Netw Model Anal Health Inform Bioinforma (2014) 3:68
DOI 10.1007/s13721-014-0068-8

ORIGINAL ARTICLE

An integrated algorithm for local sequence alignment

Sarwar Kamal · Mohammad Ibrahim Khan

Received: 27 April 2014 / Revised: 22 July 2014 / Accepted: 25 July 2014 / Published online: 19 August 2014

Abstract Local sequence alignment (LSA) is an essential part of DNA sequencing. LSA helps to identify the facts in biological identity, criminal investigations, disease identification, drug design and research. Large volume of biological data makes difficulties to the performance of efficient analysis and proper management of data in small space has become a serious issue. We have subdivided the data sets into various segments to reduce the data sets as well as for efficient memory use. The integration of dynamic programming (DP) and Chapman–Kolmogorov equations (CKE) makes the analysis faster. The subdivision process is named data reducing process (DRP). DRP is imposed before DP and CKE. This approach needs less space compared with other methods and the time requirement is also improved.

Keywords Dynamic programming · Chapman–Kolmogorov equations · Data reducing process

1 Introduction

In our previous work (Khan and Kamal 2013a, b) we have used hidden Markov model, neural network (NN), and suffix tree (ST). The outcomes are good, but it takes more space and time. Besides, we also checked the DNA damages for ontological alignment (Khan and Kamal 2013a, b).

S. Kamal (✉) · M. I. Khan
Department of Computer Science and Engineering, Chittagong University of Engineering and Technology, Chittagong, Bangladesh
e-mail: sarwar.saubdcoxbazar@gmail.com

M. I. Khan
e-mail: muhammad_ikhancuet@yahoo.com

In 1975, Frederick Sanger and his colleagues, and Alan Maxam and Walter Gilbert developed two different methods for determining the exact base sequence of a cloned piece of DNA (Robert 2002). These spectacular breakthroughs revolutionized molecular biology. They have allowed molecular biologists to determine the sequences of thousands of genes and they have even made it possible to sequence all three billion base pairs of the human genome (Robert 2002). It has made very positive impacts on functional identification of newly sequenced genes and building reshaped DNA sequences (Doolittle 1996).

Local sequence alignment locates the best approximate subsequence match within two given subsequences. Sequences might be DNA, RNA and protein sequences in biological sequences. Besides LSAs, gapless alignment such as BLAST (Altschul et al. 1990) or FASTA (Lipman and Pearson 1985, 1988) is examined (Arratia et al. 1988; Dembo and Karlin 1990; Karlin and Altschul 1990, 1993). We have noticed that some alignments with gap also play an important role to decide the proper match for the exact sequence matching. We have examined the alignments with gap such as Needleman–Wunsch (Needleman and Wunsch 1970) or Smith–Waterman (Smith and Waterman 1981; Waterman 1989, 1994) algorithms. Dynamic programming is also considered in these alignments (Batzoglou et al. 2000a, b).

Generally, nucleotides are arranged along two standards of double helix, called the forward and reverse standards with the nucleotides of one stand bonding with complementary nucleotides on the others. Normally, adenine nucleotides bond only with thymine, and cytosine nucleotides only bond with guanine; the bonded nucleotide is called the basepair (bp). Sequencing consist of four bases of DNA double strand structure. Suppose $P = a_1, a_2 \cdots a_m$ and $Q = b_1, b_2 \cdots b_n (n \leq m)$ over alphabet $\sum$. The general

case of alignment is to match the similarities of these two strings under the name of local, global or semi-local alignments.

2 Related work

According to the various algorithms, implemented through 2001 to 2005, Ning et al. (2001), Kent (2002), Furey et al. (2002), Schwarz et al. (2003) and Watanabe et al. (2005) examined the accuracy of the sequencing. In the age of information superhighway, we know that the genome sequences may have continual or near continual patterns in the given or collected data sets. As a result, the outcomes might be the same for many positions. On the contrary, the mutations and *Indel* may generate incorrect judgments for sequencing and mapping, whether it is local, global or pairwise alignment or mapping. The algorithms above do not consider the situations regarding the repetitions of the patterns, mutations and incorrect mapping. Here, we have noticed that the system rejects the data sets, makes the area small and as a consequence the calculations become complicated as well as wrong. Ewing and Green (1998) proposed a solution to overcome the ambiguity, but this method produces low-quality regions. Here, we have implemented the Markov chain concept under Chapman–Kolmogorov equations to establish probable sequenced positions. In the field of sequencing, there are a number of softwares that help to detect the sequences and the frequently used systems are PolyPhred (Stephens et al. 2006), SNP detector (Zhang et al. 2005), and novoSNP (Weckx et al. 2005). But these three systems only detect genotype sample. They are unable to solve the dynamic patterns and sequences. Our systems will cover these problems under the bounding box scheme. The softwares that are used for sequencing are mainly classified broadly into two categories: training based and detection based. The detection-based category depends on homolog finding. But the homolog finding mainly depends on database finding (Claverie et al. 1997). However, database finding is not always efficient due to the size and cost of the equipments. Training-based sequencing is not able to identify the proper coding regions due to the lack of generalizations (Pati et al. 2010). Besides, predictions are vulnerable with many false-positives identifications and sensitivities (Yok and Rosen 2011). Nowadays, many researchers have focused on the efficiency and correctness of predictions. The latest methods have been designed on the basis of advanced annotations or predictions in alignments and matching by removing false predictions (van Baren and Brent 2006) or generating new genome sequences. However, there is some scope of uncertainty in the generated sequences, which are convergent, bounded or monotonic.

