Genetic Distance Calculation based on Locality Sensitive Hashing

T. Pathirana[#1], S. Bandara[#2], G. Gamage[#3], N. Gimhana[#4], A. Wickramarachchi[*5], I. Perera[#6]

[#]*Department of Computer Science and Engineering,*
University of Moratuwa
Moratuwa, Sri Lanka
{[1]thilina.15,[2]shanaka.15,[3]gihangamage.15,[4]gimhanadesilva.15,
[6]indika}@cse.mrt.ac.lk

**The Australian National University*
[5]anuradha.wickramarachchi@anu.edu.au

***Abstract*— Measuring the genetic relatedness between different species is one of the major challenges in the field of phylogenetics. Genetic distance calculation based on DNA data is highly using a mechanism to determine inter-species relationships. Genetic distance computation can be further bifurcated as alignment-free sequencing and alignment-based sequencing. With this research, we are presenting an alignment-free genetic distance calculation technique which is based on locality sensitive hashing (LSH). By this approach, we are hashing large DNA sequences into numeric arrays and make the comparison more efficient and simplified.**

***Keywords*—phylogenetics; genetic relatedness; genetic distance; locality sensitive hashing; min-hashing**

I. INTRODUCTION

Genetic distance is the degree of genetic difference (genomic difference) between species or populations which is measured by some numerical method [1]. The average number of nucleotide differences per gene is a measure of Genetic Distance. Many other molecular data can be used for measuring genetic distances. Genetic distance is an important factor in the process of reconstructing the history of the population. As a result of many kinds of research on genetic distance suggests that Saharan, African and Eurasian people diverged about 100,000 years ago[2]. Another key aspect of genetic distance is that it is used for understanding the origin of biodiversity. The different breeds of domestic animals are investigated for the purpose of determining which breeds should be protected to maintain diversity [3].

The phylogenetic tree [4] is a branching diagram or "tree" showing evolutionary relationships between different species based upon their physical and genetic characteristics. Genetic distance calculation can be considered as the main tenacity in Phylogenetic tree construction.

Genome Sequence alignment is a methodology that is currently using to identify regions of similarities and dissimilarities between sequences to calculate the genetic distances. Due to the limitations of Alignment based method Alignment Free methodology was introduced. Our approach of using LSH (locality-sensitive Hashing) tends to be used in applications related to phylogenetics without using alignment.

Calculating genetic distances between Species using LSH (Locality Sensitive Hashing) is our approach. Since DNA sequences are larger, existing LSH implementations cannot expect the maximum accuracy. Here we are introducing a new approach of creating minHash objects by partitioning sequences for comparison. Our approach tends to give convenient results that are fast and accurate.

As a primary step of Phylogenetic tree construction, the methodology we are proposing here can obtain the distance matrix by calculating inter-species distances which makes a tremendous step in the field of bioinformatics.
As the next phase the distance matrix we calculated can be used to predict ancestors and cluster species into groups based on their common characteristics.

II. BACKGROUND

Genetic distance is the way of representing similarity or dissimilarity between DNA sequences. Genome sequence alignment [5] is a method that is used to identify similar regions in sequences [6]. There are three types of genome sequence alignment types. They are global, local and glocal. Global alignment is used when the sequences have more similarities and local alignment is used when sequences have fewer similarities. In global alignment, it aligns every remnant in every sequence. Glocal is a combined method that attempts to find the best possible alignment of sequence.

BLAST [7] (Basic Local Alignment Search Tool) is one of the most popular alignment-based tools in bioinformatics because of the functionalities and simplicity provided by the tool. It was developed by a group of scientists with the association under NIH (National Institutes of Health). In alignment-free methods such as BLAST does not consider the mismatches between sequences [8]. Because of that, its performance gets decreased when the identity of the sequence gets increases. Another major drawback of alignment-based methods is that they show less accuracy when one sequence is a repetition of another sequence. As an example, the human DNA sequence is a repetition of the mouse DNA sequence. But when use alignment-based methods such as BLAST to evaluate the similarity between those species it returns lower similarity value which is incorrect. Before applying an alignment-based algorithm it has to complete a considerable amount of preprocessing steps to make sequences applicable for the tool and it is time-consuming [9]. This is an overhead when using alignment-based approaches. Another challenge in alignment-based tools is that multiple sequences cannot be processed in real-time. However, most of the modern approaches of alignment use prefix indexes [10] and compression transformations such as Borrow Wheeler Transform.

