Summarize and Evaluate A Research Paper About…

"A computational perception of locating multiple longest common subsequence in DNA sequences."

1) Abstract & Intro:

First, in this research paper, it is talking about bioinformatics, which has become a thorny, branching topic, and raises biologists' controversy. One of these topics is long common subsequences (LCS), which allows us to learn about the arrangement of nucleotides in DNA. The sequence is placed in an array and implement "matching algorithm" to compare between two sequences. Thus, the LCS can be determined, the locations of its occurrence and the number of recurrences as well. They found that maximum length for LCS is eight. They were not satisfied with that, but they considered the calculation of time as one of their concerns (logically, time is increased relatively with progressive length, 100,200… and it took few seconds).

Second, bioinformatics has been defined as not only concerned with molecules, but software engineering, computation and measurements, and the cell being its basic unit on which all experiments and research are conducted. The writer used a picture to clarify the structure of the cell in which the metabolism, etc... takes place, and it contains the genetic information. The DNA sequence is analyzed which contain the four nitrogen bases: cytosine (C), thymine (T), adenine (A) and guanine (G), plus sugar and phosphate. As you know that DNA is responsible for the process of making the protein, and the genes that inherit will enter this process. The basic function of DNA is replication, data encoding, and gene expression. Human genome has approximately 3 billion base and 20,000 gene. Each human cell contains a nuclear genome that is divided into 46 chromosomes, and the analysis of these genes is an emerging field of research, so it is used in forensic problems and the search for regions of abnormalities for the sequence of DNA, so the disease-affected DNA sequence is compared to the standard DNA sequence. A lot of time is spent processing and working on this comparison, given that the number of nucleotides in the genome has exceeded 3 billion, as we mentioned earlier, but the algorithmic methods help reduce the time and space complexity of this problem.

LCS approach define the longest subsequence along two sequences (frequently used). This approach is machine learning, software engineering topic for bioinformatics application. Detect the emergence of disease-causing change sequences in human DNA at an early stage to avoid disease's consequences.

The aim is to assist biologist in matching sequences and find regions of abnormal variation that cause a disease.

2) Related Work

Third, there are two similarities algorithm to LCS: (MCS) maximum common sub stream, and Rabin-Karb methods Rabin - Karb is better than MCS which consume more time.

In 2012, Rubi and Arockiam suggested positional LCS to reduce time complexity, it become useful in sequence database (SDB) applications, like examine the arrangement of DNA/protein sequence. At the same year, to compare DNA sequences, Elsmady and Nuser introduce and evaluate two algorithms {long common substring LCS & longest common subsequence LCSS} and measure the precision of these two methods with variety code.

In 2011, Wang use the Dominant Point approach to calculate LCS with any number of strings, this method based on Divide and Conquer.

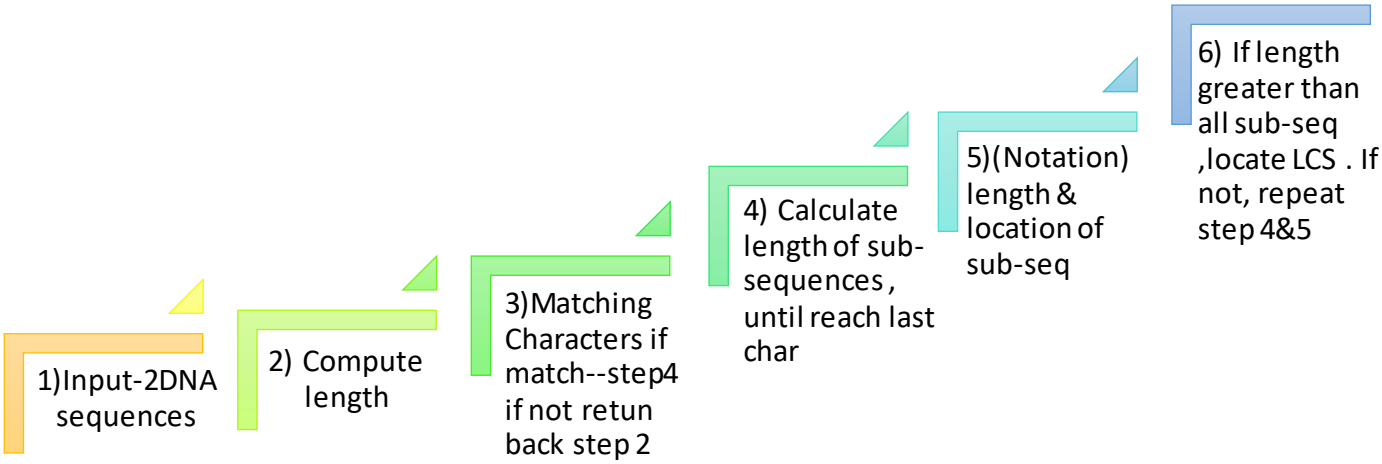
In 2010, Shukla and Agarwal determine LCS through computing relative positions of characters.

In 2015, Beal suggested that it could be solve LCS used reference-based compression tool, and the suffix tree approach identify the common substring in two sequences. Then, build direct acyclic graph (DAG) through most common substrings and define the longest path of the string.

Many biologists looked at the LCS problem tried to solve it using different techniques including, PC cluster-based parallel algorithm, chemical reaction optimization, and a fuzzy approach for multiple sequence alignment (MSA).

3) Methodology & Results:

Finally, to compute LCS between two different sequences, via MATLAB R2012b, stored DNA in an array (DNA1& DNA2 as X, Y) and applied matching process (Z where time similarity elements in arrays stored), therefore, we can notice LCS and calculate complexity. All data used were at the National Center for Biotechnology Information (NCBI) and written in FASTA format [header-sequence].



The result shown as a group of tables: -

First one recording sub-seq, length, and locus to find subsequences, for example {subsequence=GAT, length=3, location=7}.

Second table to search for subsequence with maximum length, it contains length of sequence, length, and locus. NCBI accession num (DNA1) is HG813240.1 & length: 4,412,379 bp, accession num (DNA2) is CS191411.1 & length: 16,966 bp.

Third table (in analysis step), they also calculate the computation time with five different length for example {Length of DNA-seq =300, LCS=TCGTCGAG, Location=105, Length=8, Time=5.004s}.

Last table implicate length of DNA, time computation for both algorithms [matching & information retrieving processes] +Total-time with ROC [Receiver Operating Characteristic curve] to observe relation between DNA sequence size (Byte) and time consumption for (main matching task, traversing array, total time).
Matching algorithm with Pseudo code to search about string-match, so start with length=0, if char from Seq1 match with char from Seq2 increment length by one, repeat process until reach last char in both sequences. There are variety of codes [either dependent on index or char], but all codes arrive to same result. Then, they write another code to search about maximum substring from all matching characters and its position.

Some Reference:

Alsmadi, I. and Nuser, M. (2012) 'String matching evaluation methods for DNA comparison', *International Journal of Advanced Science and Technology*, Vol. 47, pp.13–32.

Rizvi, S.A.M. and Agarwal, P. (2005) 'A new index-based parallel algorithm for finding longest common subsequence in multiple DNA sequences', *International Conference in Cognitive Systems*.