Egyptian Computer Science Journal, ECS, Vol. 37 No. 3, May 2013 ISSN-1110-2586

A New Algorithm for Overlap Graph Reduction

Amany Abdel Aziz El_Sebaai, Mohamed Abdel Fatah Belal, Hala Abdel Galil El_Sayed

Computer Science Department, Helwan University, Egypt
amany.zizo@gmail.com

Abstract

The DNA sequence assembly is very important process because it allows the DNA sequence to be scanned and used. The Overlap-Layout-Consensus is the most used technique to assemble DNA sequence. The most drawback faces this technique is the immense size of the overlap graph. There are some algorithms tried to resolve this drawback. During this paper a new algorithm is introduced to get rid of the unnecessary edges keeping the most effective weighted ones for the next phases that improve the efficiency and additionally save the time. The proposed algorithm is tested on protein from Wolbachia that is a kind of bacterium bacteria employed in testing the new assemblers. The results are compared with best buddy algorithm that utilized by Celera assembler. The results shows that the proposed algorithm is additional efficient than best buddy algorithm however it takes a extended execution time in sake of this improved efficiency. It shows additionally that it'll be more practical with the real data came from sequencer machine.

Keywords: *Graphs, DNA sequence assembly, DNA fragmentation, and overlap-layout-consensus technique.*

1. Introduction

With the boom in the field of bioinformatics [1], several new research areas rise. One of the most vital research issues is DNA assembly that makes the DNA clear for biologists. It provides deep insight into the fact of several diseases that ends up in new medicine. However it faces some issues because of the type of data and also the needs of the biologists. The big size of those data poses a deep challenge on economical processing and memory requirements to the most existing assembly systems. The aim of this work is to propose an algorithm improves the sequence assembly process, make it doable to run on any kind of computers.

The DNA sequence assembly process is handled by many alternative techniques like shortest superstring, sequencing by hybridization, hierarchical assembly and overlap-layout-consensus (OLC) technique, for additional details[2][3]. This paper works on a modification on the OLC technique. OLC contains 3 phases overlap, layout, and consensus. The modification was created on the overlap graph and also the drawback associated with its vast size. First, it had been done by Myer who uses many reduction steps to simplify the overlap graph, and generate a group of Unitigs [4, 5]. However his algorithmic rule is computationally high-priced therefore the Current assembly program finds the “best” overlap on each end of every fragment—its “best buddy” [6]. the best buddy algorithm depends on one main rule that is if the longest overlap with fragment A is fragment B and also the longest overlap with fragment B is fragment A, then fragments A and B are best buddies [7]. However this rule might ignore some overlap values that appear to be necessary to shaping the DNA sequence. So this paper tries to propose an algorithm for reducing the scale of overlap graph keeping the larger overlap weights on it for the following assembly phases as mentioned within the following parts.

This paper is divided as following. Section 2 describes the DNA fragments assembly and its two components laboratory and computer. Section 3 explains the overlap- layout-consensus technique and its three phases. Section 4 exposes to the previous work associated with overlap graph reduction. It describes Myers's algorithm and best buddy algorithm. Section 5 shows the proposed work and its modules. Section 6 introduces a new algorithm to resolve the problem and shows its complexity. The experiments and also the obtained results are mentioned in section 7. Section 8 concludes the proposed work and the future scope of it.

2. The DNA Sequence Assembly Problem

DNA consists of two strands, each of which contains nucleotides: adenine (A), cytosine (C), guanine (G) and thymine (T). (Technically, there are other elements in a DNA strand like phosphates.). The nucleotides in every strand are connected together serial. The two strands of the DNA are twisted along into the famous helix structure. Moreover, every nucleotide in a strand is connected to a complementary nucleotide within the alternative strand, where A is paired with T and C is paired with G. Thus, every strand in a DNA fully determines the opposite [8].

The fragment assembly is split in two parts: one is the laboratory section of cloning, fragmentation and reading and also the second one is the computer section of collecting the fragments together. The subsequent sections explain them in details.

A. Laboratory Section

Under favorable conditions, current sequencing technologies allow reading up to 1,000 nucleotides per sequenced fragment, and an average of five hundred to a thousand nucleotides throughout an experiment. Therefore the sequencing methodology faces a serious problem in a operative large-scale sequencing program, see automatic sequencing strategy for additional details [9].

