International Journal of Scientific Research in Computer Science and Engineering **Research Paper**
Vol.5, Issue.4, pp.79-87, August (2017) **E-ISSN:** 2320-7639

Effectiveness of *Ssaha* Algorithm For Searching Motif In Large Databases of Dna Sequences

Khumukcham Robindro[1*], Ashoke Das[2]

1*Department of Computer Science, Manipur University (A Central University), Manipur, India
[2]Department of Mathematics, Raiganj University, Raiganj, India

[*]Corresponding Author: rbkh@manipuruniv.ac.in, Tel: +91-94850-44453

Available online at: www.isroset.org

Received 10th Jun 2017, Revised 16th Jul 2017, Accepted 12th Aug 2017, Online 30th Aug 2017

***Abstract*—** Motif finding has become a very significant area of study in the post genomic era because of its various applications and usage in the bioinformatics research. The volume of the biological data is also on the ever increasing trend. Many algorithms have been proposed for efficient motif findings. In this paper, an approach is proposed to use a fast search method called SSAHA for motif finding. SSAHA is an efficient algorithm which effectively searches a given query in large databases containing multiple gigabases of DNA. In our approach, the effectiveness of SSAHA algorithm in searching large databases of DNA is exploited for searching motif in large databases of DNA sequences.

Keywords*— *Motif, Algorithm, SSAHA, DNA, Database

I. INTRODUCTION

Efficient finding of motifs in a given DNA sequences is a crucial problem today pertaining to the ever increasing volume of biological data and the huge applications it has in bioinformatics research[11]. But finding of motifs poses several complications, namely, we don't know the motif sequence, or where it may be located in the DNA sequence relative to the genes start, or they can slightly differ from gene to gene, etc. So, the problems of motif finding manifest itself in two forms. We will be attempting to use this already existing efficient search algorithm SSAHA for the first of the two forms manifested by incorporating slight changes into the algorithm wherever necessary. For the second form, since the motif to be searched is not known, a straightforward approach cannot be undertaken.

II. MOTIF

A motif is a nucleotide or amino acid sequence pattern which is having biological significance. Motifs occur repeatedly in same molecule or in many molecules. They often indicate sequence specific binding sites for proteins such as nucleases and transcription factors. By searching such patterns (motifs) in a gene, functionally important regions of genome can be recognized [5]. So they are important for locating binding sites, regulatory signals, for controlling gene expression and also for identification of potential drug sites, mRNA processing (splicing, editing, polyadenylation) and transcription termination[3],[8]. The growing usefulness of the motifs in deciphering the regulatory program of individual gene and in defining genetic regulatory networks and the abundance of computationally and experimentally derived sequence motifs make them very important tools for post-genomic ear computational biology. In existing literatures, three versions of motif search problems are identified. Planted (l,d)- motif problem is one of these three versions[12]. An (l,d) -planted motif problem is defined as: for n DNA sequences given, each of length L, find *motifs* of length l with at most d allowable mismatches in each of the given DNA sequences. Numerous algorithms on planted motif search have been proposed by many authors10]. Some of the authors are Bailey and Elkan, Lawrence et al., Pevzner and Sze, Rocke and Tompa, Buhler and Tompa, etc. Algorithms for planted motif finding, based on the basic approached employed, can be categorized into two, namely, profile-based and pattern-based algorithms [13]. The prediction of the starting positions of the occurrences of the motif in each sequence is done by profile based algorithms. And prediction of the motif itself is done by pattern-based algorithms. Some examples of pattern based algorithms are PROJECTION, MULTIPROFILER, MITRA and Pattern Branching, etc. And some examples of pattern based algorithms are CONSENSUS, MEME, Profile Branching, etc. MITRA algorithm is a pattern based algorithm and it is exact.

Different approaches are tried in motif finding [14]. There is the Brute Force Motif Finding approach, Branch and Bound Motif Search, Branch and Bound Median String Search, Consensus and Pattern Branching-Greedy Motif Search [9].

The complications in finding motifs are:

- We may not know the motif sequence
- We do not know the position of the motif with respect to the start of the gene.
- Also, motifs in different genes can differ slightly.

As already mentioned above, there are several complications in trying to identify motifs in a DNA sequence [10]. Based on these complications, the problem of motif finding manifests itself in two forms. The two forms are:

A. Known Motif and Unknown Position

Here the motif is known. And the problem is to find the position of a given motif of length *l* with up to d mismatches in a protein sequence. This is the less complex one of the two forms.

B. Unknown Motif and Unknown Position

In this form, both the motif and the location of the motif are not known. This form poses lots of complications as only the length ‘l’ and number of allowable mismatches ‘d’ is given. There is no any other information of the sequence to be found.

C. Motif and Consensus Sequence

Motifs tend to be different in different sequences up to a certain hamming distance. A motif can be represented by the notation like [XYZ] which means X or Y or Z. However, [XYZ] doesn’t indicate the likelihood of any particular match. Hence a motif can be associated with two or more patterns- the defining pattern and several other typical patterns. As an example, we can consider the defining sequence for the IQ motif. It may be taken as

[FILV]Qxxx[RK]Gxxx[RK]xx[FILVWY]

Here, x signifies any amino acid and the square bracket indicates alternatives that can be chosen.