Frameshift identification (Tech and Meinicke 2006) is an important part of the sequencing, whether cross-checking of genes or adjustment of translational initiation site (Stormo et al. 1982; Zhu et al. 2004, 2007). According to Wu et al. 2006, phylogenic fingerprint DNA sequencing is improved, but the convergent features are not checked. In the same field of multiple sequencing of proteins (Keller et al. 2011), there are no guarantees of finiteness checking. Besides, there are some established gene identifiers such as ORPHEUS (Fleischmann et al. 1995), AUGUSTUS (Keller et al. 2011), SLAM (Alexandersson et al. 2003) and GenePrimp (Stormo et al. 1982) that have strictly mapped the genes. To accomplish efficiency as well as reduce sensitivity, we have designed a complete package that coordinates the stages by maintaining the linkage and arrangements of sequences. Due to the lack of linkage of methods, the sequencing does not always answer proper actions. In ORPHEUS, we have noticed that the system does not take invariant information at the stage of matching. It also proposed that (Dhar et al. 2009) unknown motifs may be difficult to identify from intergenic regions. We have checked the alignments by OHSUMED data set. OHSUMED data set is a very informative data house for medical document classification and identifications. The OHSUMED data set, which was created for the TREC conference, has become an evaluation benchmark in automatic information classification research since 1994 (Yetisgen-Yildiz and Pratt 2005). OHSUMED is composed of 348,566 MEDLINE documents from 270 journals published between 1987 and 1991. The documents are categorized under the environment of 14,321 MeSH classes. We have selected the OHSUMED due to its smaller amount of spaces on storing and manipulating. For example, there are 753 classes with only 1 document in OHSUMED. The research result and implementation of Ruiz and Srinivasan (1999), and Lewis (1992, 1996) used 49 categories related to heart diseases with at least five training documents. The data set related to DNA and diseases has been commonly used in DNA classification as well as for the identification of specific patterns from a given DNA set and DNA collections.

3 LSA and integration

Local sequence alignment is a process that enables the sequence to match the sequence at specific parts of the total given DNA sequences. Local alignment may be on the basis of pairwise comparison of sequences, whatever the length. Local alignment is very important in DNA testing for medical science, law and justice, criminal identification, baby identification and, last but not the least, diseases identifications. Local alignment works by finding small

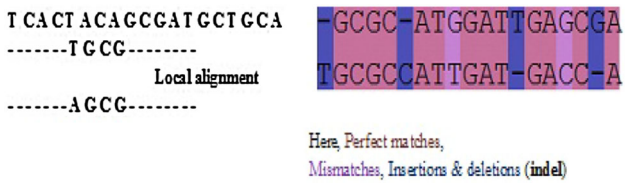

Fig. 1 The process of LSAs. Perfect matches, Mismatches, Insertions and deletions (*indel*) are marked clearly using various *colors*. The fundamental basic for the local alignment sequencing are these three measurements

polynucleotides (e.g., UAA, UCG, ATG) or any pattern that is responsible for the identification of the desired identity. Figure 1 shows the brief scenario of local sequencing.

From Fig. 1, we see that the left part of the figure indicates the LSA under the two small lengths (**T G C G** in the first row and **A G C G** in the second row). In this case the local sequence length is four and it may be any length less than or greater than this length. Pairwise alignments are frequently used in repeat match, unique match, contiguous seed, space seed, vector seed and maximum match subsequences (MMSS) (Waqaar et al. 2008), a collection of bases that is also part of the LSA. Recently, sequencing has been performed on the basis of seed selection and then extending the sequence according to the demand of the sequence (Yu et al. 2007). The right part of Fig. 1 shows the operations of the LSA where *indel* is used when there are any deletions or insertions of the nucleotides. The group of nucleotides representing perfect matches indicates the seeds of the LSA. This seeds finding process is accomplished by two phases. In the first phase, the algorithm finds the seeds under consideration of perfect matches, and in the next phase the system increases the length of the seeds toward the MMSS finding. The BLAST algorithm works for the fixed length alignment, whereas our algorithm works with any variable length sequences. We also compare the alignment of our system with MUMmer and MUMmerGPU. We have significantly noticed that RSAM is faster and moderately sensitive than BLAST, MUMmer and MUMmerGPU. Schatz et al. (2007) and Delcher et al. (2002)) depicted that both MUMmer and MUMmerGPU show the same outcome where the sensitivity is very high and the complexity is $O(n^2)$ when the sequences vary their lengths and it is always more sensitive.

Suppose $s(p,n)$ indicate the number of alignments of two sequences of DNA length n and seed size b. Basically, $s(p,n)$ is a count of (0,1) matrices under minimum two sequences or rows of the given DNA and an unknown number of columns, so that both sequences contain precisely n 1s, i.e., each column contains at least one 1. The asymptotic expression of $s(p,n)$ for a specified b as $n \to \infty$, is a function of b:

$$S(p, n) \geq \binom{2n}{n}. \tag{1}$$

According to Stirling's theory as $n \to \infty$ and seed b is fixed,

$$s(p, n) \geq \left((\pi n)^{-0.5}\right)\left(4^n + o(1)\right) \text{ as } n \to \infty. \tag{2}$$

When $s(1,n)$ there is only 2-sequence alignment. It is easy to narrate the sequencing by generating a function when $b \geq 1$:

$$T(x) = (1-x)^2 - 4x\left(x^b - x + 1\right)^2. \tag{3}$$

From Eq. (3) it is possible to decide the alignment possibilities under the finite seeds or blocks.

