A Novel Statistical Cut-Strategy for DP-based Multiple Biosequence Alignment

Ying-Tung Hsiao*, Cheng-Long Chuang#, Chun-Hung Mo*, Cheng-Chih Chien* and Joe-Air Jiang#

*Department of Electrical Engineering
Tamkang University, Taipei, Taiwan, 251, R.O.C.
Tel: +886-2-26215656 Ext. 2786
E-mail: hsiao@ee.tku.edu.tw

#Department of Bio-Industrial Mechatronics Engineering
National Taiwan University, Taipei, Taiwan, 106, R.O.C.
Tel: +886-2-33665341
E-Mail: jajiang@ccms.ntu.edu.tw

***Abstract*—In this paper, a novel cut-strategy is presented for solving the problems of multiple biosequence alignment. Sequence comparison is the most important primitive operation for analyzing of the bioinformatics data. The most fundamental method for alignment of several biosequences is the dynamic programming (DP) technique. The DP method is capable of finding optimal alignments for a set of sequences. However, when the length of the sequences increased, the DP method is impracticable due to the computational complexity is extremely high. Therefore, a new method is proposed in this paper for reducing the computational cost of the DP technique. By recursively finding the structural features of the biosequences, the proposed method can divide the biosequences into very small alignment problem, which can be directly solved by DP, or other applicable methods that can produce the results of alignment faster. By utilizing the object-oriented programming technique, the proposed method also provides low memory space consumption during execution. Moreover, the proposed algorithm has been implemented in an x86 demonstration program, and compares the effective and efficient performance with other known method.**

***Index Terms*—Bioinformatics, bimolecular sequences, multiple sequence alignment, cut-strategy, and dynamic programming.**

I. Introduction

RECENTLY, the bioinformatics is becoming to a more popular topic in the field of computational biology. The most primitive operation in bioinformatics is the biosequences alignment. The computational requirement of performing multiple biosequence alignment is extremely high. Therefore, to develop a method for reducing the computational complexity is necessary.

In nature, four nucleotides molecules were found: adenine, cytosine, guanine, and thymine. The genomics of all living organisms are formed in a series of these four nucleotides, and they are usually referred to as A, C, G, and T. For biomolecular, such as deoxyribonucleic acid (DNA), ribonucleic acid (RNA), or protein sequence, is formed in a linearly string, which consists of many small units of nucleotides molecules. These data can be analyzed for developing new drugs and other medical usage. A technique called biomolecular sequence alignment (BSA) is developed for finding the maximum similarities between multiple biosequences.

The most common and traditional method for BSA is the dynamic programming (DP) technique [1]. The DP technique was designed for alignment of two sequences, simultaneously, and it is usually very effective when aligning a pair of short sequences. However, for alignment of multiple dimensional sequences with extensive lengths, the DP technique is generally useless and impracticable due to high computational complexity and huge memory space consumption [2]. For some famous bioinformatics analysis systems, such as BLAST [3] and FASTA [4], that developed for reducing the computational requirements, work well when performing alignment between a pair of sequences with thousands of nucleotides, but the performances of the systems are significantly degrades when the lengths of the sequences are increased to millions of nucleotides [5].

To solve a high computational complexity alignment problem, a divide-and-conquer method can easily reduce the searching space for the DP technique. However, finding the cut-point is usually tough as the original problem as well. In this paper, we present a new cut-strategy that is simply analyzing the structural features between different pairs of the biosequences, the original biosequences are been recursively track down into several sub-problems, which is small enough that can be directly solved by employing DP technique. The solutions of these sub-problems are then combined together as the final solution of the longest common subsequence (LCS) problem.

This paper is organized as follow: the definitions of biosequence alignment are provided in Section II. The dynamic programming technique is introduced in Section III. The new cut-strategy for reducing the computational complexity of DP technique is proposed in Section IV. Section V presents the experiment results of the proposed cut-strategy, and comprehensive comparison with other known methods. The conclusions and discussions are provided in Section VI.