III. A Brief on SSAHA

SSAHA is fast search method used for searching large databases [1]. SSAHA stands for Sequence Search and Alignment by Hashing Algorithm. The SSAHA algorithm exploits the fact that nowadays, systems with sufficient RAM which can easily store hash table used for describing databases containing multiple gigabases of DNA are available to us. The use of hash table enables the fast search on the database. In terms of speed, memory usage and sensitivity, SSAHA is better than other search algorithms like BLAST, FASTA etc.

As mentioned above, SSAHA makes use of hash table for performing fast search on multiple gigabases of DNA. Keeping into mind that machines with sufficient RAM are available for storing a hash table describing database containing multiple gigabases of DNA, SSAHA is a good search method. As far as notations and definitions are concerned, a query sequence is represented by Q, the DNA sequences in the database are represented by D = {$S_{1,} S_{2,\dots}$}. An index *i* identifies each sequence in the database. In SSAHA, a k-tuple which is a contiguous sequence of DNA bases which is k bases long is used as an important part of the searching procedure. For motif searching in the DNA sequence, we will be considering k=1 since we will be desiring to use the property of this search method which allows mismatches in the sequence searched to address the issue of allowable mismatches in motif searching[2]. The letter *j* is used to represent the offset where offset of a k-tuple of S is the position of the first base of the tuple with respect to the first base of S. Then $W_j(S)$ is used to denote the tuple of S with offset *j*. The construction of the hash table which is used to represent the database is the first step of the algorithm. The hash table consists of the list of possible k-tuples along with their position of occurrence in the various DNA sequences of the database. The occurrences are described by an (i,j) pair where i is the index which is an integer used to refer to each DNA sequence S_i in the database and j is the offset, that is the position of the k-tuple from the start. The DNA sequences are scanned for the k-tuples and their positions are stored in the hash table. Now the hash table can be further used for the search procedure. The next step of the algorithm is the sequence search. For this, we proceed along Q base-by-base from base 0 to base n-k where n is the length of the query sequence Q. At base t, the k-tuple is represented by $W_t(Q)$. Then, we search for the r occurrences of the tuples. From the list of these positions, we compute the list of hits. The computation of the list of hits will be described as we proceed. Then the list of hits is sorted, first based on the index value then, based on the shift value for the same index and based on the offset value for the same index and shift. Then we scanned the sorted list for runs of hit which suits our search.

IV. Proposed Approach for Motif

For the two forms, the approaches to be used will have to be different. Since the proposal is to use SSAHA for the purpose of finding motif in DNA sequences, we will explore the possibilities in which the search algorithm can be applied. It is known to us that SSAHA searches a sequence provided to it in the DNA sequences stored in the database. So, for the first form of motif finding discussed above, it is very well applicable as the motif to be searched is known to us beforehand. We need to find the start location of the motifs in the sequences in the database. SSAHA can well perform this action. Also, SSAHA provides provision for searching different variations of the provided search string. This property can be exploited for countering the ‘d’ mismatches the motif can have. For the second form, we need to think of a plan as SSAHA cannot do anything if the search sequence is not known.

A. Known Motif and Unknown Position

For known motif sequence, there is a given motif which needs to be searched in the DNA sequences which we have in the database. SSAHA performs this task well.

Algorithm involving SSAHA that is being used to search the given motif:

Procedure Known MotifSearch ()

1. Set k=1 (the tuple).

2. Input the Motif to be searched, the length of the motif 'n' and number of allowable mismatches''.

3. Using the DNA sequence available in the database, construct a k-tuple hash table for the DNA sequence. (This hash table will form the basis of the search. It consists of a column for tuple and corresponding positions of occurrence of the tuple.)

3.1. The position part of the table hash elements comprising of an index, i and an offset, j i.e. (i, j).

4. Using the hash table constructed above, prepare a list of matches for the motif sequence. The list consists of five columns, base t, tuple wt(Q), positions, list of hits H, the sorted list of hits M. The list of hits H is calculated and sorted to give M as follows.

a. The 'r' positions of occurrence of the k-tuple wt(Q) is obtained at base t i.e. (i_1,j_1), (i_2,j_2), (i_3,j_3), (i_4,j_4), (i_5,j_5)…. (i_r,j_r).

b. From the above obtained list, the list of hits H is calculated as

H_k= (i_k,j_k-t, j_k) where j_k-t is the shift.

4.3. Sort the list H based on the index first.

4.4. Again, sort the runs of hits with same index based on the shift.

4.5. For those hits with same value of shift, sort again based on the offset of the hits.

5. Use the sorted list of hits for searching the motif.

5.1. Scan through the sorted list of hits to find a run of hits for which the difference of the position of the start of the run and end of the run is greater than or equal to (n-d-1) and difference of offset of end of the run and shift of start of the run is n-1.

5.2. The shift of the start of the run through the offset of the end of the run gives the position of the motif in the sequence.