3.1 Data reducing process

For LSA, it is very important to have the specified length of DNA, RNA or proteins. From the baseline paper (Waqaar et al. 2008), we have reviewed that the author has implemented two different steps: MMSS selection and MMSS anchor selections. MMSS selection by suffix tree is a costly approach due to its complexity in searching the matching at any specified lengths of DNA. The complexity of the suffix tree under this environment is $O(n + m)$ time, where $O(m)$ time is essential to build the suffix tree and $O(n)$ time is used for searching the patterns. Total time is $O(n + m)$. After than to select the anchor length it is essential to have $O(n)$ time. So the total complexity is $O(n + m) \times O(n)$, i.e., the total time complexity for the first two steps is $O(n^2 + mm)$.

We have implemented these two steps by using the *data reducing process* (DRP) length selections of the given DNA where the length of the genome is n and the number of genome is t. The total size of the bounding box is $n \times t$. The size will repeatedly divide until it reaches the unique portions of the nucleotide. Here, unique means single nucleotide. The consideration whether the length of a genome is odd or even is an important factor. The DRP (n, t) algorithm will narrate details about the process. Figure 2 shows the operations of the DRP.

Data reducing process (n, t)

1. Select the length n, with the dimensions t, and the number of genome sequences under the array as $n \times t$.
2. Subdivide the array $n \times t$ to $(n/2 \times t/2)$ in the first phase.
3. Continue step 2 until the subdivision reaches the single nucleotide.

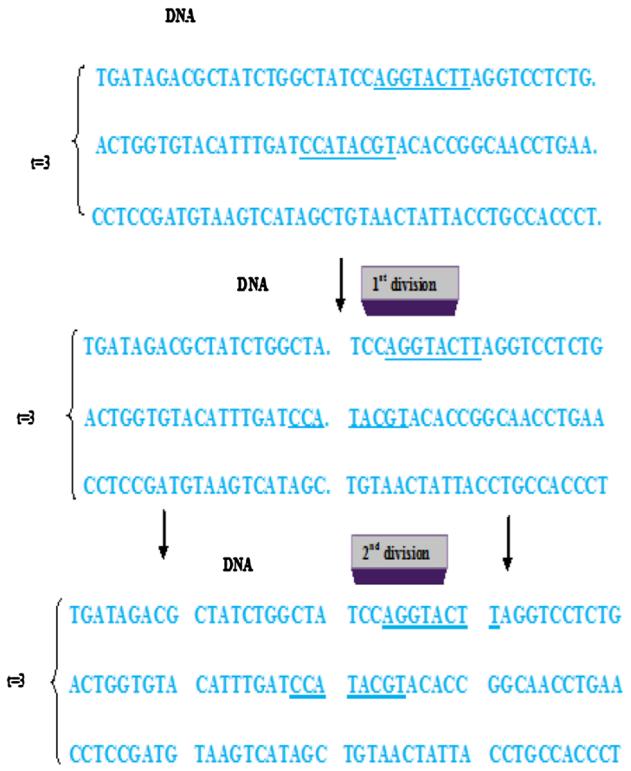

Fig. 2 DRP with length $n = 40$; genome number middle part shows the first division of the selected DNA genomes and the *last part* shows the second division of the same DNA with the same length and genomes toward unique nucleotides. This process will lead to obtain unique nucleotides for LSAs. This DRP is used for the first time in the field of bioinformatics LSAs. This is a novel idea for sequence length selections

$$(n \times t) \cdots (n/k \times t/k) \cdots (n/2k \times t/2k) \cdots (n/4k \times t/4k) \cdots (n/nk \times t/nk).$$

4. Let $(n = 2n + 1)$.
5. $n \leftarrow (n - 1)$.
6. Store $(2n + 1)$th position nucleotides; repeat steps 2 and 3.
7. if ((n/nk×t/nk) == 1)
8. Stop subdivisions.
9. Otherwise,
10. Repeat 2, 3, 4 and 5.
11. Locate, match, mismatch and indel are shown in Fig. 1.

The DRP sign DRP(S) denotes a set of nucleotides S. Here DRP $= (a_1, a_2, a_3 \cdots a_n)$ are various nucleotides elements.

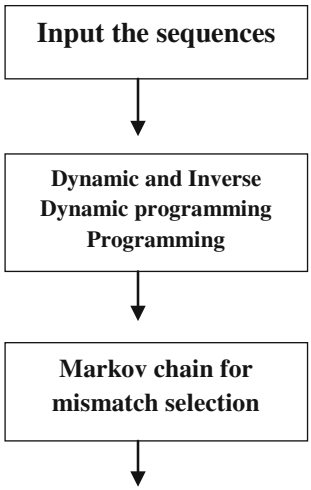

Fig. 3 In this integrated algorithm the figure shows the relationship among various machine learning approaches. The identity of this algorithm is that it contains dynamic programming and set operation along with Markov chain. The process started by taking DNA sequence as input. After taking the sequences dynamic programming is imposed on the respective sequences. This step helps to classify the sequences. At the third step, we see that the set operation makes the distinction of the DNA segments. The next step is a very important part of this algorithm. In this case the selection of matches, mismatches and indel is determined

4 Integration environments

In this algorithm we have checked the alignment by considering individual nucleotides. After taking the input of the sequences, we perform dynamic programming and inverse dynamic programming to check the alignments with set operations and check that the modifications are matches, mismatches and indel (Fig. 3).

