International Journal of Scientific Research in Science, Engineering and Technology (www.ijsrset.com)
© 2020 IJSRSET | Volume 7 | Issue 2 | Print ISSN: 2395-1990 | Online ISSN : 2394-4099
DOI : https://doi.org/10.32628/IJSRSET1207246

English Myanmar Dictionary System by Using Linear and Binary Search Techniques

Myo Ma Ma[*1], **Yin Myo KKhine Thaw**[2], **Lai Lai Yee**[3]
[*1]Faculty of Computer Science, University of Computer Studies, Mandalay, Myanmar
[2] Faculty of Information technology supporting and maintenance, University of Computer Studies, Mandalay, Myanmar
[3] Faculty of Information Science, University of Computer Studies, Mandalay, Myanmar

ABSTRACT

This paper is aimed to develop a searching method based on binary search and linear search as well as to understand the finding of search methods. The system searches the desired word for English to English and English to Myanmar. The system may help the English may help the English Language user enable to know the desired word of English and Myanmar meaning. The system output is searching word of English meaning, Myanmar meaning, part of speech, searching time and step. And also, the system finds cross reference and user's unknown word by using binary search and linear search of searching algorithm. This system is implemented by using ASP.NET platform.

Keywords : English Myanmar Dictionary, Linear Search, Binary Search, Searching Alogorithm

I. INTRODUCTION

In this paper, English Myanmar dictionary system is searched by using binary and linear search techniques. This system supports to learn and use the English Myanmar dictionary. The system can also provide the user with the ability of the searching keywords from input sentences and words. User can apply the system as an digital electronic dictionary. Searching and sorting are two of the most fundamental and widely encountered problems in computer science. Two algorithms for search are described. Searching algorithms are closely related to the concept of dictionary. Dictionaries are data structures that support search, insert, and delete operations. The binary search algorithm is used to find data stored in an array. This method is very effective, as each iteration reduces the number of items to search by one-half. Binary search is much faster than linear search for most data sets. The linear search is much faster than linear search for most data sets. The linear search is applied to look at every item in the data sets. The linear search is applied to look at every item in the data set before you find the one you are looking for. With binary search, eliminate half of the data with each decision.

Computers are also the newest way to use dictionaries. Dictionaries are issued in electronic form, taking advantage of this technology to increase the speed of lookup and cross-reference, to extend methods of searching for information, and to include recordings of pronunciations, so that user can hear words or phrases spoken aloud. This form of publication is likely to bemcome more and more usual. This system is to reduce voluminous and time-comsuming works by using Database System and searching algorithm, to avoid human errors.

The organization of this document is as follows. In Section II describes the searching approach. Section III, we devote comparing Binary and Linear Search Algorithms and experimental result. Section IV, we conclude the paper.

II. METHODS AND MATERIAL

A. *Searching Approach*

Searching is fundamental operation to which computers are applied every day. Searches are broken into uniform searches and informed searches. Main criteria for evaluating search strategies are completeness, time complexity, space complexity optimality. The outcome of experiments conducted to measure and compare the time complexity of two algorithms for finding elements in a sorted array of alphabetic. The algorithm as a C# program and measured both the number of basic operations performed by the program and its execution time. The experimental results were consistent with the theoretical predictions for this algorithm. The algorithms of interest each search a sorted array using a well-known linear searching method and binary searching method.

1) *Linear Search*: Linear search is a search algorithm, also known as sequential search that is suitable for searching a list of data for a particular value. It operates by checking every element of a list at a time in sequence until a match is found. Linear search runs in O(n).

In a linear search of an array in which the elements are sorted, examine each element in sequential order until find the element that corresponds to a key. If there are N elements in the array, the aveage users would find the key after N/2 steps. The average performance of linear search can be improved by using it on an ordered list. In the case of no matching element, the search can terminate at the first element which is greater (less) than the unmatched target element, rather than examining the entire list. An ordered list is a more efficient data structure in general for searching than an unordered list.

2) *Algorithm for Linear Search*: This algorthm was not hard to develop, and it will work very nicely for modest-sized lists. The Python in an index operations are both implement linear searching algorithms.