Consider a DNA sequence in the database (here we have considered a single sequence of DNA). Let the motif ***acgtactt*** be in the DNA sequence. Now, we will search for the motif in the DNA sequence using algorithm presented above (SSAHA).

The DNA sequence:

S1:
cctgatagacgctatctggctatccacgtacttaggtcctctgtgcgaatctatg cgtttccaaccagtactggtgtacatttgatacgtacttacaccggcaacctgaa acaaacgctcagaaccagaagt

We prepare the hash table for the DNA sequence first.

TABLE1: 1-TABLE HASH TABLE FOR THE GIVEN DNA SEQUENCE S1

w	**Positions**															
A	1,5	1,7	1,9	1,14	1,22	1,26	1,30	1,34	1,48	1,49	1,53	1,63	1,67	1,70	1,78	1,80
	1,85	1,87	1,91	1,95	1,97	1,103	1,104	1,109	1,110	1,111	1,113	1,114	1,115	1,121	1,123	1,124
	1,127	1,129	1,130	1,64												
C	1,1	1,2	1,10	1,12	1,16	1,20	1,24	1,25	1,27	1,31	1,38	1,39	1,41	1,46	1,51	1,56
	1,61	1,62	1,65	1,66	1,71	1,79	1,88	1,92	1,96	1,98	1,99	1,102	1,105	1,106	1,112	1,116
	1,118	1,120	1,125	1,126												
G	1,4	1,8	1,11	1,18	1,19	1,28	1,35	1,36	1,43	1,45	1,47	1,55	1,57	1,68	1,73	1,74
	1,76	1,84	1,89	1,100	1,101	1,108	1,117	1,122	1,128	1,131						
T	1,3	1,6	1,13	1,15	1,17	1,21	1,23	1,29	1,32	1,33	1,37	1,40	1,42	1,44	1,50	1,52
	1,54	1,58	1,59	1,60	1,69	1,72	1,75	1,77	1,81	1,82	1,83	1,86	1,90	1,93	1,94	1,107
	1,119	1,132														

This hash table is calculated only once. After that, it can be stored and reused again and again for different uses.

TABLE2: LIST OF MATCHES FOR THE MOTIF

T	$W_t(Q)$	Positions	H(index,shift,offset)	M
0	a	1,5	1,5,5	1,-4,1
		1,7	1,7,7	1,-4,3
		1,9	1,9,9	1,-3,2
		1,14	1,14,14	1,-3,3
		1,22	1,22,22	1,-1,6
		1,26	1,26,26	1,0,1
		1,30	1,30,30	1,0,2
		1,34	1,34,34	1,0,6
		1,48	1,48,48	1,1,2
		1,49	1,49,49	1,1,5
		1,53	1,53,53	1,2,4
		1,63	1,63,63	1,3,6
		1,64	1,64,64	1,3,7
		1,67	1,67,67	1,5,5
		1,70	1,70,70	1,5,9
		1,78	1,78,78	1,5,10
		1,80	1,80,80	1,6,8
		1,85	1,85,85	1,6,13
		1,87	1,87,87	1,7,7
		1,91	1,91,91	1,7,12
		1,95	1,95,95	1,7,13
		1,97	1,97,97	1,8,15
		1,103	1,103,103	1,9,9
		1,104	1,104,104	1,9,10
		1,109	1,109,109	1,9,11
		1,110	1,110,110	1,9,15
		1,111	1,111,111	1,10,13
		1,113	1,113,113	1,10,14
		1,114	1,114,114	1,10,17
		1,115	1,115,115	1,11,12
		1,121	1,121,121	1,11,16
		1,123	1,123,123	1,11,17
		1,124	1,124,124	1,12,15
		1,127	1,127,127	1,14,14
		1,129	1,129,129	1,14,17
		1,130	1,130,130	1,14,21
1	c	1,1	1,0,1	1,15,16
		1,2	1,1,2	1,15,20
		1,10	1,9,10	1,15,21
		1,12	1,11,12	1,16,18
		1,16	1,15,16	1,16,23
		1,20	1,19,20	1,17,19
		1,24	1,23,24	1,17,23
		1,25	1,24,25	1,18,21
		1,27	1,26,27	1,18,22
		1,31	1,30,31	1,19,20
		1,38	1,37,38	1,19,24
		1,39	1,38,39	1,20,23
		1,41	1,40,41	1,20,25
		1,46	1,45,46	1,22,22
		1,51	1,50,51	1,22,26
		1,56	1,55,56	1,22,27
		1,61	1,60,61	1,22,29
		1,62	1,61,62	1,23,24
		1,65	1,64,65	1,23,29
		1,66	1,65,66	1,24,25
		1,71	1,70,71	1,25,32
		1,79	1,78,79	1,26,26
		1,88	1,87,88	1,26,27
		1,92	1,91,92	1,26,28
		1,96	1,95,96	1,26,29
		1,98	1,97,98	1,26,30
		1,99	1,98,99	1,26,31
		1,102	1,101,102	1,26,32
		1,105	1,104,105	1,26,33
		1,106	1,105,106	1,27,33
		1,112	1,111,112	1,29,32
		1,116	1,115,116	1,30,30
		1,118	1,117,118	1,30,31
		1,120	1,119,120	1,30,33
		1,125	1,124,125	1,30,34
		1,126	1,125,126	1,30,37
2	g	1,4	1,2,4	1,31,37
		1,8	1,6,8	1,33,35
		1,11	1,9,11	1,33,38
		1,18	1,16,18	1,33,40
		1,19	1,17,19	1,34,34
		1,28	1,26,28	1,34,36
		1,35	1,33,35	1,34,37
		1,36	1,34,36	1,34,39
		1,43	1,41,43	1,34,40
		1,45	1,43,45	1,35,42
		1,47	1,45,47	1,36,41
		1,55	1,53,55	1,36,42
		1,57	1,55,57	1,37,38
		1,68	1,66,68	1,37,40
		1,73	1,71,73	1,37,44
		1,74	1,72,74	1,38,39
		1,76	1,74,76	1,38,44
		1,84	1,82,84	1,39,42
		1,89	1,87,89	1,40,41
		1,100	1,98,100	1,41,43
		1,101	1,99,101	1,41,44
		1,108	1,106,108	1,41,46
		1,117	1,115,117	1,43,50
		1,122	1,120,122	1,43,53
		1,128	1,121,128	1,44,48