The most approach has primarily been used is shotgun sequencing wherever DNA is choppy at random into various little segments items of best sequencing size (~1,000 base pairs), that are sequenced by the chain termination methodology to get reads. A high cutting frequency (one site per 200–250 bp) restriction enzyme could also be used, under conditions of partial digestion (10–20 percent) in order to get 1,000- to 2,000-bp fragments. Now, the fragments size is compatible to that of the sequencing system. Then, the biologists insert them into a cloning vector, plasmid or virus (usually Escherichia coli) to produce a defense reaction against invading viruses. Then it propagated in host cells. The clone cell lines containing another recombinant vector with a similar inserted DNA fragment are then isolated.

The random fragments (also known as inserts) are sometimes organized into many libraries consisting of fragments of similar size. As mentioned pervious, current sequencing technologies can only "read" between five hundred and a thousand base pairs of DNA, and, therefore, the center of the fragments remains unsequenced. This ends up in pairs of reads (also known as mate-pairs), obtained from opposite ends of a same fragment, that are naturally related. This procedure is shown in figure 1. The resulting sequences from this step contain many errors like insertion, deletion that should be resolved in assembly step. Assembly algorithms tolerate nearly 2% of error rate.

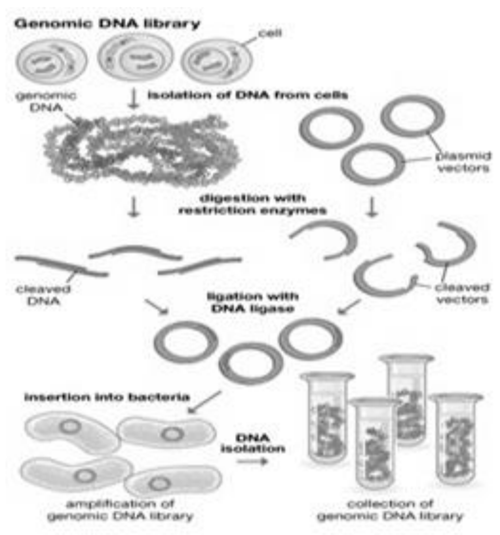


Figure (1): sequencing Method

B. Computer Section

Now, the role of the computer involves assemble the initial DNA. For instance, take into account the subsequent two shotgun sequences:

Original sequence

AGCATGCTGCAGTCATGCTTAGGCTA

First shotgun sequence

```
AGCAT-----------------------------------------
----------GCTGCAGTCATGCTTAGGCTA
```

Second shotgun sequence

```
AGCATGCTGCAGTCATGCTT-------------
-------------------------------------------AGGCTA
```

Rebuilding

AGCATGCTGCAGTCATGCTTAGGCTA

In this very simplified example, none of the fragments cover the complete length of the initial sequence, however the four reads may be assembled into the initial sequence by the overlap of their ends to align and get them organized. In different words, it's reconstructing the genome from the shotgun fragments based on sequence similarity. This problem will be further divided into phases as will be mentioned latter. The most used assembly technique is overlap-layout-consensus (OLC), the subsequent section discuss it in brief.

3. Overlap – Layout – Consensus Technique

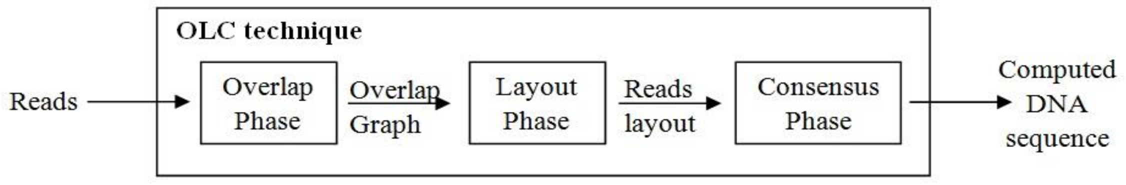


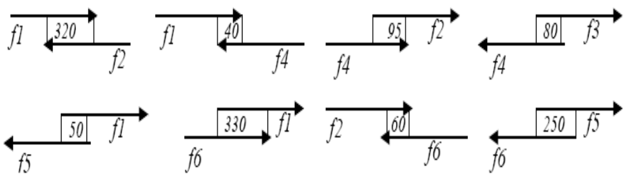
Figure (2): OLC technique

As shown in figure 2, this technique is divided into three distinct phases:

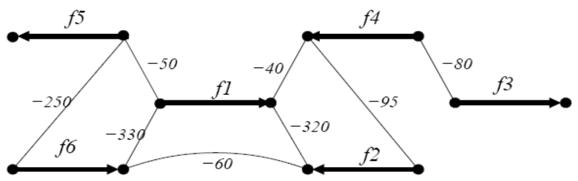
3.1 Overlap Phase

Overlap Graph OG is a directed weighted graph where each node corresponds to an element in the set of all fragments F and two nodes (a, b) are connected if there exists a suffix of which is a prefix of b [10].Find all the overlaps between the fragments and represents it on graph depending on the overlap length.