II. Biosequence Alignment

The biomolecular sequence that we discussed in this paper can be defined as that a sequence labeled as k with length of m, $s^k = (s_1^k, s_2^k, ..., s_m^k)$, where s_i^k is the ith element of the sequence s^k. The alphabet for a biomolecular sequence consists of a set of four nucleotides, which can be defined as $\Psi = \{A, C, G, T\}$.

For multiple biosequence alignment, the primary objective is to find the longest common subsequences between a set of biosequences $\{s^1, s^2, \ldots, s^n\}$. During the process of the genomic evolution, the biomolecular sequences not only evolve by point mutation, some nucleotides might be inserted into the biomolecular sequences or deleted from them, as well. Therefore, in order to produce a meaningful alignment results between the biomolecular sequences, some gaps are usually necessary to inserted into them. And the gap can be denoted by '–', so the alignment results can be defined as a set of biosequences $\{a^1, a^2, \ldots, a^n\} \in (\Psi \cup \text{'–'})^m$, where $m \geq \mathbf{max}\{m^1, m^2, \ldots, m^n\}$.

The gaps shall be inserted into the biomolecular sequences where it is actually needed, and the gap insertion must be costly to prevent the biomolecular sequences from inserting gaps everywhere. Therefore, gap penalties functions are widely used because of that they are effective to lead the dynamic programming algorithm to produces a meaningful alignment result and compute the similarity between the biomolecular sequences. However, the computational complexity of some arbitrary penalties functions is too high, that makes the multiple biosequences alignment becoming more impracticable. To overcome this problem, the proportional gap penalty function is used for reducing the computational complexity of parallel alignment of several biosequences.

The most common method for multiple biosequence alignment is dynamic programming technique. However, the computational complexity of DP-based multiple biosequence alignment is extremely huge, and it requires high computational power computer with giant volumes of memory to perform such analyses. Conventional computational biology researches were performed on supercomputers, which meet the requirement of performing multiple biosequence alignment. However, as the bioinformatics researches becomes more and more popular, it is necessary to develop a new way of analysis for researchers to do their researches on personal computers with promising results.

III. Dynamic Programming

As mentioned in Section II, the most traditional method for multiple biosequence alignment is dynamic programming technique. In computational biology, comparing of two or more DNA from different organism is the most important and primitive operation. The goal of comparing several DNA strands is to determine the degree of similarity between them, and it means the relations between them, as well.

The dynamic programming is an approach that is developed to solve sequential, or multi-stage, decision problems. The essence of dynamic programming is Richard Bellman's *Principle of Optimality*. This principle is quite instinctive as follow:

"*An optimal policy has the property that whatever the initial state and the initial decisions are; the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision.*"

In this brief introduction, the similarity determination problem was formulated as the longest common subsequence (LCS) problem. Given two sequences X and Y, and wish to find a maximum-length common subsequence of X and Y, we call that as sequence Z, if Z is a subsequence of both X and Y. The LCS problem can be solved efficiently using dynamic programming. For example, consider the DNA sequences X and Y with length m and n, respectively:

$$X = \text{GATCGGA}$$
$$Y = \text{GACCGGA} \quad (1)$$

where the dynamic programming procedure takes two matrices with dimension $m \times n$ as storages of the trace back information. The DP algorithm for solving LCS problem can be written in pseudo-code as follow:

```
LCS_Solution_Algorithm (X, Y)
  m=length(X);
  n=length(Y);
  for i=1 to m
    for j=1 to n
      if Xi=Yj
        C(i,j)=C(i-1,j-1)+1;
        B(i,j)="↖";
      else
        if C(i-1,j)≥C(i,j-1)
          C(i,j)=C(i-1,j);
          B(i,j)="↑";
        else
          C(i,j)=C(i,j-1);
          B(i,j)="←";
End LCS_Solution_Algorithm;
```

where C and B matrices are recording the LCS length and trace back direction. B matrix returned by LCS procedure can be used to construct a longest common subsequence. It is simply beginning from the lower right-hand corner $B(m,n)$ and trace through the matrix by tracing the arrows. When we meet a "↖" in the matrix entry $B(i,j)$ implies that $X_i = Y_j$ is an element in LCS. Therefore, the alignment results of DNA strands X and Y, and the LCS Z are as follow:

		0	1	2	3	4	5	6	7
	y_j		G	A	C	C	G	G	A
0	x_i	0	0	0	0	0	0	0	0
1	G	0	1	1	1	1	1	1	1
2	A	0	1	2	2	2	2	2	2
3	T	0	1	2	2	2	2	2	2
4	C	0	1	2	3	3	3	3	3
5	G	0	1	2	3	3	4	4	4
6	G	0	1	2	3	3	4	5	5
7	A	0	1	2	3	3	4	5	6

Fig 1. *C* and *B* matrices computed by *LCS Solution Algorithm*.

X = GA–TCGGA
Y = GAC–CGGA (2)
Z = GACGGA

As the example given as is stated above, we can know that the DP technique is a very effective method for alignment between a pair of biosequences. The DP technique that commonly used for pairwise sequences alignment can theoretically be extended to solve multiple dimensional alignment problems. However, the computational complexity and memory volume requirement of the DP technique increase exponentially with the amount of the biosequences. For example, if we want to solve a multiple biosequence alignment problem, that the amount of the original biosequences is n, and the average length of these biosequences is m, the computational complexity and memory volume requirement are $O(m^n)$ and $O(2m^n)$ respectively. Therefore, the DP technique is generally impracticable when n is greater than 2 or m is greater than 300.

Due to the DP technique is computational expensive for multiple biosequence alignment, in this paper, we present a structural-based cut-strategy to divide the original problems into several small units of problems, which can significantly reduce the searching space for DP technique, is introduced in next section.

IV. Proposed Cut-Strategy for DP Technique

The divide-and-conquer method has been widely used in many algorithms, such as array search algorithms and sorting algorithms. For DP technique, the divide-and-conquer method is also useful for speeding up the DP algorithm. If an anchor point (cut point) happens to reside near the middle of the final alignment, the greatest speeding up is possible. Therefore, find a cut by which the resultant global optimal alignment is divided into two parts, the original problem is reduced to two smaller subproblems. By recursively applying the procedure, we can finally track down the original problem into very small ones which may be solved directly. However, finding the optimal cutting plane is as difficult as the original problem.

In this work, we present a structural statistic cutting method for reducing the searching space of DP technique. For a set of biosequences $\{s^1, s^2,\ldots, s^n\}$, we predetermine the s^1 as the main alignment target DNA strand, a^n, c^n, g^n and t^n are amount of adenine, cytosine, guanine, and thymine in the nth DNA strand, respectively. And as^n, cs^n, gs^n and ts^n are segment amount of adenine, cytosine, guanine, and thymine in the nth DNA strand, respectively. Therefore, the average amount of adenine, cytosine, guanine, and thymine within the n DNA strands can be defined as A_{Avg}, C_{Avg}, G_{Avg}, and T_{Avg}, respectively. For example the A_{Avg} can be obtained simply apply the following equation:

$$A_{Avg} = \left(\sum_n a^n \right) \Big/ n \tag{3}$$

Based on the main alignment target DNA strand s^1, the residue amount differences of adenine, cytosine, guanine, and thymine between the n DNA strands are be denoted by A_{Diff}, C_{Diff}, G_{Diff}, and T_{Diff}, respectively, which can be obtained by the following equation:

$$A_{Diff} = a^1 \times n - A_{Avg} \times n \tag{4}$$

$$C_{Diff} = c^1 \times n - C_{Avg} \times n \tag{5}$$

$$G_{Diff} = g^1 \times n - G_{Avg} \times n \tag{6}$$

$$T_{Diff} = t^1 \times n - T_{Avg} \times n \tag{7}$$

The average segment amount of adenine, cytosine, guanine, and thymine within the n DNA strands can be defined as AS_{Avg}, CS_{Avg}, GS_{Avg}, and TS_{Avg}, respectively. For example the AS_{Avg} can be obtained simply apply the following equation:

$$AS_{Avg} = \left(\sum_n as^n \right) \Big/ n \tag{8}$$

Based on the main alignment target DNA strand s^1, the segment amount differences of adenine, cytosine, guanine, and thymine between the n DNA strands are be denoted by AS_{Diff}, CS_{Diff}, GS_{Diff}, and TS_{Diff}, respectively, which can be obtained by the following equations:

$$AS_{Diff} = as^1 \times n - AS_{Avg} \times n \tag{9}$$

$$CS_{Diff} = cs^1 \times n - CS_{Avg} \times n \tag{10}$$

$$GS_{Diff} = gs^1 \times n - GS_{Avg} \times n \tag{11}$$

$$TS_{Diff} = ts^1 \times n - TS_{Avg} \times n \tag{12}$$

After the residue amount differences and the segment amount differences have been obtained, residue weighting and segment weighting, which are denoted by W_a and W_s, respectively, are defined to determine which kind of nucleotide represents the most important structural feature within the set of biosequences $\{s^1, s^2,\ldots, s^n\}$. The costs of each kind of nucleotides can be denoted by A_{Cost}, C_{Cost}, G_{Cost}, and T_{Cost}, respectively, which can be obtained by the following equations:

$$A_{Cost} = W_a \times A_{Diff} + W_s \times AS_{Diff} \tag{13}$$

$$C_{Cost} = W_a \times C_{Diff} + W_s \times CS_{Diff} \tag{14}$$

$$G_{Cost} = W_a \times G_{Diff} + W_s \times GS_{Diff} \tag{15}$$

$$T_{Cost} = W_a \times T_{Diff} + W_s \times TS_{Diff} \tag{16}$$

The nucleotide that has the lowest cost represents that it plays an important feature between all biosequences due to the structural construction is relatively close.

For a nucleotide that has been considered as the most important feature within all biosequences, the proposed algorithm will start to match the segments of the featured nucleotide in biosequence s^1 with other biosequences. The searching range is as the equation defined as below:

$$R = MSB(S(s^1, x)) \pm C \times Length(S(s^1, x)) \tag{17}$$

where R is the index searching range within the biosequences, $S(s^1,x)$ is the xth segment of the featured nucleotide, MSB is the most significant byte index of the referenced segment, and $Length$ is the length of the referenced segment. C is a parameter for controlling the size of the searching range. If C is a big number, it is easy for the proposed algorithm to search for

matched featured segments between the biosequences. However, it could lead the algorithm to produces a non-optimal alignment due to non-optimal match. Therefore, C is typically being settled in a small number.

During the matching process, a scoring function has been defined to determine whether the pair of featured segments is meaningful or not. The scoring function ξ can be formulated as follow:

$$\xi = Match - Dist / C \tag{18}$$

where the *Match* and *Dist* are the amount of residue matched between a pair of featured segment and the distance between them, respectively. The score ξ of the pair of featured segments must be positive to ensure that the matching between them is meaningful. Moreover, for each featured segment, it will always remain the match with highest score with other featured segment. For example, there is a pair of biosequences as shown in follows:

$$\begin{aligned} X &= \mathbf{AAA}\text{CCCTT}\mathbf{A}\text{GCCTTCC}\mathbf{AAAA} \\ Y &= \text{CC}\mathbf{AAA}\text{CCTTGGCC}\mathbf{AA}\text{TTC}\mathbf{AAA} \end{aligned} \tag{19}$$