Integration Algorithm (MNU, NMNU)
Here,
MNU = matched nucleotide unit
NMNU = non-matched nucleotide unit

1. At first, input the genome sequences and identify the MNU and NMNU from the DRP algorithms.
2. Implement the dynamic programming and inverse dynamic programming to identify pairwise alignments.
3. Use the set operations on the selected part of the alignments.
4. Modify the matches, mismatches and indel.
5. Finally, train the sequencing to identify the MNU and NMNU.
6. Let (NNLN == NMNU).
7. Repeat the step 5.

8. Otherwise,
9. Repeat the steps 1 to 5.
10. End.

4.1 Dynamic and inverse dynamic programming

We have implemented dynamic programming to choose the MMSS. Consequently, we also imply inverse dynamic programming to verify the alignment that does not match.

Suppose

$P = p_l, p_2 \cdots p_n,$ and $Q = q_1, q_2, \ldots q_m,$ are two DNA sequences. The indel of two sequences is denoted by the weight M (g). At first, we consider the best alignments as $F(p, q) = \max\forall(p*, q*)$, where F is a function which relates the current alignments and new alignment. By using dynamic programming we can check the sequences $F(p, q)$ recursively.

$$F(i,j) = F\left(p_1, p_2 \cdots p_i\, q_1, q_2, \ldots q_j\right)$$

where $F(0,O) = O, F(O,j) = F\left(-, q_1, q_2, \ldots q_j\right) = M(j)$ and $F(i, 0) = M(i)$. Then,

$F(i, j) = \left\{F(i-1, j-l) + F\left(p_i, q_j\right), \max\{F(i-k, j) + M(k)\}, \max\{F(i, j-I) + M(l)\}\right.$. For LSA the function $L(p, q) = \max\{F\left(p_u, p_{u+1}, \ldots p_v, q_x, q_{x+1}, \ldots q_y\right) : 1 \le u \le v \le n,\ l \le x \le y \le m\}$. The dynamic algorithmic operation to select maximum matches subsequence for the two DNA sequences P and Q are as follows.

Maximum matches subsequences (Sequence P, Sequence Q)

Maximum Matches Sub Sequences (Sequence P, Sequence Q)
{
Input: Sequence P and Q
Output: The Maximum Matches Sub Sequences.
Reserve: Counting Matrix
G: Check the tracing table (use letter a, b, c)
n=P.length ()
m=Q.length ()
// fill in Reserve and G
for (i=0;i<m+1;i++)
for (j=0;j<n+1;j++)
if (i==0) || (j==0)
then Reserve (i,j)=0;
else if (P[i]==Q[j])
Reserve (i,j)=max { Reserve [i-1,j-1]+1, Reserve [i-1,j], Reserve [i,j-1]}
{update the entry in trace table G }
else
Reserve (i,j)=max { Reserve [i-1,j-1], Reserve [i-1,j], Reserve [i,j-1]}
{update the entry in trace table G }
then, use trace back table G to print out the optimal alignment.
}

Besides the alignment with DP for MMSS, our activity also checked the complement of MMSS by inverse dynamic programming (IDP). The inversion assures the nonmatches nucleotides only. Inversion will consider for both matches and mismatches. The inversions must be the complement of the regular pairwise alignments.

INVERSION(c, d, i, j) $= F_1$ (p_c, p_{c+1}, p_{c+2},…p_i, q_j, q_{j-1},…q_h).

Here,

$$\text{INVERSE(A)} = \text{T}$$
$$\text{INVERSE(C)} = \text{G}$$
$$\text{INVERSE(G)} = \text{C}$$
$$\text{INVERSE(T)} = \text{A}.$$

In the case of regular and authentic sequencing, the genome parts p_c, p_{c+1}, p_{c+2},…p_i, q_j, q_{j+1},…q_h must be matched after inversions.

4.2 Markov chain for mismatch selection

From the baseline work (Waqaar et al. 2008), we have examined the total process of the mismatches seeds and mismatches anchor selection where it cost time complexity $O(n^2 + mn)$ and space complexity as $O(mn(n^2 + m^2))$. We measure mismatches using Chapman–Kolmogorov formula. The formula for a stochastic process with random variable X is $X = \{X_t, t \in T\}$, where $t =$ index and it indicates the time. $X_t =$ state of the process, $T =$ index set constituted by time t.

Suppose $n = 0, 1, 2, 3\ldots$, $m = 1, 2, 3\ldots$ and $i_0 \ldots i_m \in E$. $E =$ all possible values that the random variable X_t can assume. Then,

$$\Pr\{X_{n+1} = i_1, \ldots, X_{n+m} = i_m | X_n = i_0\} = \Pr_{i_0 i_1} \cdot \Pr_{i_1 i_2} \cdots \Pr_{i_{m-1} i_m}.$$

$$\Pr\{X_{n+m} = j | X_n = i\}$$
$$= \sum_{k=0}^{\infty} \Pr\{X_{n+m} = j | X_{n+1} = k, X_n = i\} \Pr\{X_{n+1} = k | X_n = i\}$$
$$\sum_{k=0}^{\infty} \Pr\{X_{n+m} = j | X_{n+1} = k\} \Pr\{X_{n+1} = k | X_n = i\}$$
$$= \sum_{k=0}^{\infty} \Pr_{ik} \Pr_{kj}$$
$$= \Pr_{ij}^{m}$$