Assume there are given 6 fragments with the following overlap length: [11]



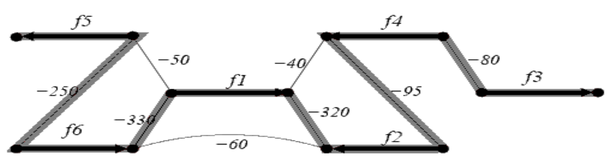
Here, the last 320 bases of fragment f1 align to the first 320 bases of the reverse complement f2 of f2, whereas f1 and f5 overlap in the first 50 bases of each.
The following overlap graph OG is obtained:



3.2 Layout Phase

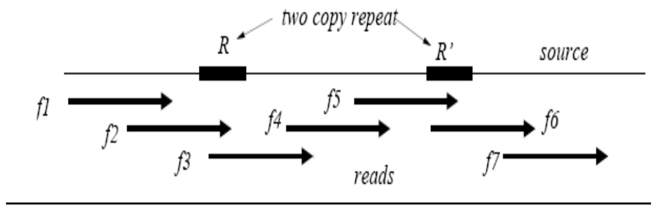
Given the overlap graph, determines a consistent layout of the fragments. (Order of fragments, concatenate them in what is called contigs then super-contigs). They attempt to maximize the weight of the resulting sub-graph, given a set of weights corresponding to the quality of the overlaps.

A simple heuristic is to select a spanning forest of the overlap graph OG that contains all fragments.

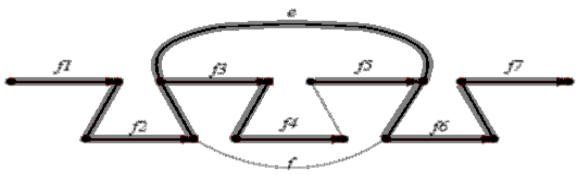


Repeats and the layout phase

The main difficulty in the layout phase is to distinguish between true overlaps and repeat-induced overlaps.



Then the spanning tree will be:



The edge e or f does not reflect the true ordering of the reads.

The layout phase proceeds in two stages:

1. Unitigging: First, all uniquely assembled contigs are produced, these are called unitigs. Unitigs represent sections of genome that can be unambiguously resolved [12].
2. Repeat resolution: Then, one attempts to reconstruct the repetitive sequence that lies between such unitigs.

3.3 Consensus phase

Constructing multiple alignments of all fragments in a given contig to determine the most likely DNA sequence (the consensus sequence) for it and try to correct errors in the obtained sequence.

```
R1          ACGCTCCAACCGCTAATACG
R2                  ATCGCTAATCCACGCCCGCCCCGC
R3        AAAC-CTCCAACCG
R4                          TGCGCGCCCGCCCCGAAACCGC
Consensus AAAC-CTCCAACCGCTAATGCGCGCCCGCCCCGAAACCGC
```

The exiting techniques compare all fragments for overlap detection by disturbed processing with an outsized scale of high performance computers. Celera genomics in human genome project reported that "Computing the set of all overlaps took roughly 10,000 central processing unit hours with a suite of four processor Alpha SMPs with 4 gigabytes of RAM. This took four to five days in period of time with 40 such machines running in parallel". Obviously, such computational support continues to be too costly [13]. This paper tries to resolve this drawback by propose a new algorithm aims to reduce the massive size of the overlap graph.

4. Previous Used Algorithms

The previous work in this point not focused on the reduction of graph only, however it's a part of the unitig creation. However it's going to ignore some overlap values which will be necessary to shaping the original DNA. Additionally it merges the vertices together to form unitig, losing some fragments from the search space when attempting to induce the final path, from reduction view only. In alternative words it's the modeling of fragments in layout. The proposed algorithm don't care about the unitig creation, it's only care about the reduction of overlap graph for further use within the preceding phases. These two algorithms are Myers and best buddy. The results of the proposed algorithm are compared with best buddy algorithm from the purpose of graph reduction only.