According to the cost calculation equation in (13)-(16), the adenine nucleotide (A) is determined as the featured nucleotide that has a relatively close structure between X and Y. Within the biosequences X and Y, the segments with bold A are the featured segments. The first and third featured segments within biosequences X and Y are meaningful because of that their matching scores are both positive. However, the second featured segment in Y has a positive score (ξ =0.5) with the third featured segment in X as well, but the score is smaller than the third featured segment in Y matching with the third featured segment in X (ξ =3). Therefore, the matching relationship between the third featured segment in X and the second featured segment in Y is dissolution. The matching result of the feature segments within X and Y is as shown in follow:

$$\begin{aligned} X &= \underline{\mathbf{AAA}}\text{CCCTT}\mathbf{A}\text{GCCTTCC}\underline{\mathbf{AAAA}} \\ &\quad\ \nwarrow \xi=2 \qquad\qquad\qquad \nwarrow \xi=2.5 \\ Y &= \text{CC}\underline{\mathbf{AAA}}\text{CCTTGGCC}\mathbf{AA}\text{TTC}\underline{\mathbf{AAA}} \end{aligned} \tag{20}$$

where the featured segments with underline are matched featured segments, and the arrows between them indicate the matching pairs of featured segments between X and Y.

While the pairs of featured segments have been identified, the original biosequences are then being divided into several sub-biosequences as follow:

$$\begin{aligned} X_1 &= \text{AAA} \quad X_2 = \text{CCCTTAGCCTTCCAAAA} \\ Y_1 &= \text{CCAAA} \quad Y_2 = \text{CCTTGGCCAATTCAAA} \end{aligned} \tag{21}$$

Then recursively apply the cut-strategy that proposed in this section to these sub-biosequences, until one of the following stop-criterions are satisfied:

1) The length of the subsequences is equal or smaller than a predetermined minima length $L_{\min}$.

2) The content of the pair of sub-biosequences are exactly the same.

3) If there is no matched pair for the primary featured nucleotide, the secondary featured nucleotide (with the cost of second lowest) will replace it (the same as the third and fourth featured nucleotides), till there is no other featured segment existed between sub-biosequences, then this stop-criterion is satisfied.

Moreover, in order to avoid the dividing process from deadlock, the same nucleotide cannot be the featured nucleotide continuously during the recursively dividing process.

Till all of the sub-biosequences satisfies the stop-criterions, the DP technique is then applied to all of the sub-biosequences, and search for the LCS between them. The final alignment result of entire biosequences is constructed by combining the LCS searching results of every sub-sequence. A completely alignment example of the proposed method, that the original biosequences is the same as given in (19), is provided in Fig. 2, and the parameter settings are W_a=1, W_s=2, C=2, and $L_{\min}$=4. As we can see in this example, that if we directly apply DP technique on the original biosequences, the total computational requirement is $O(20\times21)=O(420)$, and the memory requirement is $O(2\times20\times21)=O(840)$. However, if we utilize our proposed cut-strategy before utilizing the DP technique, the computational requirement for DP processes is reduced to $O(2\times5+5\times4+4\times4+2\times4+6\times4)=O(78)$; based on the object-oriented programming technique, the memory requirement is pruned to maximum of $O(24)$, and the alignment result is promising, as well. The reduction rate of the searching space is inversely proportional to the value of $L_{\min}$. For $L_{\min}$=3, the computational requirement for DP processes is even down to $O(43)$, and the memory requirement is maximum of $O(15)$.

As another example that has already been mentioned, as shown as in Fig. 1, that was processed by the proposed cut-strategy. Which we can see that the size of the original searching space is 49 grids, by utilizing the proposed