		1,131	1,129,131	1,44,50
3	t	1,3	1,0,3	1,45,46
		1,6	1,3,6	1,45,47
		1,13	1,10,13	1,45,49
		1,15	1,12,15	1,45,52
		1,17	1,14,17	1,46,51
		1,21	1,18,21	1,46,52
		1,23	1,20,23	1,47,50
		1,29	1,26,29	1,47,54
		1,32	1,29,32	1,48,48
		1,33	1,30,33	1,48,54
		1,37	1,34,37	1,49,49
		1,40	1,37,40	1,49,52
		1,42	1,39,42	1,49,53
		1,44	1,41,44	1,50,51
		1,50	1,47,50	1,51,54
		1,52	1,49,52	1,51,58
		1,54	1,51,54	1,52,58
		1,58	1,55,58	1,52,59
		1,59	1,56,59	1,53,53
		1,60	1,57,60	1,53,55
		1,69	1,66,69	1,53,59
		1,72	1,69,72	1,53,60
		1,75	1,72,75	1,54,60
		1,77	1,74,77	1,55,56
		1,81	1,78,81	1,55,57
		1,82	1,79,82	1,55,58
		1,83	1,80,83	1,55,59
		1,86	1,83,86	1,56,61
		1,90	1,87,90	1,57,60
		1,93	1,90,93	1,57,62
		1,94	1,91,94	1,59,63
		1,107	1,104,107	1,60,61
		1,119	1,116,119	1,60,64
		1,132	1,129,132	1,60,65
4	a	1,5	1,1,5	1,61,62
		1,7	1,3,7	1,61,66
		1,9	1,5,9	1,62,69
		1,14	1,10,14	1,63,63
		1,22	1,18,22	1,63,67
		1,26	1,22,26	1,63,69
		1,30	1,26,30	1,64,64
		1,34	1,30,34	1,64,65
		1,48	1,44,48	1,65,66
		1,49	1,45,49	1,65,72
		1,53	1,49,53	1,66,69
		1,63	1,59,63	1,66,70
		1,64	1,60,64	1,66,71
		1,67	1,63,67	1,66,72
		1,70	1,66,70	1,66,78
		1,78	1,74,78	1,67,67
		1,80	1,76,80	1,68,75
		1,85	1,81,85	1,69,72
		1,87	1,83,87	1,69,75
		1,91	1,87,91	1,70,70
		1,95	1,91,95	1,70,71
		1,97	1,93,97	1,70,77
		1,103	1,99,103	1,71,73
		1,104	1,100,104	1,71,77
		1,109	1,105,109	1,72,74
		1,110	1,106,110	1,72,75
		1,111	1,107,111	1,74,76
		1,113	1,109,113	1,74,77
		1,114	1,110,114	1,74,78
		1,115	1,111,114	1,74,79
		1,121	1,117,121	1,74,81
		1,123	1,119,123	1,75,81
		1,124	1,120,124	1,75,82
		1,127	1,123,127	1,76,80
		1,129	1,125,129	1,76,82
		1,130	1,126,130	1,76,83
5	c	1,1	1,-4,1	1,77,83
		1,2	1,-3,2	1,78,78
		1,10	1,5,10	1,78,79
		1,12	1,7,12	1,78,81
		1,16	1,11,16	1,79,82
		1,20	1,15,20	1,79,86
		1,24	1,19,24	1,80,80
		1,25	1,20,25	1,80,83
		1,27	1,22,27	1,80,86
		1,31	1,26,31	1,81,85
		1,38	1,33,38	1,82,84
		1,39	1,34,39	1,83,86
		1,41	1,36,41	1,83,87
		1,46	1,41,46	1,83,88
		1,51	1,46,51	1,83,90
		1,56	1,51,56	1,84,90
		1,61	1,56,61	1,85,85
		1,62	1,57,62	1,86,93
		1,65	1,60,65	1,87,87
		1,66	1,61,66	1,87,88
		1,71	1,66,71	1,87,89
		1,79	1,74,79	1,87,90
		1,88	1,83,88	1,87,91
		1,92	1,87,92	1,87,92
		1,96	1,91,96	1,87,93
		1,98	1,93,98	1,87,94
		1,99	1,94,99	1,88,94
		1,102	1,97,102	1,90,93
		1,105	1,100,105	1,91,91

