Appl Intell
DOI 10.1007/s10489-015-0650-5

A novel locally guided genome reassembling technique using an artificial ant system

Susobhan Baidya · Rajat Kumar De

Abstract DNA reassembling is an NP-hard problem (Brun, Theor Comput Sci 395:31–46, 2008; Medvedev et al 2007; Ma and Lombardi 2008). The present article presents a locally guided global learning system to solve the problem of genome reassembling. We have used a reference DNA sequence which is 99 % similar to an unknown DNA sequence. Two different sequences from the same organism generally have around 99 % similarity (Wei et al 2007). We have considered different DNA sequences from NCBI website (http://www.ncbi.nlm.nih.gov). Then we have simulated the tasks of cloning the sequence, followed by shearing the clones to a number of short reads. In our algorithm, we have introduced a new concept in the task of DNA reassembling using Ant Colony Optimization, where pheromone concentration is proportional to the score of assembled DNA fragments with some known reference sequences within the same organism. Unlike local overlapping, we have used here local alignment score of short reads with some known local reference region as the heuristic information. The result shows that our algorithm is capable of reassembling at par with the state-of-the-art. DNA reassembling techniques may need a massive parallel computation and huge memory space (Kurniawan et al 2008) because of size $\sim 10^9$bp of DNA sequences of mammals (Miller et al, Genomics 95:315–327, 2010; Blazewicz et al, Comput Biol Chem 33:224–230, 2009; Butler et al, Genome Res 18:810–820, 2008; Joshi et al 2011; Stupar et al, Arch Oncol 19:3–4, 2011; Quail et al, BMC Genomics 13:1471–2164, 2012), and ACO is inherently concurrent in nature (Dorigo and Stutzle 2004). Due to lack of appropriate computational resources, we had to confine ourselves to deal with the sequences of length up to $\sim 10^5 bp$. We have considered 22 sequences of different organism, including *Homo sapiens* BRCA1 (127429bp) gene. For large sequences, we have applied hierarchical BAC-by-BAC sequencing (Fig. 2) (Myers, Comput Sci Eng 1:33–43, 1999), to stitch the individual segments to retrieve the original DNA sequence.

Keywords ACO · NP hard problem · DNA · Short reads · Hierarchical BAC-by-BAC sequencing

1 Introduction

DNA sequencing is the process of determining the precise order of nucleotides in a DNA molecule. It is the methodology that determines the order of the four types of nucleotides with bases adenine, guanine, cytosine and thymine, respectively, in a strand of DNA. Its importance includes determining sequence of an unknown gene, annotating a gene by its alignment with a known one, determining amino acid sequences in proteins and thereby their folding, disease identification, identifying sequence similarity between a pair of species, and determining the flow of heredity from predecessor to successor. Biomedicine [11] and forensics are the fields, where DNA sequencing concept is used.

Sequencing a DNA strand is a very tedious and computationally intensive job. A simple organism, like bacteria DNA contains sequences ∼2Mbp and an entire DNA string is wired [11]. Due to the lack of appropriate technology, we

S. Baidya
Department of Information Technology, Heritage Institute of Technology, Kolkata, India
e-mail: susobhan.baidya@heritageit.edu

R. K. De (✉)
Machine Intelligence Unit, Indian Statistical Institute, Kolkata, India
e-mail: rajat@isical.ac.in

Published online: 10 March 2015

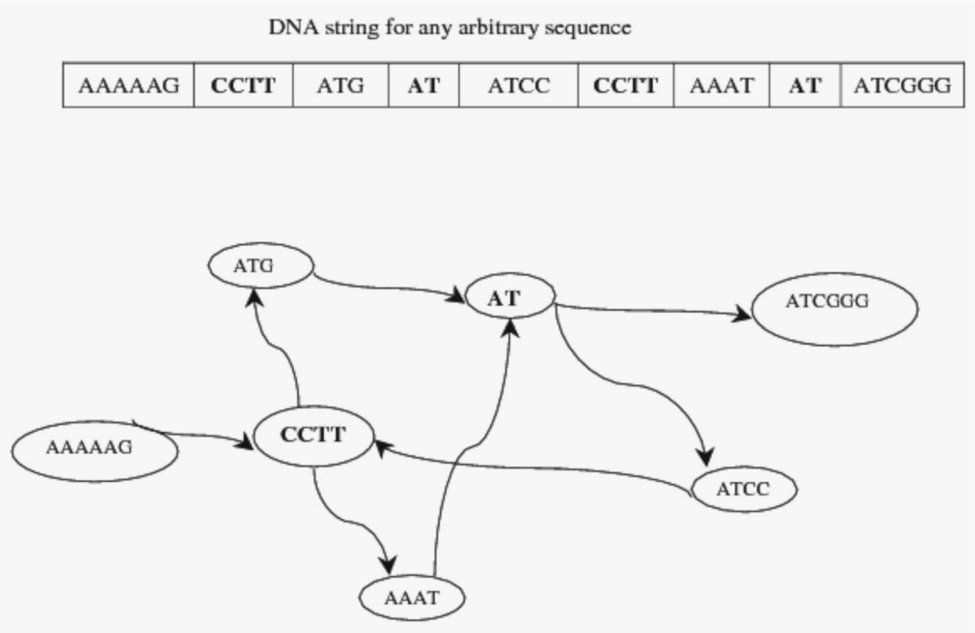

Fig. 1 Equivalent string graph for the above sequence

are unable to read the nucleotides in a DNA strand sequentially. That is why, a DNA segment, under sequencing, is cut through chemical processes [18, 26] into a number of short sequences, called reads (100bp-500bp). The reads are then reassembled using computational techniques to form the original DNA segment. DNA reassembling may need high level of parallel computation and high memory storage [21, 24]. Some of the computational challenges in DNA reassembling techniques include[1] the following.

- Whole genome sequencing techniques, under Next Generation sequencing methodology, produce a number of contigs by reassembling reads, which need to be reassembled further by sequence finishing mechanism [24].
- Overlapping between the reads may be false [30]. For example, a read with nucleotides GGGGG at one of its end has an overlap with another read with nucleotides GGGACTCCT, which may not be the actual overlap in the real sequence.
- During the cloning of DNA, there might be some deletion and mutation of bases.
- During random fragmentation, there may be loss of information at the edges of reads, and some reads may be lost too.
- Repeat regions may lead to a wrong sequence [32]. Approximately 50 % of the human genome comprises repeats (Fig. 1) [15]. Let us consider, a simple example without incorporating the complexity of clones and overlaps (Fig. 1). An equivalent string graph is shown for a particular sequence, where CCTT and AT are repeats in the sequence. If we consider AAAAAG as the starting read, the next obvious choice will be CCTT. After reaching CCTT, the correct choice will be ATG, but according to the graph, there is a chance to jump over to AAAT. Thus handling repeats are difficult to meet the exact solution.

Since the year 1997, there has been a flood of sequencing techniques. ABI370, EST sequencing, Phrap and Pyrosequencing were the successful technologies for reassembling relatively short DNA segments [28] compared to human genome. First time in 2004, human genome was assembled by IHGSC [28]. Velvet, SOAPdenovo, ABI-SOLid and ABySS [12, 14, 18, 24, 32] are some of the well known technologies, under next generation sequencing, which handle short reads (100bp-500bp) for genome $\sim 10^9$bp. Various assembler follow different algorithms, although they all primarily fall into the following categories[2].

Greedy graph based assembler: In this approach, an implicit graph is generated, where every node is a read and a weighted edge corresponds to the number of overlapping base pairs between the reads. The algorithm starts from a read, jumps to a next possible one based on their overlaps, and creates the shortest super string. This process continues until it reaches a threshold length. SSAKE was the first short read assembler [24] of this category. Phrap, Cap3 and TIGR are the assembler of this kind.