1.1 Myers's Algorithm

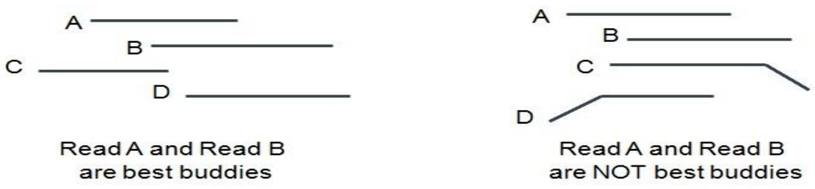
These reduction steps are:

1. Removal of containment edges. Reads completely contained within other reads in the input are removed from the graph.
2. Transitive reduction. For any set of three reads (A, B, and C), if the overlap between A and C can be inferred from the overlaps between reads A and B, and B and C, this overlap (i.e. the edge corresponding to this overlap) is removed from the graph.
3. Unique-join collapsing. Every simple path in the graph (paths that contain no branches, i.e. all the nodes have in and out-degrees equal to 1) is collapsed into a single vertex. Each such vertex represents an individual unitig [14].

Preceding algorithm is computationally too cost especially in step two. So the Current assembler finds the "best" overlap on each end of each read—its "best buddy" [6]. One of the most important assembler that follows this algorithm is Celera.

1.2 Best Buddy Algorithm

If the longest overlap with fragment A is fragment B and the longest overlap with fragment B is fragment A, then fragments A and B are best buddies [7].


A
B
C
D
Read A and Read B are best buddies
A
B
C
D
Read A and Read B are NOT best buddies

Unitigs are chains of mutually unique best buddies— adjacent reads are best buddies of each other and of no other read [6].

This takes time and space linear in the number of fragments. In rare cases results are different from graph reduction in two points: Low coverage regions and High fidelity repeat copies [6].

5. The Enhanced Technique by Proposed Overlap Graph Reduction

The OLC technique is going to be as the next figure after adding the proposed overlap Graph reduction algorithm. The first three modules in the system are developed to test it and compare its results.

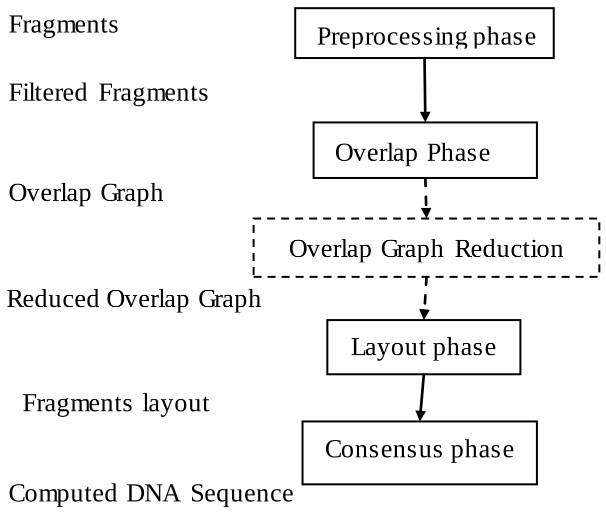


Figure (3): OLC with proposed Overlap Graph Reduction

Phase1: preprocessing phase is split into 2 steps, normalization step the data is normalized by changing the four characters contained in DNA sequence (A, G, C, T) to numbers (0, 1, 2, 3). Second step is that the filtration according quality values of sequences. The method would be much easier and lower cost since fewer reads would be needed. Phred quality values became a standard inside the sequencing community (P.Richerich, 1998). These quality values are logarithmic and vary from zero to ninety nine wherever high value indicates high quality. They're outlined as follows:

$$q = -10 * \log P$$

Where q is the quality and P is the estimated error probability of a base-call. Phred quality values have a large vary. For instance, the quality 10 implies that the probability of a base being wrong is 1 /10 or having 90th accuracy. Biologists specify an exact threshold value to get an exact data values. The values smaller than this threshold are excluded. In our program, it's an elective step if quality values for fragments are available [12].

Phase2: overlap phase creates the overlap graph by doing all comparisons between every 2 fragments. The semi-global alignment is employed to find the overlap between every 2 fragments in our dataset; therefore the overlap graph is formed. The Needleman-wunsch algorithm from dynamic programming is employed. The fundamental Needleman-Wunsch algorithm is changed to calculate the optimum semi-global alignment where start and end gaps are neglected. Such an alignment is helpful for potential overlap detection [15].

