be played on the piano keyboard or other music instruments. They are A, A#/Bb, B, C, C#/Db, D, D#/Eb, E, F, F#/Gb, G, and G#/Ab. Every song can be played at all of these keys



. Picture 2.3 Keyboard Key

In the picture above there are 2 octaves of keyboard keys with the name for each key. The number behind the key shows the octave of the key.

2.2. Brute Force Algorithm

Brute force algorithm is a kind of popular algorithm. Brute force is a straightforward approach to solve a problem, usually based on the statement of the problem and the definition of the concepts involved. Brute force algorithm solves the problem with a very simple, direct and clear manner (obvious way). Brute force algorithm is often easier to implement that more advance algorithms, and because of its simplicity, sometimes brute force algorithm can be more efficient (in terms of implementation).

The example of brute force algorithm :

a. Finding factorial of number n.
   N! = 1 x 2 x 3 x …. X n for n > 0
   N! = 1 for n = 0
   Algorithm : Multiply n numbers simultaneously
b. Multiply a matrix n x n with another matrix n x n.
   Let's say C = A x B and the matrix's elements stated as cij, aij, and bij.

$c_{ij} = a_{i1}b_{jk} + a_{i2}b_{2j} + ... + a_{in}b_{nj} = a_{ik}b_{kj}$

Algorithm : compute all elements multiply result one by one, by multiplying 2 factors with length n.

c. Finding all factors of an integer.
   Definition : integer a is factor of integer b if b is divisible by b.
d. Finding an element of a set with the maximum or minimum value.
   Problem : given a set of n integers. The integers are stated as a1, a2, …, an. Find the maximum element in the given set.
e. Sequential Search.
   Problem : given n numbers a1, a2, …, an. Find whether x is exist in the given set. If x is found in the set, return the index, if not, return 0.
f. Prime Testing.
   Problem : given an integer. Check whether the integer is a prime or not.

The characteristics of the brute force algorithm are as follows:

a. Brute force algorithm is not really a smart and efficient algorithms, because the brute force algorithm requires a number of steps that many in the settlement and of course require time proportional to the number of resolution steps.
b. Brute force algorithm is often the least preferred option because it is less efficient, but if you are looking for a basic pattern, order, or special trick, usually can help to find a more intelligent and more efficient algorithms.
c. For small problems, the simplicity of brute force is more calculated than the inefficiency. Brute force algorithm is often used as the basis to compare several efficient alternative algorithms.
d. Although brute force is not a problem solving technique that is efficient, but brute force techniques can be applied to the most of the problems.

The brute force algorithm solves the problems by following these steps:

1. Enumeration (list) any possible solutions in a systematic way.
2. Evaluate each of the possible solutions one by one and keep the best solution found so far (the best solution so far).
3. When the solution search end, announce the best solution.

The advantages of brute force algorithm are:

1. Brute force algorithm can solve almost all the problems (wide applicability).
2. Brute force algorithm is simple and easy to understand.
3. Brute force algorithm generates a viable algorithm for several important problems such as searching, sorting, string matching, matrix multiplication, etc.
4. Brute force algorithm generates a standard algorithm for computing tasks such as addition or multiplication n numbers, determine the minimum or maximum element in the table (list).

The disadvantages of the brute force algorithm are:

1. Brute force algorithm rarely produces an efficient algorithm.
2. Some brute force algorithm is slow so it can't be accepted.
3. Brute force algorithm is not as constructive or creative as other problem solver techniques.

III. IMPLEMENTATION

The algorithm will works on many steps. The first step, we should do some preparations on the string input, and some initiations. The second, we start the checking process. And the last one, we select the music key by using the checking results.

A song consists of many chords. Chords in a song can be implemented as a string. Each chord is separated by a space character in that string.

Before we work with the input string, we must make some preparations, such as preparing the chord base to compare with the song's chords, and preparing the next chord for each chord.

The chord base that we use in this experiment is based on many standard songs. Let's say that a song played in the C key. The standard chords for the C key are C, Dm, Em, F, G, and Am.

For the second preparation, we prepare the next chord for each chord. As explained before, there are 12 tones based on keyboard and many music instruments. They are A, A#, B, C, C#, D, D#, E, F, F#, G, G#. So the next chord of A is A#, the next chord of B is C, and so on. And for each of the chord, we should make a counter and initiate them to zero.

Now let's start working with the input. At first, we parse the input string into array / vector of strings. As explained in the theories, a chord consists of the following parts :

1. The root note (e.g. C).
2. The chord quality (e.g. major, maj, or M).
3. The number of an interval (e.g. seventh, or 7), or less often its full name or symbol (e.g. major seventh, maj7, or M7).
4. The altered fifth (e.g. sharp five, or #5).
5. An additional interval number (e.g. add 13 or add13), in added tone chords.

In this algorithm, we can ignore part number 3, 4, and 5 so we just have to consider the first 2 parts of chords. For example, if the chord is C#7, we just consider this as C or CM, if the chord is G#m9, we just consider this as G#m, and so on.

After we got the chords of the song with just the first 2 parts, we can start the music key checking. The checking will works like this. For every chord in the song we will check whether the chord exists in the standard chords that we have defined before. If the chord exists in the standard chords, we will increase the counter. We will do this for another 11 possible music keys.

For the first checking, we will use the standard chords in C key, they are C, Dm, Em, F, G, and Am. For the second checking we will use the stand chords in the next key, C#. We will do this by using the next chord for all the standard chords, they are C#, D#m, Fm, F#, G#, A#m. And after that we will check again with these new standard chords, and so on.