Overlap-layout-consensus (OLC): Here the method creates k-mer de Bruijn graph, where a read can appear

[1] https://www.cbcb.umd.edu/research/assembly_primer

[2] https://www.cbcb.umd.edu/research/assembly_primer

many times, but it is designated by a node for one time. The algorithm generates a number of simple paths, called contigs. Finally, the contigs are stitched manually comparing with some reference sequence. Newbler was the widely used software for this approach [24].

Eulerian path, de Bruijn graph based approach: In this method, initial steps are the same as that of OLC. Then the algorithm finds out a set of Eulerien paths from the de Bruijn graph. The paths are finally stitched manually with the help of some reference sequence. Euler, Velvet, Allpath, Abyss and SOAPdenovo are some available software [24] under this category.

Align-layout-consensus: Here a known reference sequence is used for reassembling the reads generated from a DNA. Thus the problem of assembling becomes placing the respective reads in their appropriate positions based on some sequence alignment algorithm. Projector2, Mozaik, ELAND and MUMmer[3] are some software under this category.

BAC-by-BAC (hierarchical) sequencing: In this approach, the original sequence is fragmented and mapped to particular locations by using specialized laboratory experiments (Fig. 2). Then the individual segments are fragmented into reads, and then the reads are reassembled into corresponding individual segments. These segments are further reassembled to form the original DNA sequence. Celera and Atlas are the widely used assembler of this kind [25].

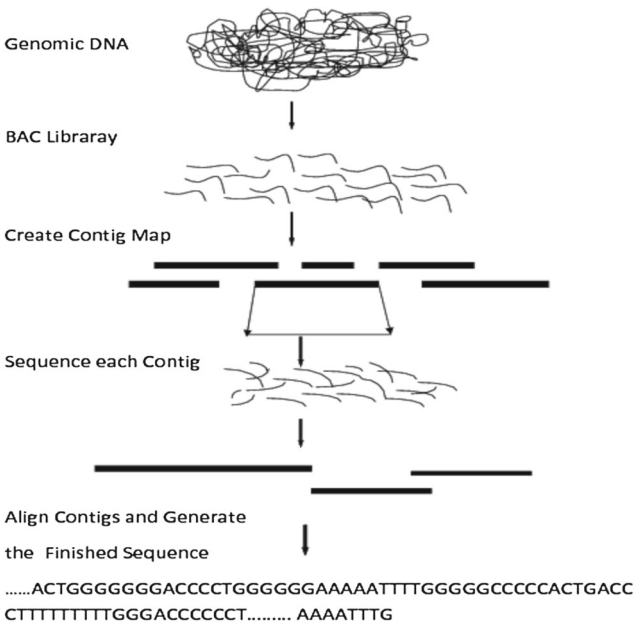

Fig. 2 Hierarchical BAC-byBAC sequencing

Several algorithms, based on nature inspired computing, exist in literature, which reassemble nucleotide sequences to form a part of DNA. The methods based on genetic algorithms (GrEA, Distributed GA, Grid based GA) [4–7] have considered sequences of length between 10^2bp and 77×10^3bp. The algorithms based on Particle Swarm Optimization (DSAPSO, CPSO) [2, 3] have considered 20×10^2bp sequences. Bees Colony Optimization algorithms (ABC, QEGA) [10] have worked with $\sim 77 \times 10^3$bp sequences. Ant Colony Optimization algorithms (ACO for sequencing by hybridization technique, ML-ACO, ACS) [13, 23, 31] have dealt with sequences of length between 5×10^{10}bp and 21×10^3bp. Most of the authors have claimed the novelty of their algorithms based on the solution quality in terms of fitness values or fragment similarity score values. Existing solutions have been obtained on the basis of overlapping information between the reads. These methods use the information of overlapping base pairs of reads as heuristics, and fitness values that are again based on overlapping base pairs. Thus, they have the tendency to select the next highest overlapping reads, and thereby there is a chance of getting stuck to local optima. In the present article, we have worked with local scores of reads, and proposed a nature inspired Ant Colony Optimization algorithm where larger sequences have been considered.

We call the present algorithm as Local Score guided Ant Colony Optimization (LSACO). Unlike de Bruijn graph, we have not used any k-mer graph, and therefore, there is no anomaly for handling repeats. Here we have considered every read as a place, even if the read repeats over the sequence. The present LSACO algorithm incorporates local score corresponding to a read as local guidance for the movement of an ant. We have compared our methodology with some existing nature based heuristic algorithms, for DNA reassembling problems, like Ant Colony Optimization algorithm, which is abbreviated here as OVACO (Overlapped base pairs guided Ant Colony Optimization), Genetic Algorithm (GA), Particle swarm optimization (PSO) and Bees colony optimization (BCO) algorithms.

We have implemented and reassembled the DNA fragments using some basic heuristic algorithms like ACO, PSO, BCO, GA, where we have incorporated the concept of overlapping base pairs, as mentioned before, and compared them with our methodology. For example, in the existing Ant Colony Optimization Algorithm, an ant starts from a starting place and moves to an unvisited

[3] https://www.cbcb.umd.edu/research/assembly_primer

place with the knowledge of pheromone concentration and overlapping base pairs between the reads. On the other hand, the present LSACO algorithm uses local score for a read along with pheromone concentration for moving an ant from one place to another. Finally the ant reaches to a target place and updates pheromone concentration based on the solution score. We have demonstrated the effectiveness of our methodology, and its superiority over existing ones on 22 sequences of different organisms, like *Taenia solium*, HIV, Influenza virus, Hepatitis B virus, Aphid lethal paralysis virus strain, *Rattus norvegicus* strain, *Drosophila melanogaster* and *Homo sapiens* breast cancer. The present methodology has resulted in 99.9 %-100 % accuracy with the original sequences. We had to restrict ourselves to the sequences of length up to $\sim$ 10^5bp due to lack of appropriate computing power.

2 Methodology

Let us consider a set of m reads generated from a given DNA sequence. For a pair of i^{th} and j^{th} reads, let w_{ij} be the number of bases that are common towards the tail end of i^{th} read and head end of j^{th} read, if both i^{th} and j^{th} reads have been generated from different clones. Otherwise, w_{ij} is set to '-1' (an arbitrary negative number). Our task is to regenerate the given DNA sequence by assembling some of these reads. For each read, we consider a place. Let $T = [\tau_{ij}]_{m\times m}$ be the pheromone concentration matrix of order $m \times m$ such that τ_{ij} represents the pheromone concentration on the arc from i^{th} place to j^{th} place. Initially, these τ_{ij}'s assume random numbers in [0, 10]. Since starting read of a given sequence is not known to us, we consider a dummy starting place.

Now m ants are assumed to be assembled at the dummy starting place, and are allowed to move to the other unvisited places based on certain criteria. The probability that q^{th} ant will move from i^{th} place to j^{th} place is given by

$$p_{ij}^{q} = \frac{\tau_{ij}^{\alpha} \times s_{ij}^{\beta}}{\sum_{j' \epsilon N_i} \tau_{ij'}^{\alpha} \times s_{ij'}^{\beta}} \quad (1)$$

Here, N_i is the set of neighboring unvisited places from i^{th} place. The term s_{ij} stands for local alignment score, obtained by Needleman-Wunsch algorithm, where the part towards the tail end of the read corresponding to i^{th} place is aligned with the head end of the read corresponding to j^{th} place, based on the part of the reference sequence between $l - \triangle$ and $l + n_{ij} + \triangle$, l being the partial solution length till an ant reaches i^{th} place, and $n_{ij} = |r_j| - w_{ij}$ being the number of added bases as the ant moves from i^{th} place to j^{th} place, and r_j being the number of bases in j^{th} read. The value of $\triangle$ is chosen to be an integer in [1, 5].