Phase3: overlap graph reduction removes the low weighted edges while not losing necessary data. During this phase a new algorithm is proposed which will save pc resources and cut back the computational time.

6. The Proposed Algorithm

This paper proposes an algorithm to efficiently solve the matter of big overlap graph size. The problem under study, as represented and surveyed within the previous sections, will be solved by the proposed algorithm. It permits us to complete the subsequent assembly phases, layout and consensus phases, employing a reduced overlap graph. As a result, it reduces the processing time and increase the system efficiency.
The proposed algorithm takes overlap graph as input and returns the reduced one as output. This algorithm will be represented in four main steps.

Step 1: search for the max overlap weight value among the predecessors of every vertex in overlap graph, adding their edges to reduced graph.

Step 2: for each edge in reduced graph, see if there is another edge in overlap graph (among the successors of the head vertex of that edge) that are greater than or equal the overlap weight of this edge. If found then, add them to reduced graph.

Step 3: if there is a vertex in reduced graph that have not any successors, then search in their successors from overlap graph and get an edge with the maximum overlap weight. Then add this edge between this vertex and the selected successor in the reduced graph.

Step 4: if there is a double edge in reduced graph, check for their overlap value (weight), and discard the smallest one.

The pseudo code for the algorithm is as following:

```
Notations:
OVG → Overlap Graph
RG  → Reduced Overlap Graph
V → vertices
E → Edges
W → weight
P (u) → predecessor of u
S (u) → successor of u
Input: OVG
Initialize:
For every vertex v ϵ V [RG] Do
       π[v]← null
Procedure:
1. For each vertex u ϵ V [OVG] Do
2.     e (u, v) ← max {w (v, u): v ϵ P (u)}
3.     RG = RG υ e (u, v)
4. For every e (v, u) ϵ E [RG] Do
5.     Find e (v, z) ϵ OVG ← max {w (v, z): z ϵ S (v)}
6.     If (w (v, z)> w (v, u) then
7.             RG= RG υ e (v, z)
8. For every vertex (u ϵ RG: S (u) =0) Do
9.     e (v, u)← max {w (v, u): v ϵ P (u)}
10.    RG = RG υ e (v, u)
11. For every e (u, v) ϵ E [RG] Do
12.    If there exist e (v, u) then
13.            If (w (u, v)> w (v, u) then
14.                   e (v, u)←null
Output: RG
```

Assume that Overlap Graph, OVG = (V, E) and Reduced Graph, RG = (V', E'). Therefore in line with the pseudo code of the algorithm the complexity may be calculated as following. The first loop in lines one to three takes O (V) that handles the first step in our algorithm. The second loop from line four to six takes O (E'). The third loop in lines eight to ten takes O (V') however this step might not be executed throughout the real running time. The fourth loop in lines eleven to fourteen takes O (E') additionally as the second loop although the no of edges could also be totally different at running. Therefore the sum of those quality values are going to be O (E') because it covers other values where E' the no of edges in reduced graph.

7. Results

Experiment Data

Wolbachia is a type of bacteria and initial microscopical organism monitored for horizontal gen transfer to multicellular organism. It has attracted attention of many researchers as a result of it would be because in revealing the evolution of virus [13]. The Wolbachia genome sequence is too massive however a smaller test Hypothetical protein sequence was used. Therefore it's possible to use real genome fragments taking into the account the computation time.

Table (1): further detail about Wolbachia Genome data

Wolbachia endosymbiont of Drosophila melanogaster		
Name	complete genome	Protein sequence
NCBI Reference Sequence: (range)	NC_002978.6	NC_002978.6 (497,224..505,755)
Locus Tag	--	WD0513
Total length	1,267,782 bp	8,531 bp

The pervious table shows further details about Wolbachia genome data and the protein sequence used in the experiment. First, using a hypothetical protein from Wolbachia, its length is 8,531 bp.