		1,106	1,101,106	1,91,92
		1,112	1,107,112	1,91,94
		1,116	1,111,116	1,91,95
		1,118	1,113,118	1,93,97
		1,120	1,115,120	1,93,98
		1,125	1,120,125	1,94,99
		1,126	1,121,126	1,95,95
6	t	1,3	1,-3,3	1,95,96
		1,6	1,0,6	1,97,97
		1,13	1,7,13	1,97,98
		1,15	1,9,15	1,97,102
		1,17	1,11,17	1,98,99
		1,21	1,15,21	1,98,100
		1,23	1,17,23	1,99,101
		1,29	1,23,29	1,99,103
		1,32	1,26,32	1,100,104
		1,33	1,27,33	1,100,105
		1,37	1,31,37	1,100,107
		1,40	1,34,40	1,101,102
		1,42	1,36,42	1,101,106
		1,44	1,38,44	1,101,107
		1,50	1,44,50	1,102,102
		1,52	1,46,52	1,102,103
		1,54	1,48,54	1,103,103
		1,58	1,52,58	1,104,104
		1,59	1,53,59	1,104,105
		1,60	1,54,60	1,104,107
		1,69	1,63,69	1,105,106
		1,72	1,66,72	1,105,109
		1,75	1,69,75	1,106,108
		1,77	1,71,77	1,106,110
		1,81	1,75,81	1,107,111
		1,82	1,76,82	1,107,112
		1,83	1,77,83	1,109,109
		1,86	1,80,86	1,109,113
		1,90	1,84,90	1,110,110
		1,93	1,87,93	1,110,114
		1,94	1,88,94	1,111,111
		1,107	1,101,107	1,111,112
		1,119	1,113,119	1,111,115
		1,132	1,126,132	1,111,116
7	t	1,3	1,-4,3	1,112,119
		1,6	1,-1,6	1,113,113
		1,13	1,6,13	1,113,118
		1,15	1,8,15	1,113,119
		1,17	1,10,17	1,114,114
		1,21	1,14,21	1,115,115
		1,23	1,16,23	1,115,116
		1,29	1,24,29	1,115,117
		1,32	1,25,32	1,115,120
		1,33	1,26,33	1,116,119
		1,37	1,30,37	1,117,118
		1,40	1,33,40	1,117,121
		1,42	1,35,42	1,119,120
		1,44	1,37,44	1,119,123
		1,50	1,43,50	1,120,122
		1,52	1,45,52	1,120,124
		1,54	1,47,54	1,120,125
		1,58	1,51,58	1,121,121
		1,59	1,52,59	1,121,126
		1,60	1,53,60	1,123,123
		1,69	1,62,69	1,123,127
		1,72	1,65,72	1,124,124
		1,75	1,68,75	1,124,125
		1,77	1,70,77	1,125,126
		1,81	1,74,81	1,125,129
		1,82	1,75,82	1,125,132
		1,83	1,76,83	1,126,128
		1,86	1,79,86	1,126,130
		1,90	1,83,90	1,126,132
		1,93	1,86,93	1,127,127
		1,94	1,87,94	1,129,129
		1,107	1,100,107	1,129,131
		1,119	1,112,119	1,129,132
		1,132	1,125,132	1,130,130

The run of hits in the sorted list of hits which has been colored red gives the positions of the motif in the sequence. The number of allowable mismatches that we wished should be less than length of the motif-1.

The Search Procedure

We will be giving a detailed description of the search procedure that we have mentioned and used above. The procedure takes as input, the motif to be searched, Q, and the length of the motif, n and the number of allowable mismatches in the search for the motif, d. Each sequence in the database is identified by its index which starts from 1. The location of each tuple in a sequence is identified by offset.

The position of a tuple as a whole is identified by the index-offset pair. The position is represented as (i, j) where 'i' is the index and 'j' is the offset. The k-tuple hash table is constituted by the tuple in one column and the corresponding positions in the rows of the table corresponding to each tuples.

After the hash table for the sequence has been prepared, it is used to prepare the list of matches for the motif. This table will consist of base t, tuple wt(Q), positions (the same as in the first hash table), the list of hits, H and the sorted list of hits ,M. The list of hits is prepared from the r positions of occurrence of the each tuples as

$H_k=(i_k, j_k\text{-}t, j_k)$ where 1<=k<=r

The list consists of all the hits corresponding to all the tuples in the motif to be searched [7]. After the list of hits is calculated, the list is sorted. First, sorting on the list is performed based on the index value of the hits. Then, for each hits with the same value of index, sorting is again performed based on the shift value of the hits. Then, for each hits with the same value of shift, sorting is again performed based on the offset of the hits. After these three rounds of sorting, the list of hits has been properly sorted. For the second and third round sorting, we scan for runs of hits with same index or hits with same shift value and call the sort procedure for the run.