The parameters α and β (in (1)) have the following effects. If $\alpha = 0$, the selection probability (p_{ij}^{q}) of j^{th} place is proportional to s_{ij}^{β}. It implies that the chance of selecting j^{th} place to be jumped from i^{th} place is maximum over the other places, if the reads corresponding to i^{th} and j^{th} places have the highest local alignment score. In this case, ACO corresponds to a classical greedy algorithm. If $\beta = 0$, only pheromone information is at work. That is, the chance of selecting j^{th} place to be jumped from i^{th} place is maximum over the places, if the pheromone concentration on the arc between i^{th} and j^{th} places is the highest among that between the unvisited neighboring places and the i^{th} place. We have considered 10 observations on HIV8929 sequence (Table 2) for choosing proper values of the parameters α and β, where the values of τ_{ij} and s_{ij} are in [1, 50] and [1, 10] respectively. Based on extensive experimentation, the most stable values of α and β values have been found to be 1 and

Table 1 α and β parameter values, considering the data instance HIV-1 isolate 8179, UK, nonfunctional gag gene

Observation number	value of α	value of β	% of accuracy in assembled DNA by LSACO
1	1	1	98
2	1	2	99.9
3	1	3	99.
4	1	4	98.7
5	1	5	98.7
6	2	1	99.9
7	3	1	99.7
8	4	1	99.8
9	5	1	98.9
10	2	4	99.9

2 (Table 1) respectively, for the problem under consideration. Actually the ratio of α and β has to be 1 : 2. Otherwise, if any one of s_{ij} or τ_{ij} suppresses the effect of the other, the solution will diverge from an optimal one. The fact is that the effect of s_{ij} and τ_{ij} should be balanced in (1). Now the task is to search for a path from the dummy starting place to a place for which the partial solution length l is greater than θ, a threshold value being the length of the reference DNA sequence. Thus, it completes the tour of q^{th} ant, and the alignment score of resulting reassembled DNA sequence with the reference one is computed by Needleman-Wunsch algorithm. Let this score be SC_q corresponding to q^{th} ant.

Then the ants will return to the dummy starting place via the same path through which they have reached their respective terminating places. During their way back, they drop pheromone on each arc between a pair of places. It is done by modifying τ_{ij} value as q^{th} ant moves, on its way back, from i^{th} place to j^{th} place. The value of τ_{ij} is modified using

$$\tau_{ij}^{(new)} = \tau_{ij}^{(old)} + \triangle\tau_{ij}^{q} \quad \forall i, j \ \epsilon P^q, \tag{2}$$

where $\triangle\tau_{ij}^{q}$ is the amount of pheromone deposited by q^{th} ant on the arc connecting i^{th} and j^{th} places, on the tour along the path P^q (the set of arcs on the path of q^{th} ant), and is zero otherwise. The term $\triangle\tau_{ij}^{q}$ is defined as $\triangle\tau_{ij}^{q} = SC_q$, if q^{th} ant has made a tour along P^q. Otherwise, it is zero.

In the next step, there will be a uniform evaporation of pheromone on all the arcs between pairs of places. Thus, for the arc between i^{th} and j^{th} places, it is

$$\tau_{ij}^{(new)} = (1-\rho)\tau_{ij}^{(old)}, \forall i, j, \tag{3}$$

where $0 < \rho < 1$ is called the pheromone evaporation rate. It is used to avoid unlimited accumulation of pheromone trails, and thereby enables the algorithm to forget a bad decision. After evaporation, all the ants deposit pheromone on the arcs that they have travelled through the tour, and the corresponding pheromone information is updated by (3). All the ants will move through the places in parallel. The number of ants that will move in parallel in an iteration, is equal to the number of reads formed [16, 17]. The steps of evaporation and update of pheromone can be altered.

It is to be mentioned here that the present methodology considers a slightly different expression for p_{ij}^{q} (1), from that existing in literature [13, 23, 31]. In literature, p_{ij}^{q} is defined as

$$p_{ij}^{q} = \frac{\tau_{ij}^{\alpha} \times w_{ij}^{\beta}}{\Sigma_{j' \epsilon N_i} \tau_{ij'}^{\alpha} \times w_{ij'}^{\beta}} \tag{4}$$

Table 2 Data sets

Serial number	Data Set	Abbreviation	Sequence Length	average no. of reads
1	TAEACTB *Taenia solium* (clone pAT6) actin gene, complete cds	Taenia1837	1837	36
2	*Taenia solium* (clone pAT5) actin gene	Taenia1899	1899	38
3	HIV-1 isolate 8179 ,UK,nonfunctional gag protein	HIV8929	8929	172
4	HIV-1 isolate 99ET8 ,Ethiopia gag polyprotein	HIV8746	8746	167
5	HIV-1 isolate 17TB4-2G10 USA gag protein	HIV8998	8998	171
6	HIV-2 clone 8407A-06-12 env pseudogene	HIV2591	2591	50
7	HIV-2 isolate 97PTHDESC11 Portugal glycoprotein	HIV2583A	2583	50
8	HIV-2 isolate 97PTHDESC17 Portugal glycoprotein	HIV2583B	2583	51
9	INFL-Sequence 37 from Patent WO2010036948	INFL6126	6126	116
10	INFL-Sequence 3 from Patent EP2483428	INFL2259	2259	46
11	INFL-Sequence 41 from Patent WO2010036948	INFL6125	6125	117
12	INFL-Sequence 3 from Patent EP2483428	INFL2261	2261	44
13	Hepatitis B , Fukuoka Red Cross HBV e-negative	HBV3212	3212	61
14	H.sapiens genomic DNA with integrated HBV DNA	HBV3958	3958	80
15	Hepatitis B virus, isolate: HBV PG-Yohko	HBV3182	3182	62
16	7574.1—HSU37574 Human BRCA1 gene	BRCA3787	3787	74
17	Aphid lethal paralysis virus strain	Aphid4175	4175	78
18	Gene silencing vector pCPCbLCVA.007	pCPCbLCVA5352	5352	105
19	*Drosophila melanogaster* copia-like element 17.6	Drosophila7439	7439	141
20	Human sequence XX-92M18 , chromosome 13. breast cancer 2	BRCA68879	68879	1326
21	*Rattus norvegicus* strain BN/SsNHsdMCW chromosome 12	norvegicus52968	52968	1011
22	*Homo sapiens* BRCA1 (BRCA1), (rpL21) pseudogene	BRCA127429	127429	2409