Because of the imperfections in alignment-based algorithms, alignment-free approaches were introduced. This is a method that quantifying sequence similarity without involving alignment at any level of the algorithm [11]. Since alignment-free methods have a linear complexity which depends on the length of the sequence, they are computationally inexpensive. In addition to that, they do not rely on assumptions regarding the evolutionary trajectories of sequence changes as alignment-based methods. However, alignment-free methods are focusing on the occurrence of sub-sequences, most of them are memory intensive. Compared to alignment-based methods, these methods are still at the development level to applying in phylogenetic applications. There have more than 100 techniques [12] in the alignment-free category with having the same characteristics we discussed above.

In current state word frequency approach is widely used to calculate genetic distances and phylogenetic applications other than alignment-based approaches [13] because of their less computational power requirement than alignment-based approach [14]. Use feature frequency profile (FFP) of whole genomes for comparison is one approach in the alignment-free method. Another alignment-free approach is Comparison Vector (CV) which calculates the normalized frequency of each possible kmer of the sequence [15]. In both FFP and CV it based on the frequency of word presence in genome sequence and calculate similarity using cosine distance function [16]. But these methods do not consider the positions of subsequence [17].

Return Time Distribution (RTD) is also an alignment-free method which is different from the above-mentioned methods. It considers the amount of time taken to reappear a kmer instead of word count [18]. As in summary, these methods are creating a signature to represent a genome sequence. [19]

In alignment-free methods, the size of the signature which is used to represent the genome sequence gets increased with the size of the genome sequence which is memory consumes. This is the major drawback of existing alignment-free methods. As a solution for this, we are proposing an approach based on Locality Sensitive Hashing (LSH).

LSH is a technique to find the near similar character sequence with high probability and efficiently [20]. This technique is most commonly used to find similar documents. LSH is deviating from conventional hashing techniques because its hash collisions are maximized, not minimized. Hashing-based approximate similarity search algorithms can be divided into two major categories. One of them is data-independent methods such as LSH. the other category is data-dependent methods such as Locality Preserving Hashing [21][22]. LSH algorithm can be subdivided into three steps. Those are shingling, min hashing and LSH. In the shingling step, we convert each document into a set of characters that has constant length. Minhashing step is the most prominent step in LSH. It is a technique to estimate the similarity between two sets in a very efficient manner. Originally this was designed to identify the similar web pages to disregard them from search results. In the minhash step, it converts each document into set of hash values which can be used to calculate the similarity between documents. The size of the minhash object does not depend on the size of the sequence.

III. RESEARCH MATERIALS AND DATASETS

A. Genomic Datasets

The whole-genome sequences were downloaded from the NCBI database (ftp.ncbi.nlm.nih.gov/genomes/genbank/). For the comparisons and results, we used genomes of Bacteria [23].

B. Development Tools and Environmental Configurations

Python was the primary programming language used to implement algorithms.

Testing Environment: 8GB RAM, 4 cores of CPU, 1TB Storage, Ubuntu 16.04

IV. METHODOLOGY

To overcome the above-mentioned drawbacks which are in the currently existing methods, we propose an approach to calculate the similarity between DNA sequences using Locality Sensitive Hashing (LSH). Since the length of a DNA sequence is higher than a normal document, currently existing LSH implementations cannot achieve the expected accuracy of DNA sequence comparing. Because in the normal LSH method, it converts a DNA sequence into a min-hash object and represents the DNA sequences using those min-hash objects. But when the DNA sequences are getting larger, the accuracy of this representation gets decreased. In our approach, we are using a DNA partitioning mechanism to overcome this drawback.

Our approach consists of four major steps. As an output of this approach, it gives the pairwise similarity between DNA sequences.

Part A - Minhashing

Minhashing is the major step of our approach which generates a signature for a given sequence by creating a minhash object which consists of minimum hash values. Within a minhash object, it contains two arrays which are called hash value array and permutation array. Following Algorithm 1 shows the process of initiating a minhash object by initiating those arrays. The number of minimum hash values that need to be generated for the sequence is denoted by the number of permutations. The permutation array is used as a parameter for random bijective permutation function which is used to update the minimum hash values. We take a Mersenne prime as the upper margin of those random values. All the hash values in the hash value array are set to the maximum hash value.