Data acquisition

A 10 copies are made, each of which were split randomly to fragments of length 400bp ± δ where δ=100 bp. Too small fragments are removed from library, the fragments at the end of every copy must be at least greater than or equal 100 bp. There are 212 fragments approximately. the experiment is made twice. First, without representing any expected sequencing errors. Then some errors such as insertions, deletion and change of nucleotides are added. The total number of introduced errors was set to 2% of total number of nucleotides in fragment. These errors were selected randomly. Next, with the probability of 1/3 in such a place a randomly chosen nucleotide was inserted, with the probability of 1/3 selected nucleotide was deleted, and with the probability of 1/3 the selected nucleotide was exchanged to another one randomly. Then the same experiment is repeated twice, without error and with 2% error, but increase the number of copies to 20 copies. The results are shown in the following table.

Table (2): the obtained results

		Without errors	with 2% errors	Without errors	with 2% errors
No of copies		10		20	
No of Fragments		212		424	
Fragment Length		200 bp-500 bp			
OVG Creating time (in sec.)		308	356	1366	1293
Using proposed algorithm (in sec.)		8	13	31	34
Using best buddy algorithm (in sec.)		0.6	0.6	1.5	1.2
No of Edges	original	44944		179776	
	proposed algorithm	340	302	1058	611
	Best buddy algorithm	274	217	1005	496

The overlap graph size

The first discovered point from these experimental results is the size of overlap graph that shrunk when running the proposed algorithm. The previous tables show the amount of edges in graph before and after reducing phase. The size of overlap graph is reduced by nearly 98% that provide us a lot of benefits:

1. It makes it quicker to complete the subsequent assembly phases.
2. These redundant data was slowing down the processing time inflicting a memory space drawback in the assembly system.
3. Make it simple to search out a path through the fragments to shape the final DNA sequence throughout a reduced time comparing to finding the path before reducing the graph size since the search space is decreased.
4. Another powerful purpose gained by our algorithm focused in never losing any vertex from graph, in other words all fragments are kept during reduction phase.

From the above results showed that the proposed algorithm works well when it applied to an extremely large real data set. So the proposed algorithm can perform better when using it for the real data came from sequencer machine graphs taking into account the running time.

By adding 2% error rate to the data to simulate the errors found in real data, the number of edges in reduced overlap graph affected by nearly 33% which considered being a great challenge to the DNA sequencing machines.

The Reduced Overlap Graph Robustness

The robustness of the proposed algorithm is measured through a comparison between proposed algorithm and best buddy algorithm. The weights vary of the remained edges in the overlap graph after reduction phase controls this measurement. When the edges were remained that holds the high weighted overlap the results were more efficient.

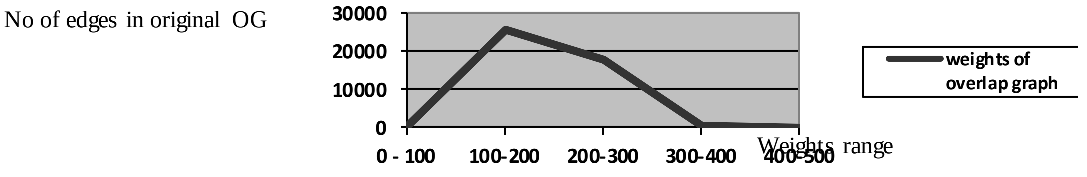


Figure (4): the weights of overlap graph for 10 copies Experiment

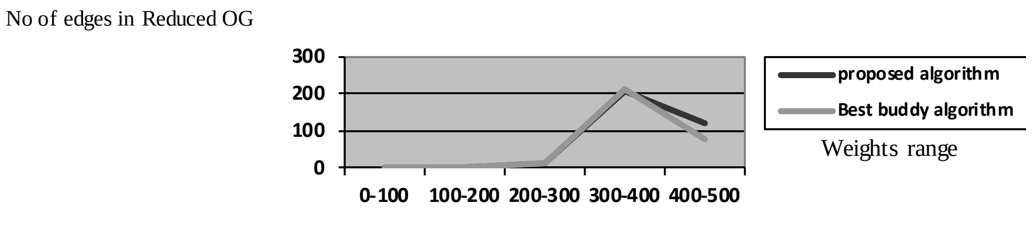


Figure (5): the remaining weights in reduced graphs in proposed algorithm and best buddy algorithm 10 copies Experiment

The execution time

The execution time for overlap graph reduction was measured by the proposed algorithm compared with best buddy algorithm. the proposed algorithm took an extended time than best buddy algorithm however in exchange for improved efficiency as shown within the previous part.