After the list of matches has been prepared, the last step of finding the motif remains. On the sorted list of hits M, we perform scan for runs of hits satisfying the conditions stated below.

1. End position of run in the list – start position of run in the list >=n-d-1

2. Offset value of the end position of the run in the list-shift value of the start position of the run in the list is equal to n-1.

The run of hits which satisfies the above stated two conditions give the position of the motif in the sequence. The shift value of the start of the run in the list gives the starting position of the motif in the sequence and the offset value of the end position of the run gives the ending position of the motif in the sequence.

V. RESULTS

We implement the search procedure described above in simple C language codes. We run the implementation for 4 DNA sequences of varying lengths having the motif with different degrees of mismatches. We performed the search by providing different values of 'd'.

The sequences we used looks like this.

Sequence-1:
cctgatagacgctatctggctatccacgtacttaggtcctctgtgcgaatctatgcgtttccaaccagtactggtgtacatttgat

Sequence-2:
acgctatctggctatccacctacttaggtcctctgtgcgaatctatgcgtttccaaccagtactggtgtacatttgatccgtac

Sequence-3:
tgaaacaaacgctcagaaccagaagtcctgatagacgctatctggctatccacctacataggtcctctgtgcgaatctatgcgtttccaaccag

Sequence-4:
gcaacctgaaacaaacgctcagaaccagaagtcctgatagacgctatctggctatccagctacctaggtcctctgtgcgaatctatgcgtttccaaccagtac

The table-3 gives the result of running the implementation for different values of 'd' on the DNA sequences given above.

TABLE3: RESULTS TABLE

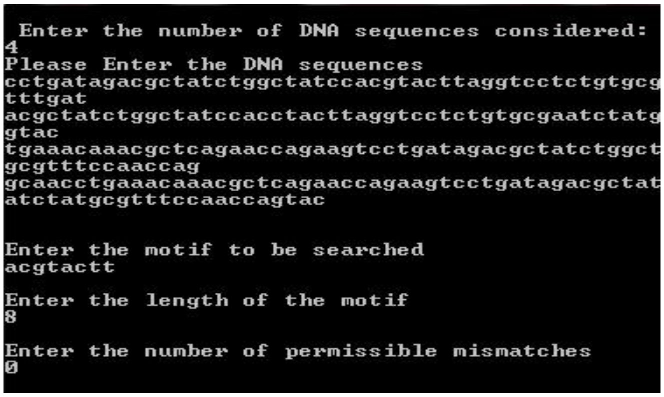
Motif	D	Positions of Motif
acgtactt	0	Sequence-1: 26---→33
acgtactt	1	Sequence-1:26----→33 Sequence-2:18----→25
acgtactt	2	Sequence-1:26----→33 Sequence-2:18----→25 Sequence-3:52----→59
acgtactt	3	Sequence-1:26----→33 Sequence-1:30----→37 Sequence-1:74----→81 Sequence-2:18----→25 Sequence-2:22----→29 Sequence-2:66----→73 Sequence-3:52----→59 Sequence-3:56----→63 Sequence-4:59----→66 Sequence-4:63----→70

We run the implementation for only up to d=3 starting form d=0. For each value of allowable mismatches, the algorithm correctly finds the locations of the motif in each of the sequences.

Snapshots of the implementation:

For d=0:

Input:



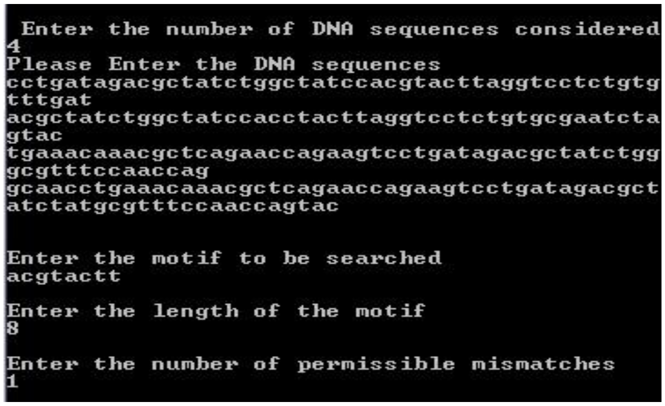
```
 Enter the number of DNA sequences considered:
4
Please Enter the DNA sequences
cctgatagacgctatctggctatccacgtacttaggtcctctgtgcg
tttgat
acgctatctggctatccacctacttaggtcctctgtgcgaatctatg
gtac
tgaaacaaacgctcagaaccagaagtcctgatagacgctatctggct
gcgtttccaaccag
gcaacctgaaacaaacgctcagaaccagaagtcctgatagacgctat
atctatgcgtttccaaccagtac


Enter the motif to be searched
acgtactt

Enter the length of the motif
8

Enter the number of permissible mismatches
0
```