Algorithm 1: Initiate Minhash Object

Require: number of permutations (P)

```
FUNCTION initiate_minhash_object (P)
    LET mersene_prime = (1<<61)-1
    LET max_hash = (1<<32)-1
    LET hashvalues = [P number of max_hash]
    LET permutation = [P number of random values < mersene_prime]
END FUNCTION
```

In Algorithm 2 it illustrates the procedure for minhasing a sequence. First, we need to initialize the minhash object as explained above. Then it needs to divide the DNA sequence into the constant length of subsequences which are called “shingles”. After generating the shingles, they need to feed for an updating method to update the minhash object according to the shingles.

Algorithm 2: Create Minhash for a Sequence

Require: DNA sequence of specie (D), Shingle size (SS), Permutations (P)

```
FUNCTION create_minhash (D,SS)
    LET minhashObj = initiate_minhash_object(P);
    LET shingleArray = split(D,SS)
    FOR each shingle in shingleArray
        minhashObj.update_by_new_shingle(shingle, minhashObj)
    END FOR
    RETURN minhashObj
END FUNCTION
```

Following Algorithm 3 elucidate the updating mechanism of a minhash object according to a shingle. The target of this mechanism is to change the minimum hash values of minhash object if the fed shingle creates a hash value which is less than the existing hash values.

Algorithm 3: Update Minhash Object

Require: Shingle (S), Minhash Object (M)

```
FUNCTION update_by_new_shingle (S,M)
    LET hashvalue = hashfunc(S);
    LET a,b = M.permutations[][]
    LET phv = generate_hashvalue_array(a, b, hashvalue)
    M.hashvalues = min(phv, M.hashvalues)
    RETURN M;
END FUNCTION
```

Part B - Partitioning mechanism

In our methodology, we are representing the DNA sequences using their minhash objects. These minhash objects consist of a constant number of minimum hash values of the DNA sequences. If the shingles which were forged those minimum hash values are diffused in the DNA sequence uniformly, the accuracy of the representation increased. Since the size of a DNA sequence is high, the diffusion of the minimum hash values tends not to be uniform.

As a solution for the above drawback, we are introducing a partitioning mechanism to increase the accuracy of minhasing. Before calculating the minhash object for a given DNA sequence, we are dividing the DNA sequence into a predefined number of partitions. Then each partition is min hashing by

considering them as different sequences. After min hashing all the partitions, we are getting a set of minhash objects for a single DNA sequence and each minhash object consists of a constant number of minimum hash values. Because of the partitioning, it is possible to achieve the approximate uniform diffusion of minimum hash values in the DNA sequence.
Figure 1 and figure 2 illustrates how partitioning provides the approximate uniform diffusion for minimum hash values.

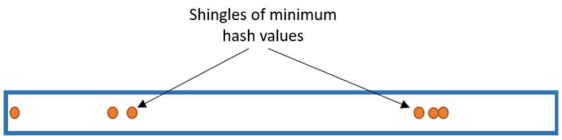

Figure 1 - minimum hash values without partitioning DNA sequence

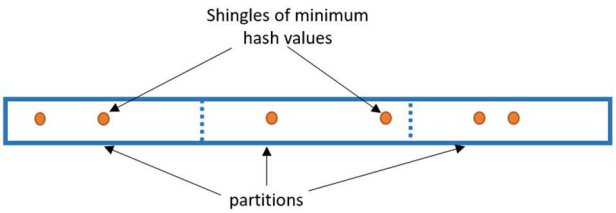

Figure 2 -minimum hash values with partitioning DNA sequence

In figure 1, it shows the minimum hash values are representing a lean region of the DNA sequence. In figure 2, it illustrates that when the partitioning is used, it provides the uniformness of the diffusion of the minimum hash values. And also it guarantees the uniformity of the diffusion to a certain level. The uniformity can be increased by increasing the number of partitions.

Part C - Parallelizing Minhashing

The partitioning approach we used above is a completely independent process for each of the DNA Sequences and as we observed in the earlier stages it took much more time to Hash the Sequences. Our main target was to partition each sequence before min hashing to increase the accuracy of the representation.