In general,

$$\Pr_{ij}^{n+m} = \sum_{k \in E} \Pr_{ik}^{n} \Pr_{kj}^{m} \text{ for all } n, m \ge 0, \text{ all } i, j \in E.$$

5 Implementation

Before starting the experiments we have marginalized the data set by sampling. Based on the sequence nature we have verified the data set by stratified sampling, because it helps to select the subgroups from a defined population so that it represents the total impact of the population. To accomplish this sampling, first of all we have to define and determine the population and sample size. Then, the desired variables have to be selected so that the sample becomes appropriate. Here, the local variables will be the nucleotides and polynucleotides. Finally, the exact classification determines the local sequences. Compared with the baseline paper (Waqaar et al. 2008), we have checked the sensitivity with specificity of the sequencing based on polynucleotide segments as sequence key factors under receiver operating characteristics (ROC).

We have implemented and experimented under the environments of Java with Integrated Development Environment NetBeans. The object-oriented implementation helped us to perform with the nucleotides (A, C, T, and G) as distinct objects. The total process was executed in an object-oriented manner instead of procedural C programming language as in Waqaar et al. (2008). Object orientation enables faster and machine-independent environments over any procedural language. The machine-independent analysis, synthesis and anti-synthesis create a powerful algorithmic procedure. The steps of the algorithm are depicted in Fig. 3.

6 Experiments and result

In most of the cases, the authors have considered one or two matrices to assess the speed and sensitivity of sequencing. We here consider six dominant metrics to identify the performance of our alignment algorithm. The matrices are: speed (X_1), complexity (X_2), space (X_3), sensitivity (X_4), accuracy (X_5) and risk (X_6). For speed, sensitivity and accuracy we measured the referential value as best, average and low. We have checked the complexity, risk, accuracy and space for the first time and many LSA tools measured the sensitivity without any standard parameter. According to the MUMmer, Delcher et al. (2007) termed the parameter '**q**' as the ratio between accurate aligned nucleotides pairs and total number of nucleotides in the given sequence. Total column score is another aspect of the MUMmer procedure. Again according to the AVID (Bray et al. 2003), the authors consider the alignment pairs which have score greater than the predefined threshold value. Instead of all of the methods above, we have concentrated on the set operations under the complete machine learning process on exons and introns. Introns measurement are also an essential part of the alignment to maintain proper checking instead of only one parameter checking (exons). For speed, sensitivity and accuracy the reference values have been checked according to the fuzzy manner, such as best (H), average (M) and low (L).

The measurements of all the parameters mentioned above have been depicted in the following tables (Table 1). We have considered the sequence length randomly starting from 500 thousand to 2,000 thousand. In this work we used Nucleotide Database from National Central for Biotechnology Information, OHSUMED database and ROSETTA (Batzoglou et al.

Table 1 The reference values for speed, sensitivity and accuracy

Parameters	Dependent parameters (antecedent)	Result (consequent)
Speed (X_1)	If X_5 is H $\wedge$ X_3 is H $\wedge$ X_2 is H	Then X_1 is H
	If X_5 is H $\wedge$ X_3 is M $\wedge$ X_2 is M	Then X_1 is $\{(H.0.3),(M,0.7)\}$
	If X_5 is H $\wedge$ X_3 is L $\wedge$ X_2 is L	Then X_1 is $\{(H.0.3),(L,0.7)\}$
Sensitivity (X_4)	If X_5 is H $\wedge$ X_2 is $\wedge$ H	Then X_4 is H
Accuracy (X_5)	If X_5 is H $\wedge$ X_4 is $\wedge$ H $\wedge$ X_6 is H $\wedge$ X_2 is H	Then X_5 is H

Table 2 The speed comparison of three algorithms: BLAST, baseline paper and this system

Iteration	Sequence length (weight)	Speed (ns)			Comparison
		BLAST	Base line paper	This system	
1	500,000	0.63	0.60	0.55	11 % higher
2	600,000	0.69	0.66	0.57	11 % higher
3	700,000	0.71	0.67	0.60	12 % higher
4	800,000	0.73	0.69	0.61	15 % higher
5	900,000	0.80	0.72	0.60	14 % higher
⋮	1,000,000	⋮	⋮	⋮	
	1,100,000				
	⋮				
20	2,000,000	0.95	0.90	0.67	24 % higher

2000a, b) database instead of only one database. It is clearly visible that alignment varies from database to database. Table 2 depicts the speed comparisons of BLAST, baseline paper (Waqaar et al. 2008) and our algorithms. The input of the sequence length is considered as the weight of the comparisons of this system.

The very interesting and effective observation of RSAM is that as the size increases, the performance of the RSAM increases. NNs and Markov process enables faster calculations in RSAM. Table 3 below is the comparison of the complexity of the three algorithms. As the size increases, the complexity of the BLAST and Baseline Paper algorithm increases. But RSAM is the same, though the sizes of the sequence increase.

From the figure above it is clearly visible that the high–low lines among BLAST, baseline paper and RSAM indicate the differences and faster response of RSAM algorithm compared with the other two. Receiver operating characteristics analysis for the data set according to the CPU utilization is shown Figs. 4, 5, 6.