Output:

```
The positions of the motif are:
```

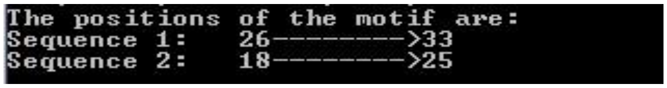
For d=1:

Input:



```
 Enter the number of DNA sequences considered
4
Please Enter the DNA sequences
cctgatagacgctatctggctatccacgtacttaggtcctctgtg
tttgat
acgctatctggctatccacctacttaggtcctctgtgcgaatcta
gtac
tgaaacaaacgctcagaaccagaagtcctgatagacgctatctgg
gcgtttccaaccag
gcaacctgaaacaaacgctcagaaccagaagtcctgatagacgct
atctatgcgtttccaaccagtac


Enter the motif to be searched
acgtactt

Enter the length of the motif
8

Enter the number of permissible mismatches
1
```

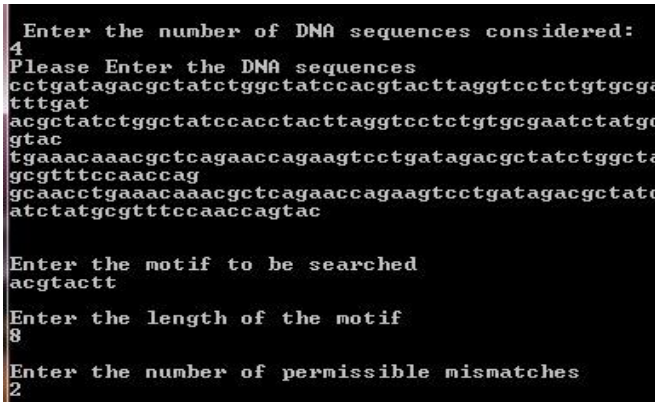
Output:



```
The positions of the motif are:
Sequence 1:   26--------->33
Sequence 2:   18--------->25
```

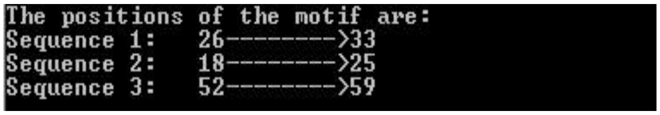
For d=2:

Input:



```
 Enter the number of DNA sequences considered:
4
Please Enter the DNA sequences
cctgatagacgctatctggctatccacgtacttaggtcctctgtgcga
tttgat
acgctatctggctatccacctacttaggtcctctgtgcgaatctatgc
gtac
tgaaacaaacgctcagaaccagaagtcctgatagacgctatctggcta
gcgtttccaaccag
gcaacctgaaacaaacgctcagaaccagaagtcctgatagacgctatc
atctatgcgtttccaaccagtac


Enter the motif to be searched
acgtactt

Enter the length of the motif
8

Enter the number of permissible mismatches
2
```

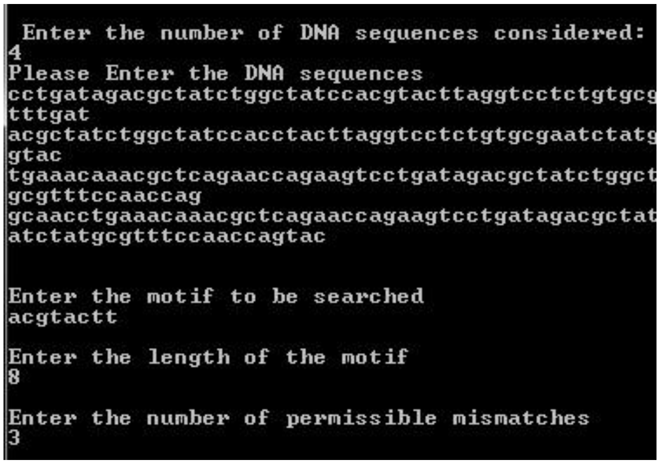
Output:



```
The positions of the motif are:
Sequence 1:   26-------->33
Sequence 2:   18-------->25
Sequence 3:   52-------->59
```

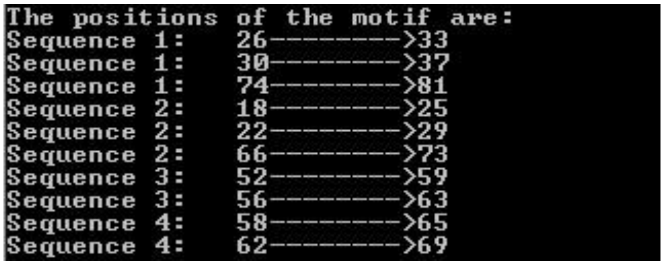
For d=3:

Input:



```
 Enter the number of DNA sequences considered:
4
Please Enter the DNA sequences
cctgatagacgctatctggctatccacgtacttaggtcctctgtgcg
tttgat
acgctatctggctatccacctacttaggtcctctgtgcgaatctatg
gtac
tgaaacaaacgctcagaaccagaagtcctgatagacgctatctggct
gcgtttccaaccag
gcaacctgaaacaaacgctcagaaccagaagtcctgatagacgctat
atctatgcgtttccaaccagtac


Enter the motif to be searched
acgtactt

Enter the length of the motif
8

Enter the number of permissible mismatches
3
```

Output:



```
The positions of the motif are:
Sequence 1:   26--------->33
Sequence 1:   30--------->37
Sequence 1:   74--------->81
Sequence 2:   18--------->25
Sequence 2:   22--------->29
Sequence 2:   66--------->73
Sequence 3:   52--------->59
Sequence 3:   56--------->63
Sequence 4:   58--------->65
Sequence 4:   62--------->69
```

VI. CONCLUSION

In this post genomic era, the volume of biological data is exploding and efficient methods are required for utilizing these ever increasing data so that it may not get wasted. Motif finding, also, pertaining to its many applications and uses in bioinformatics research has become a topic of high significance. SSAHA is an efficient and good search algorithm used for searching large databases. Computational results of SSAHA show that it is three or four orders of magnitude faster than other search algorithms like BLAST and FASTA. By making use of this efficient search algorithm which effectively searches large databases, our proposed method of motif finding has successfully utilized the ever increasing biological data for finding the sequences of high significance i.e. motifs efficiently. In future, there is scope for furthering the work towards using SSAHA for finding the motif for the case of unknown motif and unknown position form. Works can be done in this direction on how to bring in this efficient search algorithm in this form of motif searching.

REFERENCES

[1] Zemin Ning, Anthony J. Cox, and James C. Mullikin (2001) *SSAHA: A First Search Method for Large DNA Databases*

[2] Altschul, S.F., Gish, W., Miller, W., Myers, E.W., and Lipman, D.J.(1990). *Basic Local Alignment Search Tool. J. Mol. Biol.215: 403–410.*

[3] Altschul, S.F., Madden, T.L., Schäffer, A.A., Zhang, J., Zhang, Z.,Miller, W., and Lipman, D.J. (1997). *Gapped BLAST and PSI-BLAST: A new generation of protein database search programs.Nucleic Acids Res. 25: 3389–3402.*

[4] Benson, G. (1999). *Tandem Repeats Finder: A program to analyzeDNA sequences. Nucleic Acids Res. 27: 573–580.*

[5] Delcher, A.L., Kasif, S., Fletschmann, R.D., Peterson, J., White, O.,and Salzberg, S. (1999). *Alignment of whole genomes. NucleicAcids Res. 27: 2369–2376.*

[6] Gusfield, D. (1997). *Algorithms on strings, trees and sequences: Computer science and computational biology. Cambridge University Press,*Cambridge, UK.

[7] Knuth, D.E. (1998). *The art of computer programming vol. 3: Sorting and searching. Addison-Wesley, Reading, MA.*

[8] Lipman, D.J. and Pearson, W.R. (1985). *Rapid and sensitive protein similarity searches. Science 227: 1435–1441.*

[9] Zhang, Z., Schwartz, S., Wagner, L., and Miller, W. (2000). *A greedy algorithm for aligning DNA sequences. J. Comp. Biol. 7: 203–214*

[10] Anjali Mohapatra, P.M. Mishra, S. Padhy (2007): *Motif Search in DNA Sequences Using Generalized Suffix Tree, 10th International Conference on Information Technology.*

[11] Elena Zheleva and A.N.Arslan, 2005, *Fast motif search in protein sequence database*, Department of Computer Science, University of Vermont.

[12] N. S. Dasari, R. Desh, and Z. M.(2010) *An efficient multicore implementation of planted motif problem.* In Proceedings of the International Conference On High Performance Computing and Simulation, pages 9–15.

[13] Sanguthevar Rajasekaran, Sudha Balla, Chun-His Huang(2004): *Exact Algorithms for Planted Motif Problems.*Dept. of Computer Science and engineering, University of Connecticut.

[14] J. Davila, S. Balla, and S. Rajasekaran.(2007) *Fast and practical algorithms for planted (l, d) motif search.* IEEE/ACM Transactions on Computational Biology and Bioinformatics, 4:544–552 .

AUTHORS PROFILE

Khumukcham Robindro is currently working as an Assistant Professor in the Department of Computer Science, Manipur University. He joined department on 5th November, 2014. He is currently Principal Investigator (PI) of e-Varaha Project under ITRA, Media Lab Asia, MeiTy, Government of India, in the Department of Computer Science, Manipur University. His research interest includes Intelligent Systems, Knowledge Discovery, and Machine Learning etc.

Ashoke Das is Presently working as Associeate Professor in The Department of Mathematics, Raiganj University, West Bengal, India. He obtained his Bachelors, Masters and Doctoral Degree from the University of North Bengal, Darjeeling, West Bengal. He is actively engaged in the administration of Raiganj University. His area of interest are include Applied Mathematics, Real Analysis, Informatics and Computing, Society and Mathematical Applications etc .