Hence, as a solution for the above drawback, our next approach was to do our main task minHash running concurrently and in parallel. Here we used modules that provide interfaces for running tasks using pool of thread. After partitioning is done each thread which is in the thread pool occupies a partition of the sequence and generate a minhash object for that particular partition and store it in a global array. When all the partitions complete by generating minhash objects, it combines all the minhash objects and concatenates them to create a single minhash object for the particular DNA sequence as shown in figure 3. This approach is more efficient than generating minhash objects sequentially.

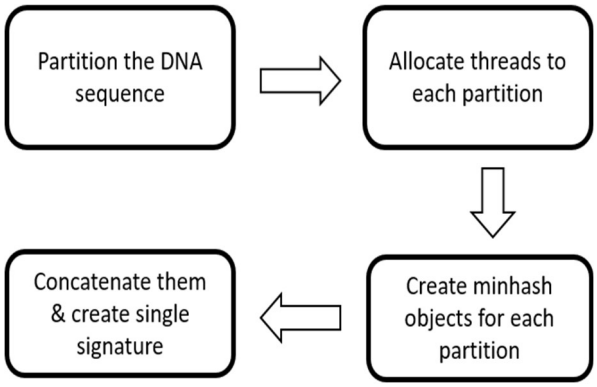

Figure 3 - the process of generating a minhash object

Part D - Comparing

Our Main Target was to obtain the Similarity between independent each species. As our stepwise approaches, we first obtain the minhash Objects which consist of Different Minhash minimum Values. When Comparing with the above two steps Comparing MinHashes and obtain the similarities takes a lesser amount of time.

In this approach, we obtain the Jaccard similarity [24] between the obtained DNA Minhash objects which consists of minimum hash values. The Jaccard Similarity coefficient is a statistic used for determining the similarity and diversity of sample sets. It is the coefficient obtained by the division of intersection of two samples by the union of the Sample sets as given in the following equation.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}.$$

Jaccard coefficient outputs the similarity between the Minhashes obtained for different species in an efficient way.

The LSH based methodology in which we partition each sequence and obtain minhash objects with minimum hash values is a novel approach that we used to compare species. Using a Concurrently min hashing approach gives efficient results than the traditional approach.

V. RESULTS AND EVALUATIONS

In this section, we are mainly focusing on the evaluations done on genome Comparison using LSH. When Using LSH for our approach we had to consider some parameters which

are partition time and Shingle size to obtain results with a maximum accuracy of the distances between the species.
When the partition size increases Comparison time decreases. Because when the partition size increases, no of partitions decreases. Hence the partitions to compare decreases, which decreases comparison Time as shown in Figure 4.

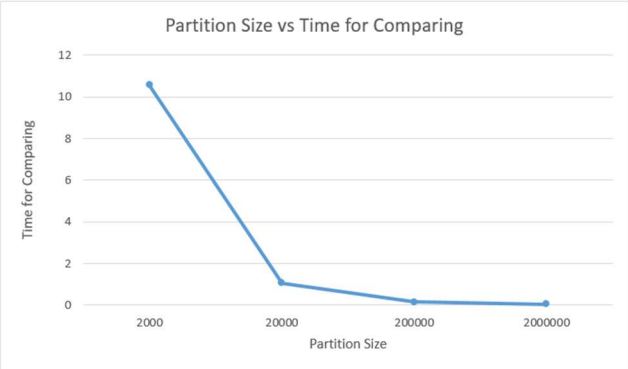

Figure 4 - Partition size vs Time for comparing graph

Our next approach is to obtain minHash objects from the partitions we obtained. But the time taken for min hashing varies with the Partition Size we use as shown in Figure 5. When partitioning size increases, time taken for min hashing increases up to some limit and then decreases.

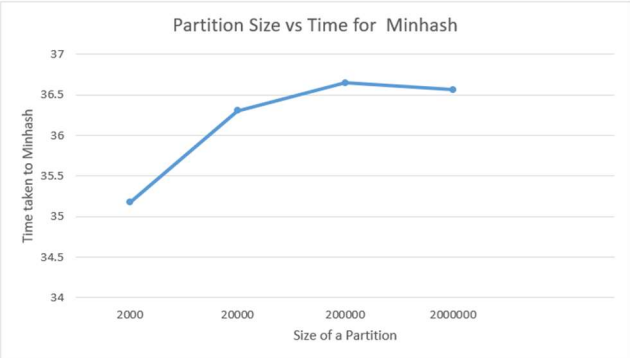

Figure 5 - Partition Size vs Time for Minhashing graph

For our evaluations, we had to select the exact partition size which maximizes the accuracy and efficiency. By considering Figure 4 and Figure 5, we selected 20000 as the partition size for our approach.