Table 3 below shows the complexity comparison of BLAST, baseline paper (Waqaar et al. 2008) and RSAM. The BLAST is a linear method to check the local sequence and exon sequences. Linearity is good for simple and small size of the sequences and exons finding is straightforward. But it becomes difficult when the length of sequence and exon are not fixed. We consider the complexity with CPU utilization considering per-million nucleotides sequences. CPU utilization is the ration between user time and wall clock time. Besides the wall clock time, we have minimized the number of comparisons by incorporating the parallel comparisons (Akl 1985). Suppose k stands for the total number of comparison rounds (time) of any algorithms for n elements. Then the total comparisons are denoted by a function Total (k,n). So in view of complexity comparisons, Total (k,n) is the upper bound as worst case. On the contrary, minimum total number of comparisons Minimum (k,n) is lower bound as best case. In this work, we have significantly noticed that for maximum rounds of

Table 3 Complexity comparison of BLAST, baseline paper and this system

Iteration	Sequence length (Weight)	Complexity		
		BLAST	Baseline paper	RSAM
1	500,000	$O(n^2)$	$O(n)$	Ω ($n\log n$)
2	600,000	$O(n^2)$	$O(n)$	Ω ($n\log n$)
3	700,000	$O(n^2)$	$O(n)$	Ω ($n\log n$)
4	800,000	$O(n^2)$	$O(n)$	Ω ($n\log n$)
5	900,000	$O(n^2)$	$O(n)$	Ω ($n\log n$)
⋮	1,000,000	⋮	⋮	⋮
	1,100,000			
20	2,000,000	$O(n^2)$	$O(n)$	Ω ($n\log n$)

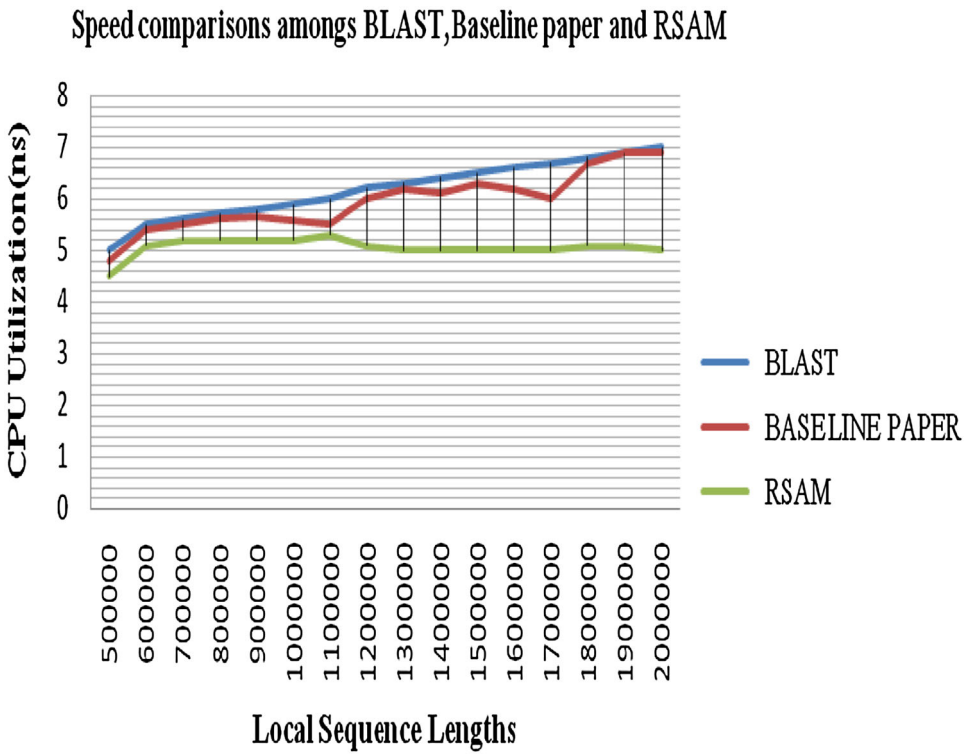

Fig. 4 The CPU utilization for the sequence length starting from the range of 500,000 to two millions nucleotides with the exons and introns in the sequences. We have clearly noticed that RSAMA performs better irrespective of any length and as the size of the nucleotides increases. It is interesting that as the lengths of the sequences increase RSAM performing better

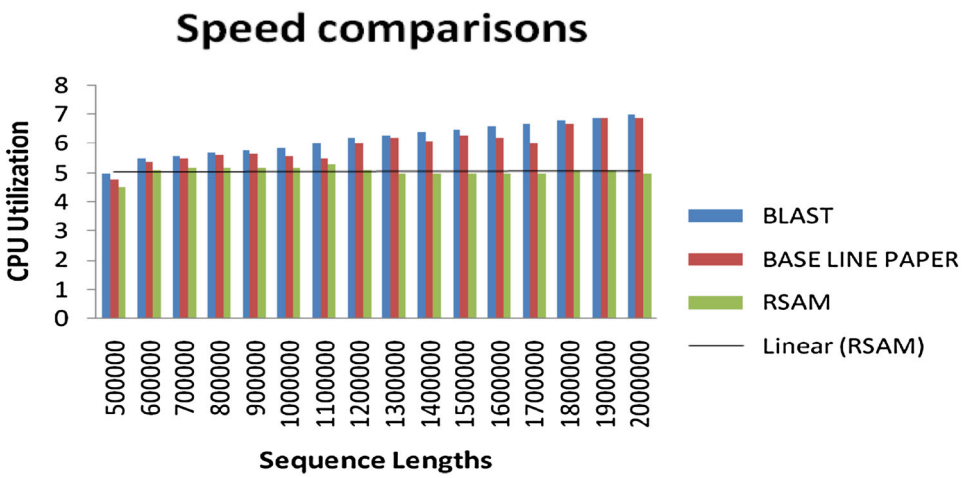

Fig. 5 ROCs based on checking the linearity at speed of RSAMA where the speed range is approximately 5 ns. The sequence starts from the length 500,000 to 2,000,000 bp