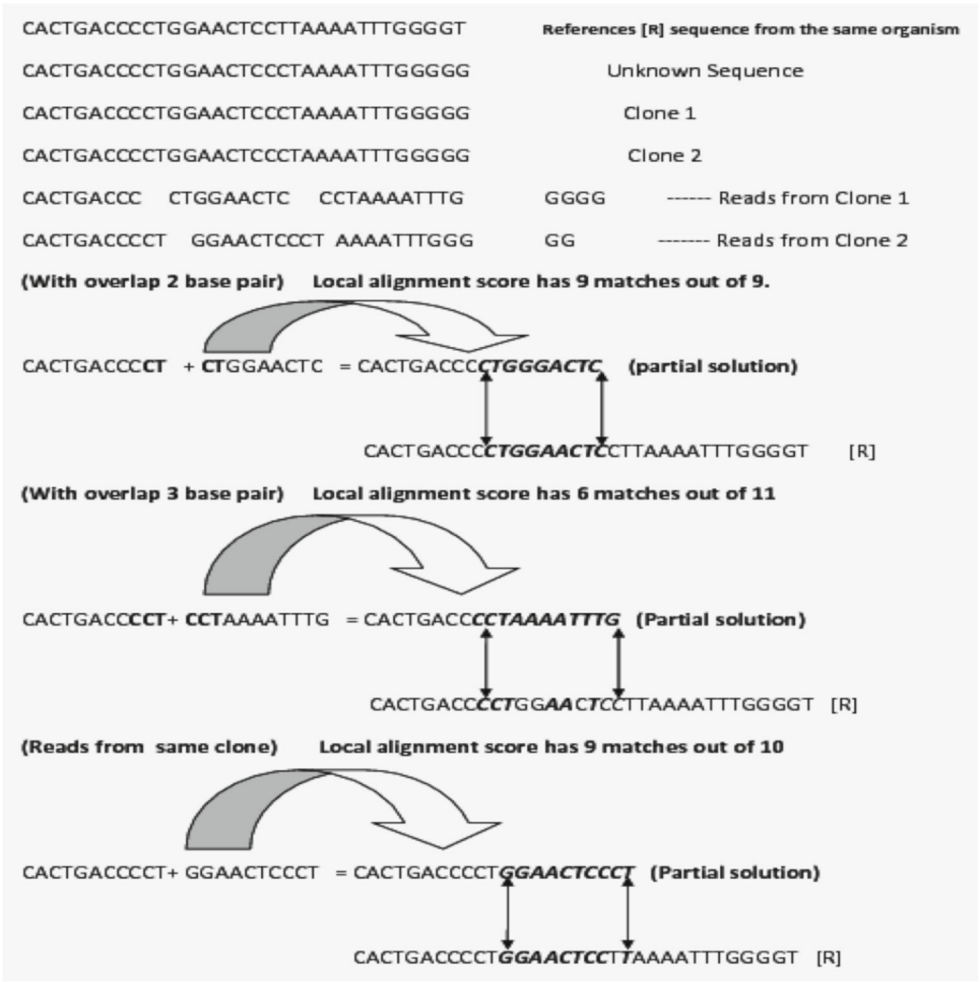

Fig. 3 Clones and Reads

In earlier techniques, the authors considered the maximum overlap among the reads for reconstruction of a fragmented DNA. They used w_{ij} as local heuristics, where w_{ij} is the overlap between i^{th} and j^{th} reads. In the following section, we explain how to assemble the reads, where we have considered the case of concatenation of two reads, if successive reads belong to the same clone. It is to be mentioned here that this type of concept already exists in the methods of PET or Pair Ended Read [33]. In Section 3, we show that our approach is little better over the existing Ant Colony approach in terms of accuracy of the assembled sequence. The accuracy of final assembled sequence is estimated by its alignment with the original sequence using Needleman-Wunsch algorithm.

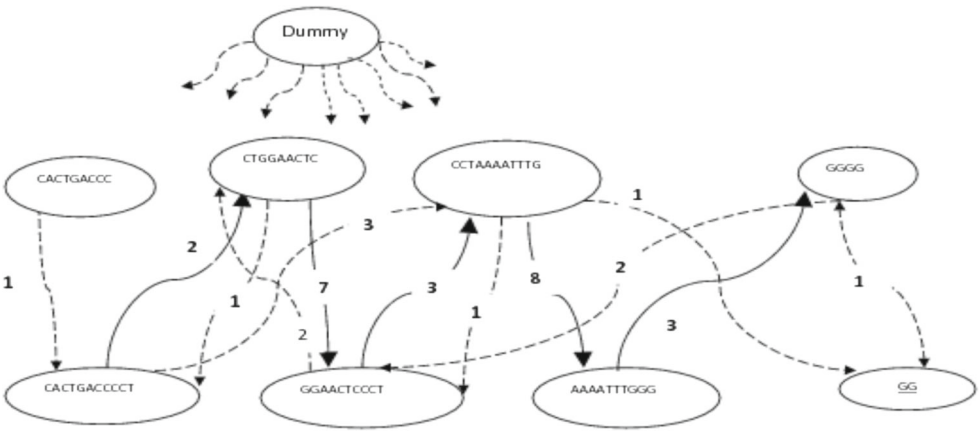

Fig. 4 Equivalent plane with possible movement by the matrix W

Explanation with an example Here we explain our methodology with a simple example depicted by Figs. 3 and 4. Here we try to portray that maximum overlap is not always a good decision for a jump from one place to another, but the source of the reads, i.e, the source clone information is also an important information for reassembling the reads. In Fig. 3, we consider a known reference DNA and an unknown DNA sequence. The unknown DNA sequence has been cloned twice, and then they have been fragmented into eight reads. We consider the fact that about 99 % of the sequence is the same within the same organism. Initially, we have eight places corresponding to these eight reads without any information on the starting reads and end reads.

According to Fig. 3, the starting read may be CACTGACCCCT or CACTGACCC, and finishing read may be GGGGG or GG, which is not known to us. Thus, a dummy starting place has been considered. The dummy place is connected with all the places with some logical overlaps, and the reads within the same clone are also connected. For simplicity, we have not shown these connections. An ant will start from the dummy place, and looks to jump to a place based on probability values (in (1)). Our methodology first creates a matrix $W = [w_{ij}]_{m\times m}$, such that the arc between i^{th} and j^{th} places is labeled with w_{ij} (Fig. 4). The path indicated by a solid arrow is one of the correct sequence paths. The weight w_{ij} between the places i and j represents the extent of overlapping. In general, $w_{ij} \neq w_{ji}$. Since we consider each clone separately and create reads separately, we have to check the reads, if they have been generated from the same clone. The matrix $W = [w_{ij}]_{m\times m}$ has the following properties.

- If i^{th} and j^{th} reads are from different clones with an overlap of k ($k \geq 0$) base pairs, $w_{ij} = k$, where $i \neq j$.
- If i^{th} and j^{th} reads are from the same clone then $w_{ij} = -1$, where $i \neq j$.
- If $i = j$ then $w_{ij} = 0$.

Let us suppose that we have got two reads GGGACTCCCT and CCTAAAATTTG from different clones, and thus the longest prefix cum suffix is of length three. After assembling, it will be GGGACTCCCTAAAATTTG. If they belong to the same clone then j^{th} read is appended to i^{th} read, and the result will be GGGACTCCCTCCTAAAATTTG. Figure 4 represents an instance of Fig. 3. For simplicity, we consider only the connection like $w_{ij} = k$, where $k > 0$. Let us consider that after starting from