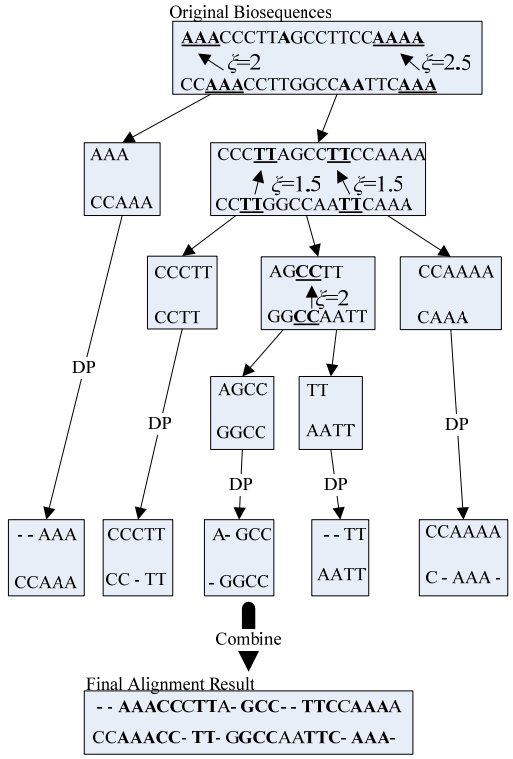

Fig. 2. Illustration of the decomposition and alignment procedures of the proposed cut-strategy ($L_{\min}$=4).

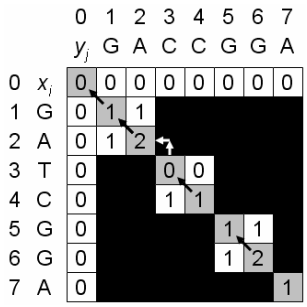

Fig. 3. Illustration of the reduction of searching space of the proposed cut-strategy (L_{min}=2).

TABLE 1.
List of Virus DNA used for evaluation.

Type	Nucleotide Accession	Max bp	Avg. bp
HIV	**AY290947**, AY290946, AY290944, AY290933, AY290930, AY290929, AY290928, AY290925.	450	445
Influenza A Virus	**AY664773**, AY664771, AY664770, AY664769, AY664767, AY664766, AY664765, AY664760.	2191	1352.5
SARS	**AY502932**, AY502931, AY502930, AY502929, AY502929, AY282752, AY278491, AY278554.	29742	29731.9

(The bold accessions indicate to be the primary alignment target.)

cut-strategy. Which we can see that the size of the original searching space is 49 grids, by utilizing the proposed cut-strategy, the searching space is successfully reduced to 13 grids, as shown as in Fig. 3. Therefore, we can see that the proposed method is capable of reducing searching space of the biosequences alignment problem.

V. Experimental Results

In this section, several types of real viruses are presented for evaluating the novel cut-strategy that is proposed in this paper. The viruses that used for evaluating are human immunodeficiency virus (HIV), influenza 'A' virus, and severe acute respiratory syndrome corona-virus (SARS), respectively, as listed in Table 1. The evaluation samples are obtained from the Entrez Protein web database for simulation [6].

The simulation program is implemented on a P4-2.0 GHz personal computer with 1024MB RAM by C language. The parameters for the proposed cut-strategy are W_a=1, W_s=2, C=2, (where the values of W_a, W_s and C are always suggested) and L_{min}=3 (where the value of L_{min} is adjustable, and should be settled in the interval between 2 and 4). The simulation results are illustrated in Fig. 4. It shows that the computational time to the sequence number increase almost linearly. The amount of matched nucleotides between different numbers of biosequences is provided in Table 2. Thus, we can see that the proposed cut-strategy was successfully achieved the goal of reducing searching space of the biosequences alignment problems with promising results.

Note: The amount of matched nucleotides drops significantly when we try to simultaneously align four IAV samples due to

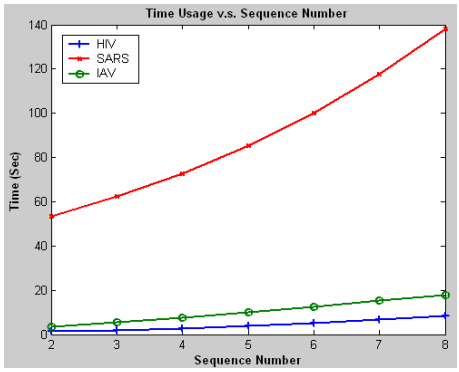

Fig. 3. Computational time of the proposed cut-strategy for alignment on different sequences listed in TABLE 1.

TABLE 2
Amounts of matched nucleotides
between difference number of biosequences.

	2	3	4	5	6	7	8
HIV	442	435	435	416	416	416	416
IAV	1222	1216	596	596	596	594	594
SARS	29690	29690	29690	29687	29670	29668	29660

the length of IAV sample AY664769 is only 669 bp.