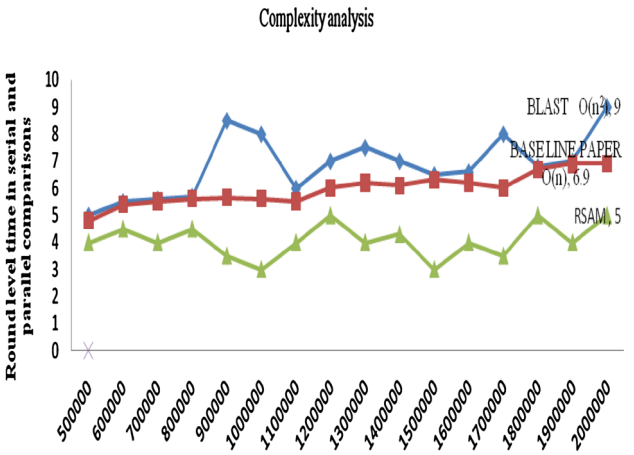

Fig. 6 The pictorial descriptions of the complexity analysis of the sequencing under the ROC environments and calculation. Here, we found that the complexity of this algorithm was Ω (*n*log*n*)

comparisons, the complexity is Ω (*n*log*n*). For sequence lengths from 500,000 to 2,000,000, the complexities among BLAST, baseline paper (Waqaar et al. 2008) and our algorithms are as follows.

In our baseline paper (Waqaar et al. 2008), there is no clue to compare the sensitivity of the sequencing and suffix tree is implemented in patterns matching and searching. We, on the other hand, measured the sensitivity by comparing the performance between suffix tree and suffix array. In view of this parameter, both suffix tree and array have the benefit in space and time values. Due to the fundamental benefits of suffix tree over suffix array, it performs better searching and matching. On the contrary, suffix array requires less space than suffix tree. For very long sequence length, suffix array is better than suffix tree.

7 Conclusion

In this work and activity, we have implemented and proposed the combined machine learning approach for sequence alignment and local sequencing. We have noticed that the combination of neural network, Markov chain, set operation, dynamic and IDP, bounding box algorithm for fixing the lengths of the local sequence make our revised sequence alignment matching (RSAM) faster and accurate compared with BLAST and baseline paper. We also successfully checked the performance between suffix tree and suffix array under sensitivity measurement. Bounding box algorithm is very interesting and newly imposed idea for maximum matching subsequence selections. One parameter we were unable to measure was the Specificity with Sensitivity in the ROCs analysis. In future, we will check this parameter.

References

Akl S (1985) Parallel sorting algorithms. Academic Press, USA

Alexandersson M, Cawley S, Pachter L (2003) SLAM: cross species gene finding and alignment with a generalized pair hidden Markov model. Genome Res 13:496–502

Altschul SF, Gish Miller W, Myers EW, Lipman DJ (1990) Basic local alignment search tool. J Mol Biol 215:403–410

Arratia R, Morris P, Waterman MS (1988) Stochastic scrabbles: a law of large numbers for sequence matching with scores. J Appl Probab 25:106–119

Batzoglou S, Pachter L, Mesirov JP, Berger B, Lander ES (2000a) Human and mouse gene structure: comparative analysis and application to exon prediction. Genome Res 10:950–958

Batzoglou S, Pachter L, Mesirov JP, Berger B, Lander ES (2000b) Human and mouse gene structure: comparative analysis and application to exon prediction. Genome Res 10:950–958

Bray N, Dubchak I, Pachter L (2003) Avid: a global alignment program. Genome Res 13:97–102

Claverie JM, Poirot O, Lopez F (1997) The difficulty of identifying genes in anonymous vertebrate sequences. Comput Chem 21:203–214

Delcher AL et al (2002) Fast algorithms for large-scale genome alignment and comparison. Nucleic Acids Res 30(11): 2478–2483

Delcher AL et al (2007) Identifying bacterial genes and endosymbiont DNA with Glimmer. Bioinformatics 23:673–679

Dembo A, Karlin S (1990) Strong limit theorems of empirical functional for large exceedances of partial sums of IID variables. Ann Probab 19(4):1737–1755

Dhar PK, Thwin ST, Tun K, Tsumoto Y, Maurer-Stroh S, Eisenhaber F, Surana U (2009) Synthesizing non-natural parts from natural genomic template. J Biol Eng 3:2

Doolittle RF (1996) Methods in enzymology, vol 266. Academic Press, San Diego

Ewing B, Green P (1998) Base-calling of automated sequencer traces using *phred*. II. Error probabilities. Genome Res 8:186–194

Fleischmann RD, Adams MD, White O, Clayton RA, Kirkness EF, Kerlavage AR (1995) Whole-genome random sequencing and assembly of Haemophilus influenza Rd. Science 269:496–512

Furey T, Kent WJ, Sugnet C, Roskin K, Pringle T, Zahler A, Haussler D (2002) The human genome browser at UCSC. Genome Res 12:996–1006

Karlin S, Altschul SF (1990) Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. Proc Natl Acad Sci USA 87:2264–2268