Given; String array, String
Return: index of String in array, if string is in array, or -1 if string is not in array.

```
public static int LinSearch (String [] a, String s)
    {
      for (int i=0; i<a.lengrh; i++)
          if(s.equals (a[i])) return i;
    return -1; // return -1 if elt is not in a
    }
```

Users are searching through the list of items one by one until the target value is found. This algorithm translates direcly into simple code. This strategy is called a linear search.

3) *Binary Search*: Basic technique is to compare the search element with the element which is in the middle of the search space and then to restrict further searching into the appropriate half of the search space (this can be done because the search space is sorted).

Then, at each step, the process is repeated (cutting the remaining search space in half at each step) until either the search element is found or we've run out of elements to compare and the element was not in the search space. The items in an array sort them in either ascending or descending order on the key first, then we can obtain much better performance with an algorithm called binary search.

First, computer the key with the item in the middle position of the array. If there's a match, we can return immediately. If the key is less than the middle key, then the item sought must lie in the upper half of the array, So, repeat the procedure on the lower (or upper) half of the array.

4) *Algorithm for Binary Search*:

Given: String array, String
Return: index of String in array, if string is in array, or -1 if string is not in array.

```
public static int binsSearch (String [] db, s)
{
 int lower =0 //First possible position of s
 int upper = db.length;//First impossible position
of s
 int middle; // Middle position between lower,
       upper
 while (lower < upper)
  {
     middle = (lower + upper) / 2;
     if (s.compareTo (db[middle]==0)
    return middle;
     else if (s.compareTo (db[middle]<0)
    upper = middle;
     else
    lower = middle + 1;
  }
      return -1;
  }; // Element not found.
```

An application of the "divede and conquer" strategy, requires the array (search space) to be sorted.

- *Loop invarient*: The search value (sometimes called the target value), provided it is present in the search space, will be found between the indices low and high inclusively.
- *Termination condition*: Either the target value is found or at most one item is left in the array to be search. Initially, low index = 0, and high = size - 1. The loop should terminate whenever high <= low, provided it has not terminated earlier on as success.

B. *Complexity of Algorithms*

An algorithm provides a satisfactory solution to a problem; it must always produce the correct answer and it should be efficient. One measure of efficiency is the time used by a computer to solve a problem using the algorithm, when input values are of specified size. The second measure is the amount of computer memory needed to implement the algorithm, when input values are of specified size.

An analysis of the time needed to solve a problem of a particular size involves the time complexity of the algorithm. An analysis of the computer memory required involves the space complexity of the algorithm. These two factors form the computational complexity of the algorithm. Time complexity tells us if a problem can ever be finished or when it can be finished. Space complexity depends on the particular data structures used to implement the algorithm.

1) *Time Complexity*: The time complexity of an algorithm can be expressed in terms of the number of basic operations used by the algorithm when the input has a particular size. The basic operations can be addition, multiplication, division or comparison of integers.

Since different computers need different amount of time for basic operations, time complexity is described in term of the number of operations required instead of actual computer time. Additionally, computers are different in their performance speed for the same opertions.

The part of the algorithm that has the greatest influence on both algorithms' execution time is the number of iterations the algorithm performs. Thus, the basic operation is defined as how much time the loop control condition is calculated.

2) *Time complexity Analysis of Linear Search*: If there are n items in our collection - whether it is stored as an array or as a linked list - then it is obvious that the worst case, when there is no item in the collection with the desired key, then n comparisons of the key with keys of the items in the collection will have to be made. To simplify analysis and comparison of algorithms, look for a dominant operation and count the number of times that dominant opeartion has to be performed.

The array is to be sorted or not.
"Brute-force" searching technique.