The algorithm of the checking will be explained in the pseudo code below :

```
procedure check (input base : array[0..m-1] of string, input chords : array[0..n-1] of string, input/output cnt : array[0..11] of integer)
{base : standard chords that will be used as chord bases; chords : all chords in the song; cnt : counter for each possible music keys}
DECLARATION
i,j,k : integer
found : Boolean
ALGORITHM
 i traversal 0..n-1
  j traversal 0..m-1
   found <- false
   k traversal 0..11
    if (chords[j] = base[k]) then
     found <- true
     break
    {end if}
   if (found)
    cnt[i] <- cnt[i] + 1
   {end for k}
  {end for j}
  j traversal 0..11
   base[j] <- next[base[j]]
  {end for j}
 {end for i}
```

After we've got the counter for every music keys, we can start the third phase, selecting the right music key for the song. How will we select the right music key? The first step, we will find the key with the maximum counter. If there is just a key with the maximum counter, we can say that we have found the answer.

If there are more than one key with the maximum counter, we will do the second step. In this step we will check the last chord of the song. In the standard songs which is played in C key, the last chord will be C or Am, C for the songs with 'happy' or 'neutral' feeling, and Am for the songs with 'sad' or 'gloomy' feeling. Let's say the 2 keys that have maximum counter are C and G. If the last chord is C or Am, so the music key for the song is C, and if the last chord is G or Em, so the music key is G.

The algorithm for the selection will be explained in the pseudo code below :

```
function select (input chords : array[0..n-1] of string, input cnt : array[0..11] of integer) -> integer
{chords : all chords in the song; cnt : counter for each possible music keys}
DECLARATION
key1,key2 : string
maxi,imaxi : integer
found : Boolean
ALGORITHM
 maxi <- 0
 cntmaxi <- 0
 i traversal 0..11
  if (maxi < cnt[i])
   maxi <- cnt[i]
   imaxi <- I
  {end if}
 {end for i}
 i traversal 0..11
  if (maxi = cnt[i])
   cntmaxi <- cntmaxi + 1
  {end if}
 {end for i}
 if (cntmaxi = 1)
  ➔ imaxi
 {end if}
 else
  i traversal 0..11
   if (maxi = cnt[i]) and ((chords[n-1] = key1) or
   (chords[n-1] = key2)) then
    imaxi <- i
   {end if}
   else
    key1 <- next[key1]
    key2 <- next[key2]
   {end else}
  {end for i}
  ➔ imaxi
 {end else}
```

The brute force algorithm above will be effective to find the music key for a song with single key. What if the song uses many keys? We can't use the algorithm as explained above. But we can improve the algorithm above.

First we can split the song by the part of the song, such as coda, reff, verse, bridge, etc. Why do we have to do this? It's because the change of the key of a song usually happens when we change to the other part of the song. After we have split the song to some parts, we can do the algorithm above for every parts of the song. The algorithm above will works in this more complex song as effective as before.

IV. CONCLUSION

Brute force algorithm is really suitable to find the music key of a song because of the small constraints of the problem. By using brute force, we can have all possibilities of the answers, and because of the small constraint, it will be very effective. There is so much possibilities of the development of this algorithm in the

future and it can help people who wants to learn about music but doesn't really have the natural gift in the music.

V. ACKNOWLEDGMENT

I would like to express the deepest appreciation to my teacher Mr. Rinaldi and Mrs. Masayu, for their guidance in Algorithm Strategy course in ITB. Without their teaching and their encouragement, this paperwork wouldn't be completed.

I would also like to extend my thanks to all of my friends who had helped me in the completion of this paperwork.

Finally, I wish to thank my parents for their support and encouragement throughout my study.

REFERENCES

[1] Ilmu Pengertian. Algoritma Brute Force. December 20, 2013 (1.00 AM) < http://ilmupengertian.blogspot.com/2013/02/algoritma-brute-force.html>

[2] Zebra Keys. 12 Keys of Music. December 19, 2013 (19.00 PM) < http://www.zebrakeys.com/lessons/beginner/musictheory/?id=12>

[3] Loovo. Dasar Pembentukan Tangga Nada & Chord. December 19, 2013 (19.15 PM) < http://loovomusic.blogspot.com/2013/05/dasar-pembentukan-tangga-nada-chord.html>

[4] Apel, Willi. 1969. *Harvard Dictionary of Music.* Cambridge: Harvard University Press.

[5] White, John D. 1976. *The Analysis of Music*.

[6] Stephenson, Ken. 2002. *What to Listen for in Rock: A Stylistic Analysis* . New Haven: Yale University Press.

[7] Kennan, Kent Wheeler. 1970. *The Technique of Orchestration, second edition*. Englewood Cliffs, New Jersey: Prentice-Hall, Inc.

[8] Boretz, Benjamin. 1995. *Meta-Variations: Studies in the Foundations of Musical Thought*. Red Hook, New York: Open Space.

[9] Munir,Rinaldi. 2009. Diktat Kuliah IF3051 Strategi Algoritma. Program Studi Teknik Informatika STEI ITB

[10] Benward & Saker. 2003. *Music: In Theory and Practice, Vol. I*, Seventh Edition.

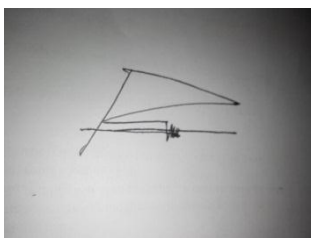
[11] Károlyi, Otto. 1965. Introducing Music. Penguin Books.

[12] Benjamin, Horvit, and Nelson. 2008. *Techniques and Materials of Music*

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 20 Desember 2013



Ridho Akbarisanto / 13511005