Table 3 Comparison of LSACO, OVACO, PSO, BCO, GA

Serial Number	Data set	% of accuracy by LSACO	% of accuracy by OVACO	%of accuracy by PSO	%of accuracy by BCO	%of accuracy by GA
1	Taenia1837	99.9	99.9	98.7	99.3	70.2
2	Taenia1899	100	99.9	98.7	99.2	70.2
3	HIV8929	99.9	99	97.8	98.2	66.3
4	HIV8746	99.9	98.5	97.3	97.7	65
5	HIV8998	100	99.9	98.7	99.1	65.6
6	HIV2591	100	99.8	98.6	99	70.1
7	HIV2583A	100	99.9	98.7	99.2	69.7
8	HIV2583B	100	99.9	98.6	99.2	70
9	INFL6126	99.9	99.9	98.7	99	70.2
10	INFL2259	100	99.9	98.5	99.2	70.1
11	INFL6125	99.9	99.9	98.4	99	68.2
12	INFL2261	99.9	99.8	98.6	99.2	70
13	HBV3212	99.9	99.8	98.5	99	70.1
14	HVB3958	100	99.9	98.4	99.2	69
15	HBV3182	100	99.9	98.7	99.1	70
16	BRCA3787	100	99.9	98.7	99.1	70.2
17	Aphid4175	100	99.9	98.6	99	70
18	pCPCbLCVA5352	100	99.9	98.2	99.2	66.3
19	Drosophila7439	100	99.9	97.5	99.1	66.6
20	BRCA68879	100	98.6	82.1	85	55.5
21	norvegicus52968	100	98.9	84	87.3	57
22	BRCA127429	100	95.8	75.2	82	52.4

the dummy starting place, an ant moves to CACTGACCCCT. The partial tour length (l) is 11 and partial solution is CACTGACCCCT (Fig. 3). Now the reads corresponding to neighboring places are GGAACTCCCT, AAAATTTGGG, GG, CTGGAACTC and CCTAAAATTTG. If the next read is CTGGAACTC then i corresponds to CACTGACCCCT and j to CTGGAACTC. They belong to different clones and have an overlap of 2bp. After assembling, partial solution will be CACTGACCCCTGGAACTC and l is 18. The darker portions of the arrow is the region to be considered for calculation of s_{ij}, which is equal to $((L_\theta/(L_l)) \times 10$, where L_θ being local score over the darker region which is indicated by arrows. The local score calculated as described in Fig. 3. Let L_l be the length of the considered region. Here $L_l = 9bp$. It implies the matching to 9bp out of 9bp within that portion. If we go for a larger overlap from current place CACTGACCCCT to next place CCTAAAATTTG, then local matching reduces to 6bp out of 11bp. It may lead to a less accurate solution (which is true for the current example) due to assembling the reads CACTGACCCCT and CCTAAAATTTG. Thus we get a partial solution CACTGACCCCTAAAATTTG, where partial solution length becomes 19. In reality, we have to consider a little drift $\triangle$ over the considered region in left and right sides. This process of visiting the places will continue until $l \leq \theta$. From the above examples (Figs. 3 and 4), it is clear that only the knowledge of overlap between the reads may not lead to an optimal solution.

Let us now consider the case where we do not know whether the reads belong to the same clone. In that case, there might be a wrong partial solution, which may lead to inaccurate solution. For example, consider the places corresponding to the reads CATGACCC, CTGGAACTC, CCTAAAATTTG and GGGG, belonging to the same clone. These places have been selected by an ant one by one. The ant will assemble those reads with the knowledge of overlaps only, and finally the assembled sequence

Table 4 Statistical Analysis by Wilcoxon Test for LSACO and OVACO, W_t-value: 0, Mean Difference: 0.08, Sum of positive ranks: 190, Sum of negative ranks: 0

Serial Number	Data set	% of accuracy by LSACO	% of accuracy by OVACO	Difference	Ranked Difference
1	Taenia1837	99.9	99.9	0	n/a
2	Taenia1899	100	99.9	0.1	6
3	HIV8929	99.9	99	0.9	15
4	HIV8746	99.9	98.5	1.4	17.5
5	HIV8998	100	99.9	0.1	6
6	HIV2591	100	99.8	0.2	14
7	HIV2583A	100	99.9	0.1	6
8	HIV2583B	100	99.9	0.1	6
9	INFL6126	99.9	99.9	0	n/a
10	INFL2259	100	99.9	0.1	6
11	INFL6125	99.9	99.9	0	n/a
12	INFL2261	99.9	99.8	0.1	12.5
13	HBV3212	99.9	99.8	0.1	12.5
14	HVB3958	100	99.9	0.1	6
15	HBV3182	100	99.9	0.1	6
16	BRCA3787	100	99.9	0.1	6
17	Aphid4175	100	99.9	0.1	6
18	pCPCbLCVA5352	100	99.9	0.1	6
19	Drosophila7439	100	99.9	0.1	6
20	BRCA68879	100	98.6	1.4	17.5
21	norvegicus52968	100	98.9	1.1	16
22	BRCA127429	100	95.8	4.2	19

The critical value of W_t (parameter of Wilcoxon Test) for $N_p = 19$ at $p \leq 0.05$ is 46. Therefore, the result is significant at $p \leq 0.05$

becomes CATGACC**C**TGGAACT**C**CTAAAATTT**G**GGG. This is not an actual partial solution as we consider logical overlaps among the reads CATGACCC, CTGGAACTC, CCTAAAATTTG and GGGG, where there is no actual overlap.

Algorithm 1

Input: $W = [w_{ij}]_{m\times m}$, weight values representing the extent of overlap between pairs of reads obtained from different clones,
known reference sequence 'seq', sequence length 'θ' ,
the pheromone concentration matrix $T = [\tau_{ij}]_{m\times m}$.
Output: An optimal reassembled DNA sequence.

Do the following steps till there is further improvement of a solution in consecutive two iterations.

- *Step I*: Initialize $\tau_{ij}s$, in [1,5].
- *Step II*: Put m ants at the dummy starting place and allow them to begin their journey in parallel.
- *Step III*: For each q^{th} ant, do the following steps:
  - *Step III.1*: Allow q^{th} ant to move from current place i to next possible place j by (1), where $w_{ij} > 0$ or $w_{ij} = -1, \forall j$.
    - *Step III.2.a*: If q^{th} ant has reached the terminating place for which partial solution length $l \geq \theta$, then calculate SC_q values using Needleman-Wunsch algorithm, for the alignment of a sequence with the reference sequence seq, and *Go to Step IV*.
    - *Step III.2.b*: Else *Go to Step III.1*.
- *Step IV*: For each q^{th} ant, update pheromone concentration using (2) for each arc on the path of q^{th} ant.
- *Step V*: Evaporate pheromone on all the arcs using (3).
- *Step VI*: Retain the best sequence so far obtained, and *Go to Step II*.

3 Results

In this section, we demonstrate the effectiveness of LSACO in reconstructing several genome sequences based on next generation sequence data. We have considered 22 sequences of different organisms, like *Taenia solium*, *HIV*, *Influenza*, *Hepatitis B*, *Drosophila*, *Breast cancer of human*, *Aphid lethal paralysis*, obtained from NCBI database[4]. Due to lack of appropriate computational resources, we had to confine ourselves with the sequences up to the length of $\sim 10^5$bp.

Here we have applied our Local Score Guided Ant Colony algorithm (LSACO) to aforesaid data sets, and its performance has been compared with some existing methods based on Ant colony optimization algorithms (OVACO) [13, 23, 31], Genetic Algorithms (GA) [4–6], Particle Swarm Optimization (PSO) [2, 3] and Bees Colony Optimization (BCO) algorithms [8–10]. In all these cases, the authors have considered the overlap base pairs as local heuristics for their problems. In our algorithm, we have considered local score s_{ij} of the reads as local heuristics. For validating the solution, output sequences are rewarded by calculating its score of alignment with the original sequence. Besides, we have performed Wilcoxon two-tailed statistical rank test [1] for comparing the performance of the algorithms as well as the validation of the results.

The main challenges of next generation sequencing is a high level of computation. The algorithm of Needleman-Wunsch has the time and space complexities of $O(N^2)$, where N is the total number of base pairs in the reference sequence. We have implemented LSACO in 64 bit operating systems in HP Proliant server with 16GB RAM and 2-quad core processor. It is very hard to handle the sequences of length more than 10^5bp as for every tour by an ant, LSACO calls Needleman-Wunsch algorithm that needs $O(N^2)$ time.