TABLE 2.1 LINEAR SEARCH

Sequential Search		
Best case	need to look at only 1 element	O(1)
Avearage case	need to look at ½ the elements	O(N)
Worst case	need to look at all n elements	O(N)

- *Avearge case*: The target value is somewhere in the array. In fact, since the target value can be anywhere in the array, any element of the array is equally likely. So on avearge, the target value will be in the middle of the array. The dominant opertion is the comparison, since the search requires n comparisons in the worst case, this is a O(n) (pronounce this "big-Oh-n" or "Oh-n" algorithm.
- Best case: The target value is in the first element of the array and requires a single comparison and computer scientists denote this O(1). The average time depends on the probability that the key will be found in the collection. Thus, in this case, estimation of the average time is of little utility. So, the search takes some tiny and constant amount of time. In real life, don't care about the best case because it so rarely actually happens.
- Worst case: The target value is in the last element of the array. So, the search takes an amount of time proportional to the length of the array. Computer scientists denote this O(n). If the performance of the system is vital, i.e., its part of a life-critical system, then must use the worst case in our design calculations as it represents the best guaranteed performance.

Searching in Linear Search Array
With the data set:
Assembly, Accessory, Ram, Vdu, Jargon, Telephony, Zone

Assembly	Accessory	Ram	Vdu	Jargon	Telephony	Zone

Searching for "Jargon"

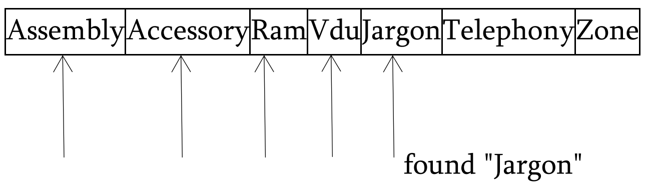


Fig 2.1 Searching by Linear Search

"Jargon" is found at 5th time. Linear search means looking at each element of the array, in turn until find the target. Binary search takes an amount of time proportional to half the length of the array also proportional to the length of the array O(n). So, finding the fastest way to search (or any task) is good, because it save time.

3) *Time Complxity Analysis of Binary Search*

TABLE 2.2 BINARY SEARCH

Binary Search		
Best case	need to look at only 1 element	O(1)
Avearage case	need to look at $\log_2(n)$ elements	O($\log_2 n$)
Worst case	need to look at $\log_2(n)+1$ elements	O($\log_2 n$)

As data size n grow, doubling the data yields rutimes that increment by a constant amount. This is an O(n) algorithm. The time complexity will never exceed O(n).

The array is split in half each time; the number of steps is always going to be equal to the base-2 logarithm of n, which is considerably less than O(n). So, an even better choice would be to set the upper bound to log N, which is the upper limit that we known we are guaranteed never to cross. Therefore, a more accurate cliam is that binary search is a logarithmic, or O($\log_2$ n), algorithm.

We can divide a set of n times in half at most $\log_2$ n times. Thus, the running time of a binary search is proportinal to log n this is a O(log n) algorithm. Binary search requires a more complex program than our original search, and thus for small n it may run slower than the simple linear search. However, for large n,

$$\lim_{n\to\infty}\left(\frac{logn}{n}\right) = 0$$

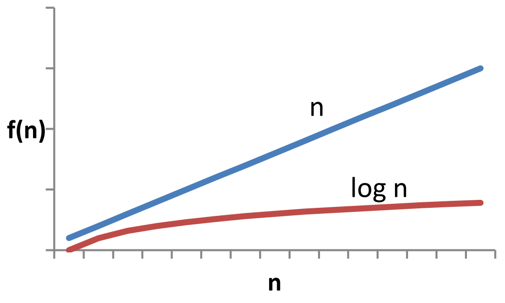


Fig2.2 Comparison of Binary Search and Linear Search Run Time

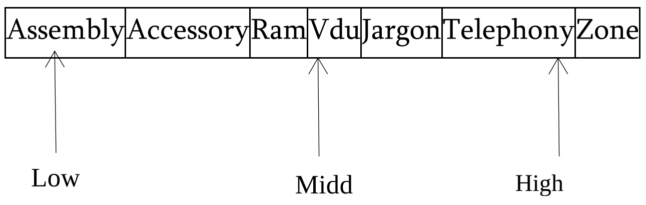
Thus, at large n, log n is much smaller than n, consequently an O(log n) algorithm is much faster than an O(n) one shown in Fig2.2. In the worst case, insertion may require n operations to insert into a sorted list.