Karlin S, Altschul SF (1993) Applications and statistics for multiple high-scoring segments in molecular sequences. Proc Natl Acad Sci USA 90:5873–5877

Keller O, Kollmar M, Stanke M, Waack S (2011) A novel hybrid gene prediction method employing protein multiple sequence alignments. Bioinformatics 27:757–763

Kent WJ (2002) BLAT—the BLAST-like alignment tool. Genome Res 12:656–664

Khan MI, Kamal MS (2013a) RSAM: an integrated algorithm for local sequence alignment. Arch Des Sci 66(5):395–412 (ISSN 1661-464X)

Khan MI, Kamal MS (2013b) Sequencing ontology alignment for DNA annotation and damage identification. Eur J Sci Res 103(3):441–450

Lewis D (1992) An evaluation of phrasal and clustered representations on a text categorization task. In: Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval, ACM, vol 15, pp 37–50

Lewis D, Schapire R, Callan J, Papka R (1996) Training algorithms for linear text classifiers. In: Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval, ACM, pp 298–306

Lipman DJ, Pearson WR (1985) Rapid and sensitive protein similarity searches. Science 227:1435–1441

Lipman DJ, Pearson WR (1988) Improved tools for biological sequence comparison. Proc Natl Acad Sci USA 85:2444–2448

Needleman SB, Wunsch CD (1970) A general method applicable to the search for similarities in the amino acid sequence of two proteins. J Mol Biol 48:443–453

Ning Z, Cox AJ, Mullikin JC (2001) SSAHA: "A fast search method for large DNA databases". Genome Res 11:1725–1729

Pati A, Ivanova NN, Mikhailova N, Ovchinnikova G, Hooper SD, Lykidis A (2010) GenePRIMP: a gene prediction improvement pipeline for prokaryotic genomes. Nat Methods 7:455–457

Robert WF (2002) Molecular biology, 2nd edn. McGraw-Hill, New York, pp 7105–7107 (ISBN: 0-07-112287-7)

Ruiz M, Srinivasan P (1999) Hierarchical neural networks for text categorization (poster abstract). In: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval, ACM, pp 281–282

Schatz MC et al (2007) High-throughput sequence alignment using Graphics Processing Units. BMC Bioinform 8:474

Schwarz DS, Hutvagner G, Du T, Xu Z, Aronin N, Zamore PD (2003) Asymmetry in the assembly of the RNAi enzyme complex. Cell 115:199

Smith TF, Waterman MS (1981) Comparison of bio-sequences. Adv Appl Math 2:482–489

Stephens M, Sloan JS, Robertson PD, Scheet P, Nickerson DA (2006) Automating sequence-based detection and genotyping of SNPs from diploid samples. Nat Genet 38:375–381

Stormo GD, Schneider TD, Gold L, Ehrenfeucht A (1982) A use of the 'Perceptron' algorithm to distinguish translation initiation site in *E. coli*. Nucleic Acids Res. 10:2997–3011

Tech M, Meinicke P (2006) An unsupervised classification scheme for improving predictions of prokaryotic TIS. BMC Bioinformatics 7:121

van Baren MJ, Brent MR (2006) Iterative gene prediction and pseudo gene removal improves genome annotation. Genome Res 16:678–685

Waqaar H, Alex A, Bharath R (2008) An efficient algorithm for local sequence alignment. In: 30th Annual international IEEE EMBS conference vancouver, British Columbia, Canada, August 20–24, 2008

Watanabe T, Takeda A, Mise K, Okuno T, Suzuki T, Minami N, Imai H (2005) Stage-specific expression of microRNAs during Xenopus development. FEBS Lett 579:318

Waterman MS (1989) Mathematical methods for DNA sequences. CRC Press, Boca Raton

Waterman MS (1994) Introduction to computational biology. Chapman & Hall, London

Weckx S, Del-Favero J, Rademakers R, Claes L, Cruts M, De Jonghe P, Van Broeckhoven C, De Rijk P (2005) novoSNP, a novel computational tool for sequence variation discovery. Genome Res 15:436–442

Wu WS et al (2006) Computational reconstruction of transcriptional regulatory modules of the yeast cell cycle. BMC Bioinform 7:421

Yetisgen-Yildiz M, Pratt W (2005) The effect of feature representation on Medline document classification. In AMIA Annual Symposium Proceedings. American Medical Informatics Association, vol 23, p 849

Yok NG, Rosen GL (2011) Combining gene prediction methods to improve meta genomic gene annotation. BMC Bioinform 12:20

Yu GX, Snyder EE, Boyle SM, Crasta OR, Czar M, Mane SP et al (2007) A versatile computational pipeline for bacterial genome annotation improvement and comparative analysis, with Brucella as a use case. Nucleic Acids Res 35:3953–3962

Zhang J, Wheeler DA, Yakub I, Wei S, Sood R, Rowe W, Liu PP, Gibbs RA, Buetow KH (2005) SNP detector: a software tool for sensitive and accurate SNP detection. PLoS Comput Biol 1(5):e53

Zhu HQ, Hu GQ, Ouyang ZQ, Wang J, She ZS (2004) Accuracy improvement for identifying translation initiation sites in microbial genomes. Bioinformatics 20:3308–3317

Zhu HQ, Hu GQ, Yang YF, Wang J, She ZS (2007) MED: a new non-supervised gene prediction algorithm for bacterial and archaeal genomes". BMC Bioinform 8:97