3.1 Implementation issues

We have taken the sequences from NCBI website and considered them as known sequences. We have altered a known DNA sequence by deletion and mutation in such a way that the known sequence is similar to the new one by 99 %. We have considered the new sequence as the given unknown DNA sequence that has been fragmented into short reads, and the known one as the reference sequence. We have created simple reads of length 100bp to 300bp with 3X coverage. Then we have deleted a few reads randomly. It is done to simulate the fact that some reads may be lost during their formation by some biochemical processes. In our methodology, we have generated m reads from an unknown DNA sequence with 3X coverage. In each iteration, an ant, in its journey, has assembled the reads with the help of the reference sequence based on a local score. Finally the score

[4] http://www.ncbi.nlm.nih.gov

of the assembled DNA has been considered for updating pheromone concentration.

Here we have considered the simple reads with 1 % to 3 % information missing during the generation of reads. Normalization of pheromone concentration values is required after a certain number of iterations, otherwise there will be an overflow. We have considered $s_{ij} \geq 1$ and $SC_q \geq 0$. We have normalized the values of SC_q in [0, 1000], τ_{ij} in [1, 50] and s_{ij} in [1, 10]. Since we have assumed a dummy starting place, the sizes of both the matrices T and W have become $(m + 1) \times (m + 1)$. We have used the concept of Roulette wheel selection for (1) and (4). Finally the results of LSACO have been compared with that of the existing algorithms using Wilcoxon two-tailed statistical rank test.

For a sequence of length above 10Kbp, we have applied the concept of hierarchical sequencing, where the original sequence is fragmented and mapped to a particular location by using specialized laboratory experiments (by STS probe or by other location marker). We have cut an entire sequence into those of length around 10Kbp, and each segment has overlap with contiguous segments. Then we have fragmented individual segments randomly into reads, and reassembled the reads into corresponding individual segments. Finally, we have reassembled individual reassembled segments further and retrieved an optimal sequence.

3.2 Data sets

We have considered 22 sequences of different organisms (Table 2). Among them, two *Teania solium* sequences - TAEACTB *Taenia solium* (clone pAT6) actin gene and *Taenia solium* (clone pAT5) actin gene have been considered.The genes pAT6 and pAT5 have lengths 1837bp and 1899bp, respectively. For both the sequences, we have created 36 and 38 reads, respectively. Similarly, three human immunodeficiency virus sequences of HIV-1, and three of HIV-2 of lengths 8929bp, 8746bp, 8998bp, 2591bp, 2583bp and 2583bp respectively, have been considered. For all these sequences, 172, 167, 171, 50, 50 and 51 reads, respectively, have been generated. Four sequences of influenza virus of lengths 6126bp, 2259bp, 6125bp and 2261bp have been considered. For these sequences, 116, 46, 117 and 44 reads, respectively, have been generated. For three sequences of hepatitis-B virus of lengths 3212bp,

Table 5 Statistical Analysis by Wilcoxon Test for LSACO and PSO, W_t-value: 0, Mean Difference: 1.27, Sum of positive ranks: 253, Sum of negative ranks: 0

Serial Number	Data set	% of accuracy by LSACO	% of accuracy by PSO	Difference	Ranked Difference
1	Taenia1837	99.9	98.7	1.2	1.5
2	Taenia1899	100	98.7	1.3	5
3	HIV8929	99.9	97.8	2.1	17
4	HIV8746	99.9	97.3	2.6	19
5	HIV8998	100	98.7	2.3	5
6	HIV2591	100	98.6	1.4	10.5
7	HIV2583A	100	98.7	1.3	5
8	HIV2583B	100	98.6	1.4	10.5
9	INFL6126	99.9	98.7	1.2	1.5
10	INFL2259	100	98.5	1.5	13.5
11	INFL6125	99.9	98.4	1.5	13.5
12	INFL2261	99.9	98.6	1.3	8
13	HBV3212	99.9	98.5	1.3	10.5
14	HVB3958	100	98.4	1.6	15
15	HBV3182	100	98.7	1.3	5
16	BRCA3787	100	98.7	1.3	5
17	Aphid4175	100	98.6	1.4	10.5
18	pCPCbLCVA5352	100	98.2	1.8	16
19	Drosophila7439	100	97.5	2.5	18
20	BRCA68879	100	82.1	17.9	21
21	norvegicus52968	100	84	16	20
22	BRCA127429	100	75.2	24.8	22

The critical value of W_t (parameter of Wilcoxon Test) for $N_p = 22$ at $p \leq 0.05$ is 65. Therefore, the result is significant at $p \leq 0.05$

Table 6 Statistical Analysis by Wilcoxon Test for LSACO and BCO, W_t-value: 0, Mean Difference: 0.77, Sum of positive ranks: 253, Sum of negative ranks: 0

Serial Number	Data set	% of accuracy by LSACO	% of accuracy by BCO	Difference	Ranked Difference
1	Taenia1837	99.9	99.3	0.6	1
2	Taenia1899	100	99.2	0.8	5.5
3	HIV8929	99.9	98.2	1.7	13
4	HIV8746	99.9	97.7	2.2	14
5	HIV8998	100	99.1	0.9	10
6	HIV2591	100	99	1	11.5
7	HIV2583A	100	99.2	0.8	5.5
8	HIV2583B	100	99.2	0.8	5.5
9	INFL6126	99.9	99	0.9	10
10	INFL2259	100	99.2	0.8	5.5
11	INFL6125	99.9	99	0.9	10
12	INFL2261	99.9	99.2	0.7	2
13	HBV3212	99.9	99	0.9	10
14	HVB3958	100	99.2	0.8	5.5
15	HBV3182	100	99.1	0.9	10
16	BRCA3787	100	99.1	0.9	10
17	Aphid4175	100	99	1	11.5
18	pCPCbLCVA5352	100	99.2	0.8	5.5
19	Drosophila7439	100	99.1	0.9	10
20	BRCA68879	100	85	15	16
21	norvegicus52968	100	87.3	12.7	15
22	BRCA127429	100	82	18	17

The critical value of W_t (parameter of Wilcoxon Test) for $N_p = 22$ at $p \leq 0.05$ is 65. Therefore, the result is significant at $p \leq 0.05$

3958bp and 3182bp, corresponding numbers of reads are 61, 80 and 62 respectively. In the case of the sequence of *Drosophila melanogaster* of length 7439bp, 141 reads have been generated. For Aphid lethal paralysis virus of length 4175bp, 78 reads have been generated. We have considered a sequence for Gene silencing vector *pCPCbLCVA.007* of length 5352bp, for which 105 reads have been generated. Finally, we have considered three sequences of breast cancer gene of human and rat of lengths 68879bp, 52968bp, and 127429bp, respectively. In order to handle these genes of large base pairs, we have cut individual segments around 10Kbp (Fig. 2). For all these sequences, 1326, 1011 and 2409 reads have been generated. The entire data set is depicted in Table 2.

3.3 Analysis

For all the aforesaid data sets, we have got accuracy of about 99.9 %-100 % (Table 3) using LSACO. Although the results of LSACO and the existing OVACO are almost similar, OVACO has resulted in about 98 % accuracy in some cases. The results of PSO and BCO have been found to be worse than that of OVACO, as they have resulted in 85 % and 75 % sequence accuracy in the worst cases respectively. The accuracy of the results obtained by GA has been found between 52 % and 70 %. The results of LSACO for which the accuracy was less than 100 %, may due to the fact that there has been a variation in length of the given reference sequence with the sequence from which reads have been generated.