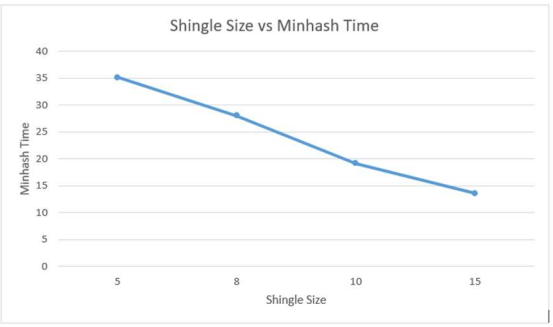

Figure 6 - Shingle size vs Time for Minhashing graph

With the shingle size we use, the minhash time varies as shown in Figure 6. We observe that the MinHash Time decreases with the Shingle Size. But when we increase the shingle size it is difficult to obtain common shingles between species. Therefore, Accuracy decreases. Although, the minHash Time is high we use a shingle size of 5 for our evaluation with the Consideration of the accuracy of comparing Species.

VI. DISCUSSION AND FUTURE WORK

Our focus of this paper is to compare Species using their DNA sequences based on our new approach of using LSH (Locality sensitive hashing) for genome comparison. Rather than using the traditional approach of using LSH we introduced a new method of using LSH by partitioning DNA sequences. The main important decision we had to consider with our approach is the partition size and number of shingles which gives the maximum accuracy of the distances between the species. The final outcome of the approach that we used to generate genetic distances between species is to generate an accurate distance matrix efficiently, in consideration of obtaining the phylogenetic tree. The distance matrix is playing a major role in tree construction. The reliability of the tree, as well as the evolutionary relationships of taxonomies, depends upon the accuracy of the distance matrix we obtained.

As our next step, we suppose to construct phylogenetic trees based on the distance matrices we are building based on this methodology. For phylogenetic tree construction, we are proposing a novel approach that is based on machine learning.

VII. CONCLUSION

Locality sensitive Hashing is a methodology that can be used with whole-genome sequence comparison which can be used for different genomes with diverse lengths. We obtain the the similarity between species by comparing the minhash objects that we created for a given DNA sequence. Most of the DNA sequences are quite large for min hashing. We partition the DNA sequences and make them easier for min hashing. For a sequence, all the partitions were minhashed to obtain a set of minhash objects. Each minhash object consists of a constant number of minimum hash values. Hence memory allocated for a DNA sequence after min hashing is constant. Therefore comparing with larger DNA sequences our approach consumes the same amount of memory. Furthermore, it is computationally efficient. This simplifies the genome representation and makes easier for comparison. With our approach, we obtain the similarity between species by comparing pairwise to obtain Jaccard similarity between them. Furthermore, we are using a concurrent approach for generating minhash objects which is time-consuming. The evaluation shows that our approach

guaranteed to provide the distance matrix that compares the genomes within a short period of time.

REFERENCES

[1] Dogan, Ismet & Dogan, Nurhan. (2016). Genetic Distance Measures: Review. Turkiye Klinikleri Journal of Biostatistics. 8.87-93.10.5336/biostatic.2015-49517.

[2] Nei, M.; A. K. Roychoudhury (1974). "Genic variation within and between the three major races of man, Caucasoids, Negroids, and Mongoloids". *The American Journal of Human Genetics*. 26: 421–443. PMC 1762596. PMID 4841634.

[3] Ruane, J. (1999). A critical review of the value of genetic distance studies in conservation of animal genetic resources. Journal of Animal Breeding and Genetics, 116(5), 317-323. Chicago.

[4] Baum, David. "Reading a Phylogenetic Tree: The Meaning of Monophyletic Groups." *Nature Education* 1, no. 1 (2008): 190. http://www.nature.com/scitable/topicpage/reading-a-phylogenetic-tree-the-meaning-of-41956(Opens in a new window)(Opens in a new window).