- In the list, find the new item belongs, using binary search in O(log n) operations.
- In the worst case, the new item is the first in the list, requiring n move operations. A similar analysis will show that deletion is also an O(n) operation.

Searching in Binary Search Array

With the data set:

Assembly, Accessory, Ram, Vdu, Jargon, Telephony, Zone



Searching for " Accessory "

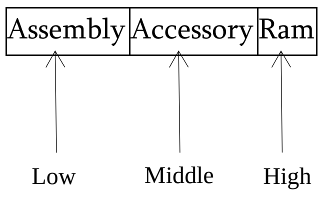


Figure 2.2 Searching by Binary Search

Searching for "Accessory" is found at middle.

- The first iteration for the loop, search region is the whole array.
- Second, it's half the array; target is less than the middle. So, goes to the left. Target is found at middle.

For Kth iteration of the binary search, the search region is ($1/2^{k-1}$) of the array. The time complxity of binary search is O($\log_2$ n). O($\log_2$ n) is exactly the same as the O(log n).

Know from math class that $\log_a$ x =$\log_b$ x/logb at the relationship between logs to various bases is simply a constant(1/log$_b$ a). Therefore, O(log n) is the same as O(log$_b$ n) for any base b- don't care about the base.

C. *Languages in Dictionary*

Dictionary, listing of the words of a language, usually in a alphabetical order but sometimes also by topic, with their meanings or their equivalents. A dictionary may also contain pronunciations, syllavications etymologies (world histories) and examples of usage.

The term dictionary is also applied to any systematic list of special terms such as abbreviations, slang, or etymology, or to a list in which the special terms of a particular subject are defined. Some dictionaries focus on particular subjects such as science, biology, geography, mathmetics, history, or philosopy. Some dictionaries are called encyclopedic, because they not only define but also offer additional descriptive and explanatory information and identify many biographical and geographical names.

Computers have been important tools in lexicography for decades; the need to store, sort, and retrieve huge amounts of linguistic information drew publishers to electronic methods. A lexicographer can see not only individual occurrences of a word, but also occurrences sorted by nearby words, by grammatical form, or by context. Computational linguists use such data to analyze language patterns along with word meanings, with the overall goal of enabling computers to understand and generate language as skillfully as human users do.

Myanmar language is one of the south-east languages written in Myanmar script. Myanmar script has been a majority language of Myanmar for over 1000 year old. Myanmar writing system derives from the Brahmin-related scripts which flourished in India from about 500 B.C to over 300 A.D. Myanmar. Myanmar script is a syllable writing system was constructed from consonants, consonant combination symbols, vowel symbols related to relevant consonants and diacritic marks indicating tone level. Overall writing direction is from left to right. Traditionally, Myanmar language alphabet is recognized as containing 33 consonants, 12 vowels and some conjunction alphabets.

III. COMPARING BINARY AND LINEAR SEARCH ALGORITHMS

So far, user have developed two solutions to users' simple searching problem. Which one is better? The linear search algorithm is much eaiser to understand and implement. Users expect that the binary search is more efficient because it doesn't have to look at every value in the list. Intuitively, then, usesr might expect the linear search to be a better choice for small lists and binary search a better choice for larger lists.

Linear search was faster for lists of length 10 or less, and there was not much difference in the range of length 10-1000. After that, binary search was a clear winner. The empirical analysis has confimed our intuition, but these are results from one particular machine under specific circumstances. Another approach is to analyze the algorithms abstractly to see how efficient they are. Other factors being equal, users expect the algorithm with the fewest number of steps to be the more efficient. But, how do users count the number of step? For example the number of times that either algorithm goes through its main loop will depend on the particular inputs.

Users have already guessed that the advantage of binary search increases as the size of the list increases. Computer scientists attack these problems by analyzing the number of steps that an algorithm will take ralative to the size or difficulty of the specific problem instance being solved. For searching, the difficulty is determined by the size of the collection. Obviously, it takes more steps to end a number in a collection of a million than it does in a collection of ten.