We have compared the performance of existing LSACO with the other algorithms using online[5] Wilcoxon non-parametric statistical tool. Tables 3, 4, 5, 6 and 7 show results of the present algorithm and the existing ones. Here, the value of the number of participants in Wilcoxon Test is represented by N_p. In Tables 4-7, there is no negative rank difference. Results depict that the sum of negative differences is zero for all the aforesaid tables, and thereby $W_t = 0$ (parameter of Wilcoxon Test) for all the cases. Thus we can conclude that the results of LSACO are significant at $p \leq 0.05$ over the existing algorithms considered here.

[5] http://www.socscistatistics.com/tests/signedranks/Default2.aspx

Table 7 Statistical Analysis by Wilcoxon Test for LSACO and GA, W_t-value: 0, Mean Difference: 29.77, Sum of positive ranks: 253, Sum of negative ranks: 0

Serial Number	Data set	% of accuracy by LSACO	% of accuracy by GA	Difference	Ranked Difference
1	Taenia1837	99.9	70.2	29.7	1.5
2	Taenia1899	100	70.2	29.8	3.5
3	HIV8929	99.9	66.3	33.6	16
4	HIV8746	99.9	65	34.9	19
5	HIV8998	100	65.6	34.4	18
6	HIV2591	100	70.1	29.9	7
7	HIV2583A	100	69.7	30.3	12
8	HIV2583B	100	70	30	10
9	INFL6126	99.9	70.2	29.7	1.5
10	INFL2259	100	70.1	29.9	7
11	INFL6125	99.9	68.2	31.7	14
12	INFL2261	99.9	70	29.9	7
13	HBV3212	99.9	70.1	29.8	5
14	HVB3958	100	69	31	13
15	HBV3182	100	70	30	10
16	BRCA3787	100	70.2	29.8	3.5
17	Aphid4175	100	70	30	10
18	pCPCbLCVA5352	100	66.3	33.7	17
19	Drosophila7439	100	66.6	33.4	15
20	BRCA68879	100	55.5	44.5	21
21	norvegicus52968	100	57	43	20
22	BRCA127429	100	52.4	47.6	22

The critical value of W_t (parameter of Wilcoxon Test) for $N_p = 22$ at $p \leq 0.05$ is 65. Therefore, the result is significant at $p \leq 0.05$

Table 8 shows the overall execution time of the aforesaid algorithms. It has been observed that the execution time of LSACO is high compared to OVACO, PSO, BCO and GA, but LSACO is more stable and accurate. On the other hand, for shorter sequences, the results are more or less similar for LSACO and OVACO, while for sequences of greater length, the algorithm OVACO becomes a little unstable compared to LSACO as far as the accuracy is concerned.

OVACO takes less amount of time as it considers the overlap information for moving one place to another by an ant. The value of w_{ij} has been estimated before starting the tours by ants, and thereby time complexity for computing p_{ij}^{q} (4) has been less. That is why, there is a chance to miss the actual neighboring read which has no overlap with the current read (reads from the same clone). The same thing happens in the case of PSO, BCO and GA, where the solution space has been created using the information on overlap between two adjacent reads. Thus the methods of OVACO, PSO, BCO and GA run faster compared to LSACO at the cost of quality of the solutions. It is due to the fact that LSACO considers all the unvisited places corresponding to overlapping and non-overlapping reads (within the same clone) for calculating of s_{ij}-values (1). These values have been calculated for each movement of an ant. For n unvisited places corresponding to the reads of $m$bp each, the time complexities for selecting the next place from a place are $O(n)$ and $O(n \times m)$ for OVACO and LSACO respectively.

As mentioned before, LSACO has provided the results with accuracy between 99.9 % and 100 %. In a few cases, we have set the difference of reference sequence length and unknown sequence length by 1-10 bp. Due to this difference in length between the reference and the unknown sequences, we have got accuracy of 99.9 %, instead of 100 %, for some cases. LSACO reconstructs a sequence from the reads, by taking target length θ, where θ is the length of known reference sequence length. On the other hand, in the cases of identical length of reference sequence and unknown sequence, LSACO has resulted in 100 % accuracy. The mutual score between reference DNA and unknown DNA has been set as 99 %-99.9 %.

Table 8 Comparison of execution time in seconds

Serial Number	Data set	Execution time by LSACO	Execution time by OVACO	Execution time by PSO	Execution by BCO	Execution time by GA
1	Taenia1837	24	19	26	16	25
2	Taenia1899	28	21	29	18	20
3	HIV8929	2331	950	1290	808	2090
4	HIV8746	2137	660	896	561	1452
5	HIV8998	1226	740	1004	629	1628
6	HIV2591	34	19	26	16	42
7	HIV2583A	31	22	30	19	48
8	HIV2583B	41	20	27	17	44
9	INFL6126	504	262	356	223	576
10	INFL2259	25	14	38	27	42
11	INFL6125	350	262	356	223	577
12	INFL2261	28	18	24	15	40
13	HBV3212	49	36	49	31	79
14	HVB3958	104	76	110	65	167
15	HBV3182	51	38	51	32	84
16	BRCA3787	90	68	93	58	150
17	Aphid4175	185	92	125	78	202
18	pCPCbLCVA5352	401	166	225	141	365
19	Drosophila7439	1201	414	562	352	911
20	BRCA68879	3391	1304	1769	1108	2869
21	norvegicus52968	3379	1488	2019	1265	3274
22	BRCA127429	9350	1774	2407	1508	3903

4 Conclusions

In the algorithm LSACO, we have discriminated the reads from the same clone and different clones. Thus, it has been assured to reduce the chances of false alignment of the fragments in a target sequence. This fact has been explained with an example in Figs. 3 and 4. We have applied LSACO, OVACO, PSO, BCO, GA to 22 sequences obtained from DNA of different organisms. Due to lack of appropriate computational resources, we were forced to restrict the size of the sequences of length up to $\sim 10^5$bp. LSACO has been able to assemble the reads with 99.9 % to 100 % accuracy at the cost of a higher execution time compared to that of OVACO, PSO, BCO and GA. It has been observed that the method based on GA, has not been able to provide good results (accuracy 50 %-70 %) due to the fact that the crossover operator has changed sequence of reads to a larger extent than the other intelligent agents, like ants, bees and swarm particles. These intelligent agents change the sequence of reads a little extent by changing pheromone or nectar concentration, or by changing the velocity and position of the particles. Sequence of reads should be changed a little extent. The intelligent agents always guide the movement, and after the movement there is a feedback from the environment. Thus the method reorganizes or rejects the bad solutions. As there is a source of bad substrings in majority of solution space in GA, it may not provide a good solution. Thus GA got stuck with a wrong arrangement of fragments.

The results clearly depict that LSACO has provided good solutions. It is clear that if existing algorithms, including DSAPSO, CPSO, ABC, ACS and ML-ACO, other than GA, would use local score of the reads at the time of generating the solutions, it would provide more accurate solutions. The main disadvantage of LSACO is associated with higher time complexity. Our methodology checks all the local reads (unvisited places) when an ant jumps from a current place to another. If we can classify the read set or identify the region of reads in the target sequence, then there is a chance of thinning of local unexplored read set. Thus, it may reduce the time complexity sharply. If n tours can be performed and fitness of n solutions can be evaluated in parallel by using separate thread in a high performance computer, the execution time will be dropped significantly, although there will be a factor of $O(N^2)$ for evaluation of the score for solutions, N being number of base pairs in the sequence. The number of reads or the graph size does not have much effect on the time complexity of LSACO as

the computation time for the path exploration remains $O(n)$ with the number of reads or graph size being n. However, space complexity remains very high to maintain the relations among the reads. Here N is very high in comparison to n. Representation of A, C, T or G is done by character data type, where the size of the character is one byte and two bytes respectively in C++ and Java. There is a further chance to reduce the time complexity for evaluation of the scores of solutions by compressing the solution strings, if we can store substrings of 4^8 nucleotides into one byte and substrings of 4^{16} nucleotides into two bytes. In that case, there will be an additional time complexity for encoding and decoding, which will be negligible in the context of the solution's score evaluation. However, it incurs the problem of resolutions due to the compression of solutions. It may be difficult to compute the scores of solutions. Thus, reduction of time complexity in global or local score evaluation will be a new light to next generation sequencing by LSACO.