[5] Vinga, S; Almeida, J (Mar 1, 2003). "Alignment-free sequence comparison-a review". *Bioinformatics*. 19 (4): 513–23. doi:10.1093/bioinformatics/btg005. PMID 12611807.

[6] Mount DM. (2004). *Bioinformatics: Sequence and Genome Analysis* (2nd ed.). Cold Spring Harbor Laboratory Press: Cold Spring Harbor, NY. ISBN 978-0-87969-608-5.

[7] "BLAST: Basic Local Alignment Search Tool", *Blast.ncbi.nlm.nih.gov*, 2019. [Online]. Available: https://blast.ncbi.nlm.- nih.gov/Blast.cgi. [Accessed: 08- May- 2019].

[8] "BWA," Github, [Online]. Available: https://github.com/lh3/bwa. [Accessed 9 May 2019].

[9] "Website." [Online]. Available: https://genomebiology.biomedcentral-.com/articles/10.1186/s13059-017-1319-7. [Accessed: 08-May-2019].

[10] "Minimap2: pairwise alignment for nucleotide sequences. - NCBI - NIH." https://www.ncbi.nlm.nih.gov/pubmed/29750242. Accessed 20 Mar. 2019.https://github.com/lh3/bwa. Accessed 20 Mar. 2019.

[11] Hide, W; Burke, J; Davison, DB (1994). "Biological evaluation of d2, an algorithm for high-performance sequence comparison". *Journal of Computational Biology*. **1** (3): 199–215. doi:10.1089/cmb.1994.1.199. PMID 8790465.

[12] M. A. R. Cheong Xin Chan, "Next-generation phylogenomics," Biol. Direct, vol. 8, p. 3, 2013.

[13] Chan, CX; Ragan, MA (Jan 22, 2013). "Next-generation phylogenomics". *Biology Direct*. **8**: 3. doi:10.1186/1745-6150-8-3. PMC 3564786. PMID 23339707.

[14] R. Bromberg, N. V. Grishin, and Z. Otwinowski, "Phylogeny Reconstruction with Alignment-Free Method That Corrects for Horizontal Gene Transfer," PLoS Comput. Biol., vol. 12, no. 6, p. e1004985, June. 2016.

[15] S. GE, J. SR, W. GA and K. SH, "Alignment-free genome comparison with feature frequency profiles (FFP) and optimal resolutions.", 2009. [Online]. Available: https://www.nc- bi.nlm.nih.gov/pubmed/19188606. [Accessed: 10- August- 2019].

[16] Apostolico, A; Denas, O (March 2008). "Fast algorithms for computing sequence distances by exhaustive substring composition". Algorithms for Molecular Biology. 3.

[17] Leimeister, CA; Boden, M; Horwege, S; Lindner, S (2014). "Fast alignment-free sequence comparison using spaced-word frequencies". Bioinformatics. 30 (14): 1991–1999. doi:10.1093/bioinformatics/btu177. PMC 4080745. PMID 24700317.

[19] G. Gamage , N. Gimhana, A. Wickramarachchi , V. Mallawaarachchi , I. Perera Alignment-free Whole Genome Comparison Using k-mer Forests International Conference on Advances in ICT for Emerging Regions 2019.

[18] E. al Kolekar P, "Alignment-free distance measure based on return time distribution for sequence analysis: applications to clustering, molecular phylogeny and subtype... - PubMed - NCBI." [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/22820020. [Accessed: 11-August-2019].

[20] Rajaraman, A.; Ullman, J. (2010). "Mining of Massive Datasets, Ch. 3".

[21] Zhao, Kang; Lu, Hongtao; Mei, Jincheng (2014). "Locality Preserving Hashing". pp. 2874–2880.

[22]Tsai, Yi-Hsuan; Yang, Ming-Hsuan (October 2014). "Locality preserving hashing". pp. 2988–2992. doi:10.1109/ICIP.2014.7025604. ISSN 1522-4880.

[23] Ncbi.nlm.nih.gov. (2019). FTP access to GenBank data Bacteria. [online] Available at: ftp.ncbi.nlm.n ih.gov/genomes/genbank/bacteria/ [Accessed 7 Aug. 2019].

[24] Niwattanakul, Suphakit & Singthongchai, Jatsada & Naenudorn, Ekkachai & Wanapu, Supachanun. (2013). Using of Jaccard Coefficient for Keywords Similarity.