TABLE 2.3 LINEAR AND BINARY RUN TIME

List Size	Having	
	Linear	Binary
1	1	0
2	2	1
4	4	2
8	8	3
16	16	4

How many steps are needed to end a value in a list of sizes n? Users are particularly interested in what happens as n gets very large. Let's consider the linear search first. If users have a list of ten items, the most work users' algorithm might have to do is to look at each item in turn. The loop will iterate at most ten times. Suppose the list is twice as big. Then, users might have to look at twice as many items. If the list is three times as large, it will take three times as long, etc. In general, the amount of time requried is linearly related to the size of the list n. This is what copmputer scientists call a linear time algorithm. What about the binary search?

Suppose the list contains sixteen items shown in Table 2.3. Each time through the loop, the remaining range is cut in half. After one pass, there are eight items left to consider. The next time through there will be four, then two, and finally one.

How many times will the loop execute? It depends on how many times we can have the range before running out of data. Each extra iteration of the loop doubles the size of the list. If the binary search loops i times, it can end a single value in a list of size 2i. Each time through the loop, it looks at one value (the middle) in the list. To see how many items are examined in a list of size n, need to solve this relationship: n=2i for i. In this formula, i is just and exponent with a base of 2. Using the appropriate logarithm, give us this relationship: $i = \log_2 n$. Just remember that this value is the number of times that a collection of size can be cut in half.

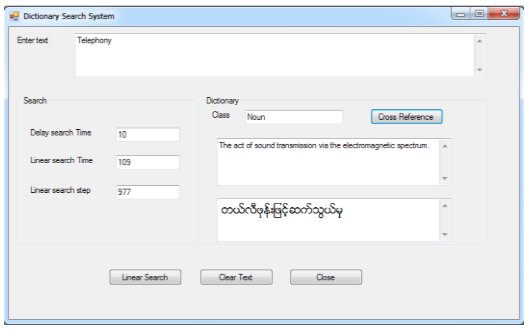
Therefore, what does this bit of math tell us? Binary search is an example of log time algorithm. The amount of time it takes to solve a given problem grows as the log of the problem size. In the case of binary search, each additional iteration doubles the size of the problem that can solve.

A. *Preliminary Testing*

After finishing the design phase, the system is implemented by ASP.net [5] and prliminary observation is carried out. This system is the dictionary system to retrieve and view unknown word for user. In dictionary, user can search the information that he or she wants to know. Run time(step) of seaching data is calculated by linear and binary runtime algorithm.

The system also displays the searching time in millisecond. When user enters the keyword, the system displays the meaning of the word in English and Myanmar as well as part of speech.

The system can search words by using linear search algorithm as shown in figure 3.1.



Enter text: Telephony

Delay search Time: 10

Linear search Time: 109

Linear search step: 977

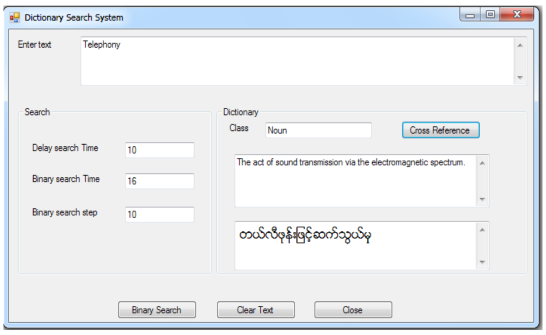
Class: Noun

The act of sound transmission via the electromagnetic spectrum.

တယ်လီဖုန်းဖြင့်ဆက်သွယ်မှု

Fig 3.1 Linear Search by by Inputing Keyword

The system can search words by using binary search algorithm as shown in Figure 3.2.


Dictionary Search System

Enter text: Telephony

Search
Delay search Time: 10
Binary search Time: 16
Binary search step: 10

Dictionary
Class: Noun
The act of sound transmission via the electromagnetic spectrum.
တယ်လီဖုန်းဖြင့်ဆက်သွယ်မှု

Fig 3.2 Binary Search by by Inputing Keyword

Figure 3.3 shows the graph of the comparison of seaching time (in millisecond).