References

1. Garca S, Molina D, Lozano M, Herrera F (2009) A study on the use of non-parametric tests for analyzing the evolutionary algorithms behaviour: a case study on the CEC2005 Special Session on Real Parameter Optimization. Journal of Heuristcs 15:617–644
2. Indumathy R, Uma Maheswari S (2014) Solving DNA Sequence Assembly Using Particle Swarm Optimization With Inertia Weight and Constriction Factor. International Journal of Soft Computing and Artificial Intelligence 2(1):90–94
3. Verma RS, Singh V, Kumar S (2011) DNA Sequence Assembly using Particle Swarm Optimization. Int J Comput Appl 28(10):34–38
4. Fang S-C, Wang Y, Zhong J (2005) A Genetic Algorithm Approach to Solving DNA Fragment Assembly Problem. J Comput Theor Nanosci 2:1–7
5. Parsons RJ, Forrest S, Burks C (1995) Genetic algorithms, operators, and DNA fragment assembly. Mach Learn 21(1-2):11–33
6. Nebro AJ, Luque G, Luna F, Alba E (2008) DNA fragment assembly using a grid-based genetic algorithm. Comput Oper Res 35(9):2776–2790
7. Luque G, Alba E, Khuri S (2005) Parallel Computing for Bioinformatics and Computational Biology, WILEY, Chapter-12: Assembling DNA Fragments with a Distributed Genetic Algorithm
8. Karaboga D, Akay B (2009) A comparative study of Artificial Bee Colony algorithm. Appl Math Comput 25:108–132
9. Karaboga D, Ozturk C, Karaboga N, Gorkemli B (2012) Artificial bee colony programming for symbolic regression. Inf Sci 209:01–15
10. Firoz JS, Sohel Rahman M, Saha TK (2012) Bee Algorithms for Solving DNA Fragment Assembly Problem with Noisy and Noiseless data. GECCO '12 Proceedings 14th Annual Conference on Genetic and Evolutionary Computation. ACM, NY, pp 201–208
11. Ansorge WJ (2009) Next generation DNA sequencing techniques. New Biotechnol 25(4):167–260
12. Blazewicz J, Bryjaa M, Figlerowicz M, Gawrona P, Kasprzak M, Kirton E, Platt D, Przybytek J, Swiercz A, Szajkowski L (2009) Whole genome assembly from 454 sequencing output via modified DNA graph concept. Comput Biol Chem 33:224–230
13. Blum C, Valles MY, Blesa MJ (2008) An ant colony optimization algorithm for DNA sequencing by hybridization. Comput Oper Res 35:362–3635
14. Brun Y (2008) Solving NP-complete problems in the tile assembly model. Theor Comput Sci 395:31–46
15. Butler J, MacCallum I, Kleber M, Shlyakhter IA, Belmonte MK, Lander ES, Nusbaum C, Jaffe DB (2008) Allpaths: De novo assembly of whole-genome shotgun micro reads. Genome Res 18:810–820
16. Dorigo M, Maniezzo V, Colorni A (1996) Ant system: optimization by a colony of cooperating agents. IEEE Trans Systems Man Cybern Part B 26:29–41
17. Dorigo M, Stutzle T (2004) Ant Colony Optimization. MIT Press, London
18. Isakov O, Shomron N Deep sequencing data analysis: Challenges and solutions. Bioinformatics Trends and Methodologies, Intech, November 2011, ch-29:Deep Sequencing Data Analysis
19. Joshi N, Srivastava S, Kumar M, Kavalan J, Karandikar SK, Saraph A (2011) Parallelization of velvet, a de-novo genome sequence assembler. IEEE International Conference on High Performance Computing
20. Kurniawan TB, Ibrahim Z, Saaid MFM, Yahya A (2008) Implementation of ant system for DNA sequence optimization. NANOSciTech, Shah Alam
21. Ma X, Lombardi F (2008) Combinatorial optimization problem in designing DNA self-assembly tile sets. 2008 IEEE International Workshop on Design and Test of Nano Devices, Circuits and Systems, pp 73–76
22. Medvedev P, Georgiou K, Myers G, Brudno M (2007) Computability models of sequence assembly. Workshop on Algorithms in Bioinformatics, Philadelphia, 289–301
23. Meksangsouy P, Chaiyaratana N (2003) DNA fragment assembly using an ant colony system algorithm. Proceedings Evolutionary Computation. CEC '03 3:1756–1763
24. Miller JR, Koren S, Sutton G (2010) Assembly algorithms for next generation sequencing data. Genomics 95:315–327
25. Myers G (1999) Whole-genome dna sequencing. Comput Sci Eng 1:33–43
26. Myllykangas S, Buenrostro J, Ji HP (2012) Overview of sequencing technology platforms. Bioinformatics for High Throughput Sequencing, 11–25
27. Quail MA, Smith M, Coupland P, Otto TD, Harris SR, Connor TR, Bertoni A, Swerdlow HP, Yong G (2012) A tale of three next generation sequencing platforms: comparison of ion torrent, pacific biosciences and illumina Miseq sequencers. BMC Genom 13:1471–2164
28. Scheibye-Alsing K, Hoffmann S, Frankel AM, Jensen P, Stadler PF (2009) Sequence assembly. Comput Biol Che:33
29. Stupar M, Vidovi V, Luka D (2011) Functions of human noncoding DNA sequences. Arch Oncol 19:3–4
30. Treangen TJ, Salzberg SL (2011) Repetitive DNA and next-generation sequencing: computational challenges and solutions. Nat Rev Genet 113(1):36–46
31. Wei L-T, Yang C-B, Ann H-Y, Peng Y-H (2007) Ant colony optimization algorithms for sequence assembly with haplotyping. 6th Conference on Information Technology and Applications in Outlying Islands, Yunlin, Taiwan, 260–268
32. Zerbino DR, Velvet EB (2008) Algorithms for de novo short read assembly using de Bruijn graphs. Genome Res 1:821–829
33. Fullwood MJ, Wei C-L, Liu ET (2009) Next-generation DNA sequencing of paired-end tags (PET) for transcriptome and genome analyses. Genome Res 19:521–532

Susobhan Baidya is an Assistant Professor of the Heritage Institute of Technology, Kolkata, India. He has completed Bachelor of Technology in Computer Science and Engineering and Master of Technology in Computer Science and Engineering in the year 2005 and 2007 respectively, from Calcutta University, India. He has published two research articles in conference proceedings. His research interest includes bioinformatics, computational biology and soft computing.

Rajat K. De is a Professor of the Indian Statistical Institute, Kolkata, India. He completed his Bachelor of Technology in Computer Science & Engineering, and Master of Computer Science and Engineering in the years 1991 and 1993, from Calcutta University and Jadavpur University, India, respectively. He obtained his Ph.D. degree from the Indian Statistical Institute, India, in 2000. Dr. De was a Distinguished Postdoctoral Fellow at the Whitaker Biomedical Engineering Institute, the Johns Hopkins University, USA, during 2002-2003. He has about 70 research articles published in international journals, conference proceedings and in edited books to his credit. His research interest includes bioinformatics, computational biology, systems biology, pattern recognition and soft computing.