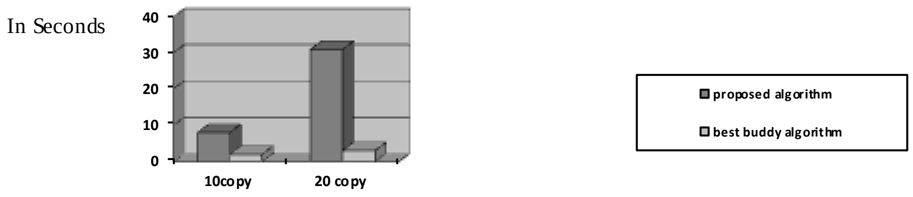


Figure (6): the execution time used for the samples

8. Conclusions and Future Work

In this paper, a new algorithm for Graph reduction in OLC was introduced to resolve the matter associated with large size of OG. There are 2 algorithms supported by the most assemblers Myer's algorithm and best buddy algorithm. The proposed algorithm takes OG as input and come back the reduced on as output. It's divided into four main steps.

A lot of benefits were gained from adding this phase. It made it faster to complete the subsequent assembly phases. Additionally these redundant data was slowing down the processing time inflicting a memory space drawback within the assembly system. It absolutely was simple to search out a path through the fragments to shape the original DNA sequence during a reduced time to finding the path before reducing the graph size since the search space was reduced. Another powerful advantage was preserving the graph vertices, in alternative words fragments were kept during the reduction phase.

A small Hypothetical protein sequence from Wolbachia genome was employed to test the proposed algorithm; it was fragmented to simulate the real data. The experimental results were done and compared with best buddy algorithm. The results showed that the proposed algorithm was additional efficient than best buddy algorithm since it kept the edges that hold the high weighted values. unfortunately the proposed algorithm took a extended execution time than best buddy algorithm in sake of this improved efficiency. It showed additionally that the proposed algorithm was more practical with a bigger data set that proved that it might perform better if used for the real data came from sequencer machine.

This work is expanded by finishing the remaining phases within the OLC technique. the most purpose that must be addressed is that the second drawback that faces the layout phase. it's the repeat-induced overlaps that it's difficult to distinguish between true overlaps and false one. It needs extra analysis, deep understanding and additional time which can be available within the future. additionally increasing the genes that's used to test the new technique or even use a real data sequence. after finishing the OLC technique phases, it is compared with the other assembly techniques.

References

[1] Arthur M. Lesk, "Introduction to Bioinformatics", University of Cambridge, New York, USA, 2002, Chapter1.

[2] Neil C. Jones and Pavel A. Pevzner.. "An Introduction to Bioinformatics Algorithms", MIT Press, 2004.

[3] Davidson College, north of Charlotte in Davidson, N.C., "Sequencing Whole Genomes"

[4] Myers EW: "Toward Simplifying and Accurately Formulating Fragment Assembly", J Comp Bio, 1995.,

[5] Myers EW: "The fragment assembly string graph", Bioinformatics, 2005.

[6] Michael Schatz,"Celera Assembler-Theory and Practice", University of Hawaii, August 13, 2006.

[7] Megan Smedinghoff, "Improving the Draft Assembly of the Horse Genome", Applied Mathematics and Scientific Computation, University of Maryland - College Park, 2009.

[8] Mihai Pop, Steven L. Salzberg, Martin Shumway, "Genome Sequence Assembly: Algorithms and Issues", the Institute for Genomic Research, July 2002.

[9] Frédéric Dardel & François Képès translated by Noah Hardy, "Bioinformatics Genomics and post-genomics", 2006, By John Wiley & Sons Ltd, Chapter1.

[10] Hershel Safer, "Introduction to computational Molecular Biology", March 1998.

[11] Mihai Pop, "Shotgun Sequence Assembly", the Institute for Genomic Research (TIGR), Rockville USA, Dec 2003.

[12] Martti T. Tammi ,"The Principles of Shotgun Sequencing and Automated Fragment Assembly", Center for Genomics and Bioinformatics, Karolinska Institutet, Stockholm, Sweden, April 2003.

[13] Satoko Kikuchi & Goutam Chakraborty, "Heuristically tuned GA to solve Genome fragment Assembly Problem", Graduate School of software and Information science, Iwate prefectural University, Japan, July 16, 2006.

[14] Daniel D Sommer1, Arthur L Delcher1, Steven L Salzberg and Mihai Pop, "Minimus: a fast, lightweight genome assembler", February 2007

[15] Setubal J., Meidanis,J., "Introduction to Molecular Biology", Chapter 3.