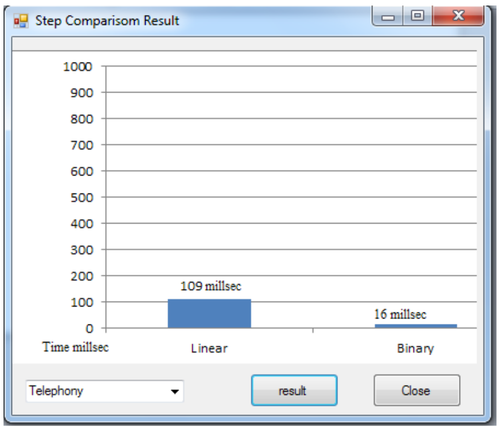


Figure 3.3 Time comparison Graph

Figure 3.4 shows the run time of the two algorithms and demonstrates the logarithmic growth in run time.

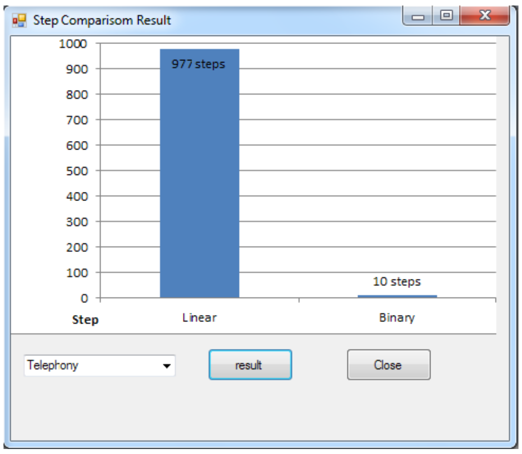


Figure 3.4 Time comparison Graph

B. *Experimental Result*

As shown in Table 3.1, user can also see the experimental results of the binary and linear search of the system. Binary search run time is faster than linear search of the system. Binary search runtime is faster than linear search tun time. The linear search algorithm is much eaiser to understand and implement. On the other hand, we expect that the binary search is more efficient, because it doesn't have to look at every value in the list. Therefore, researchers and users use the linear search to be a better choice for small list and binary search a better choice for larger list.

Table 3.1Experimental Results

Tes t	Keyword	Searching time (millisecond)		Run time (step)	
		Linea r	Binary	Linea r	Binary
1	Assembly	15	16	113	10
2	Accessory	15	16	26	7
3	Ram	187	16	840	8
4	Vdu	375	16	840	10
5	Jargon	78	16	577	7
6	Telephon y	109	16	977	10
7	Zone	203	15	1096	8

IV. CONCLUSION

Searching data in efficient way is important in most of computer systems. This software is implemented by using ASP.Net platform. The user can view and study over the entire search method using linear search and binary search.

A comparison may be made for two types of searches such as Binary Search and Linear Search used in the system. Binary Search can produce faster than linear search in average case and worst case. The linear search algorithm is much easier to understand and implement. On the other hand the binary search is more efficient because it doesn't have to look at every value in the list.

The linear search is considered to be a better choice for small list and binary search a better choice for larger list. The time comparison and step comparison of two search algorithms can enhance the system appearance. The system provides searching for the meaning of the word, part of speech, cross reference and pronuciation of word by using linaer search and binary search.

V. REFERENCES

[1]. Harriet Fell, Javed Aslam,, "Algorithms for Search," in Title of Discrete Structure, CSU200, Fall 2007

[2]. "http://www.cs.grinnell.edu/~walker/fluency_book/labs/run-time-searching.html"

[3]. "http://www.en.wikipedia.org/wiki/Search_alogrithm

[4]. "http://www.studytonight.com/data-structures/search-algoritms

[5]. Tom Dykstra. ASP.NET Web Deployment using Visual Studio (April,2013), Microsoft Corporation

Cite this article as :

Myo Ma Ma, Yin Myo KKhine Thaw, Lai Lai Yee, "English Myanmar Dictionary System by Using Linear and Binary Search Techniques", International Journal of Scientific Research in Science, Engineering and Technology (IJSRSET), Online ISSN : 2394-4099, Print ISSN : 2395-1990, Volume 7 Issue 2, pp. 209-217, March-April 2020. Available at doi : https://doi.org/10.32628/IJSRSET1207246
Journal URL : http://ijsrset.com/IJSRSET1207246