VI. Conclusion

The sequence alignment has been widely utilized in the area of computational biology researches. There are many popular algorithms has been applied to solve the biosequence alignment problems, such as dynamic programming, hashing method, and some heuristic methods. However, for multiple biosequence alignment problems, they became much more complicated due to the searching space to number of biosequences is increasing exponentially, and most of the traditional methods are too slow or inaccurate. Therefore, to develop a new method to reduce the searching space with promising alignment result is necessary.

The divide-and-conquer method is known as effective to reduce the searching space for biosequence alignment problem. However, to find the optimal cut-points between different sequences is as hard as the original alignment problem. In this work, we presented a statistics method to extract the structural features, and use the information for dividing the biosequences into several extremely small sub-biosequences, then directly apply DP technique to solve the alignment problems. The experimental results show that the proposed cut-strategy is able of achieving high reducing rate on the searching space for DP technique to find the best alignment faster, and also capable of producing promised results.

In this work, a statistical method for dividing several long biosequences into small ones, which is the main idea of this paper, is proposed. Please note that the demonstrated results show that the proposed method is able to turn the pure DP method, which is known as a computational expensive algorithm, into a linear computational complexity algorithm. The performance plot shown in Fig. 3 can be improved if we apply other modern methods instead of the original DP method.

REFERENCES

[1] S. M. Waterman, "General methods of sequence comparison*," Byll. Math. Biol.*, vol. 46, 1984, pp. 473-500.

[2] P. Clote and R. Backofen, *Computational Molecular Biology: An Introduction*, Jhon Wiley and Sons, Ltd., Chichester, England, 2000.

[3] R.C. Gonzalez and R.E. Woods, *Digital Image Processing, second ed.* Prentice Hall, 2001.

[4] W.R. Pearson, "Comparison of methods for searching protein sequence databases," *Protein Sciences*, vol. 4, no. 6, June 1995, pp. 1145-1160.

[5] A.L. Delcher, S. Kasif, R.D. Fleischmann, J. Peterson, O. White, and S. Salzberg, "Alignment of whole genomes," *Nucleic Acids Research*, vol. 27, no. 11, pp. 2369-2376, May 1999.

[6] URL: http://www.ncbi.nlm.nih.gov/entrez/query.fcgi

[7] T.H. Cormen, C.E. Leiserson, and R.L. Rivest*, Introduction to algorithms, second ed.*, McGraw-Hill, Cambridge, Massachusetts London, England, 2001.

[8] J.C. Setubal and J. Meidanis, *Introduction to computational molecular biology*, Brooks/Cole publishing company, USA, 1997.

[9] A. Griffiths, W. Gelbart, J. Miller, and R. Lewontin, *Modern genetic analysis*, W.H. freeman and company, New York, 1999.

[10] S. Skiena, *The algorithm design manual*, Springer, New York, 1998.

[11] B. Bergeron*, Bioinformatics Computing*, Prentice Hall PTR, 2002.

[12] D. Balode, A. Ferdats, I. Dievberna, L. Viksna, B. Rozentale, T. Kolupajeva, V. Konicheva, and T. Leitner, "Rapid Epidemic Spread of HIV Type 1 Subtype A1 among Intravenous Drug Users in Latvia and Slower Spread of Subtype B among Other Risk Groups," *J. AIDS Res. Hum. Retroviruses*, vol. 20, no. 2, pp.245-249, 2004.

[13] S.-H. Yeh, H.-Y. Wang, C.-Y. Tsai, C.-L. Kao, J.-Y. Yang, H.-W. Liu, I.-J. Su, S.-F. Tsai, D.-S. Chen, and P.-J. Chen, "Characterization of severe acute respiratory syndrome coronavirus genomes in Taiwan: Molecular epidemiology and genome evolution," *Proc. Natl. Acad. Sci. U.S.A.*, vol. 101, no. 8, pp. 2542-2547, 2004.