ANALYSIS PAPER

INTRODUCTION

Objective:- The main objective of this project is to build a pipeline to perform a preliminary genome assembly of the source organism Listeria monocytogenes taking input as short read DNA sequence, assembling a transcriptome from RNA raw reads , annotating the genes within the genome and finally predicting the function for those genes.

Scope:-

Whole genomes can now be sequenced affordably thanks to the invention of next-generation sequencing. However, due to short read lengths, incomplete data, repetitive sections, polymorphisms, and sequencing errors, de novo genome assembly continue to be difficult. Scientists have been motivated to decipher the genetic information of other non-model animals because the whole-genome shotgun sequencing of some model organisms is complete. Improvements in the correctness and completeness of genomes can be somewhat facilitated by the preparation of the paired-end and mate-pair libraries as well as improving the depth of sequencing coverage. It is always necessary to put in significant additional work to achieve a full genome, such as Sanger sequencing and customized assembly methods. Whole genome sequence (WGS) analyses have been utilized to support the findings of the listeriosis outbreak investigations. As source of organism, Listeria monocytogenes is one of the common food borne pathogen, Gram positive bacteria that causes listeriosis in humans. The organism is found in moist environments, soil, water, decaying vegetation and animals. When people eat contaminated food, they develop listeriosis.

Here we have utilized NGS data to commence with the genome assembly and chose Illumina sequence reads as it is comprehensive, rapid and the most accurate.

Input data :- The raw reads of Listeria monocytogenes DNA(SRR14252414) for genome assembly and RNA (SRR9691137) for transcriptome assembly were extracted from NCBI SRA(Sequence Read archive). For genome assembly, the scope of the sample was a monoisolate, the spots reads were 1011776, reads read 2023552 and the bacterial sample was isolated from animal sources. For transcriptome assembly, the scope of the sample was multispecies and the bacterial sample tested under stimulated food processing conditions.

All the assemblies were sequentially run to finally predict the protein on Discovery high performance computing terminal by submitting batch jobs. The

scripts were modified and orderly executed by running sbatch scripts which displayed desired output files and accomplished all the tasks. The jobs were submitted in a scratch directory which is more performant in accessing the data. The scripts were run sequentially to run a pipeline and all the desired scripts, results, folder ,logs were made to run the scripts without an error. All the folders were shifted back to the home directory after running it from scratch directory.

RESULTS

Tools ,resources and methodology:-

1. **Genome assembly**

   To retrieve the sequence reads from NGS of the source organism, used fasterq-dump and created a shell script getNGS.sh .After the data extraction, the reads were quality trimmed using **trimmomatic-0.39-2 version** which is a Java based quality trimmer that uses sliding window to direct the quality scores below a specific threshold and it also removes any extra adapters that get attached to beginning or end of cDNA fragments. Created a shell script trim.sh. In the script, specified trimmomatic paired end reads, threads as one which indicate server threads, phred33 indicating the quality encoding method. Thereafter, paired and unpaired output files are directed to get the same number of reads in the left and right file and low quality reads are eliminated. To remove the number of bases from the beginning used headcrop. Illuminaclip specifies the mismatch that are allowed in adapter match.The leading and trailing determine the minimum quality for trimming at the start and end of reads.Sliding window checks for the minimum average quality for bases in that window and the minimum length of reads to be kept.

   After read trimming, the data were passed into the first stage of pipeline of genome assembly which is the **SPAdes genome assembler v3.13.1**. The Spades assemble the genome on the basis of overlapping reads to get the quality trimmed reads. To run the assembly created a shell script runSpades.sh and assembler will assemble the contigs, scaffolds and create an assembly graph in GFA format. After the assembly, runned a shell script runQuast.sh which checks the quality of assembly by computing various metrices N50,L50,NG50,L50,genome fraction. Finally run sbatch_assembleGenome.sh to run the whole program and assemble the genome which will create the several output ,log files and plots.The assembly is further analysed on the basis of result.txt and reports.

2. **Assembling a transcriptome from RNA raw reads**:-

   Before taking an input RNA rawreads, builded a genome database using GSNAP which use ListeriaGmap database to perform the alignment of RNA seq reads. Built a script ListeriBuild.sh for the alignment. Used scaffolds.fasta to build the database. Following which processed input data RNA raw reads of source organism by retrieving it from NCBI SRA using fasterq-dump command. Thereafter, used **trimmomatic** tool to trim the low quality reads, clean up the reads and removing adapter remnants. This generated paired and unpaired reads. After successful trimming, the reads were assembled using **Trinity** as it reconstructs the full length transcriptome and is better than other denovo assemblers. Created a shell script trinityDeNovo1.sh to list all the left and right reads and run them in a comma separated lists. After which , shell script runtrinityDeNovo.sh was run on CPU mode 4 and the assembly commenced with working on reconstructing the transcripts using three independent software modules Inchworm, Chrysalis and Butterfly sequentially. The jellyfish extract and count the kmers from the reads, inchworm assembles initial contigs by greedily extending sequences with most abundant kmers, chrysalis clusters the overlapping contig ,build de Bruijn graph for each cluster and partitions reads between the clusters, Butterfly resolves alternatively spliced and paralogous transcripts independently for each cluster.This will result in output files in trinity denovo that will be generated in results. To analyse the trinity results created a shell script analyzetrinityDenovo.sh which stored the standard output to results as text file counting the number of trinity genes,transcripts and generate a statistics of all the transcript contigs and stats based on longest isoform per gene. Finally created a shell script sbatch_trinityDenovo.sh to combine all the shell scripts and create the desired output files and directories

3. **Annotating the genes within the genome:**-

   To annotate genes within the genome used **blast**, **Transdecoder** and **hmmscan** to identify protein sequences similarity and protein predictions using input data .Input data containing Trinity de novo data of Listeria genome from previous module was extracted and proteins were predicted using Transdecoder which follows a 4 step process:

1. Transdecoder.long orf - which searches the longest open reading frame and translate to amino acid sequences.
2. Blastp - which identifies proteins similarity by aligning the long orf to guide the prediction expression process

3. Hmmscan - which uses Hidden markov model to scan and protein domains and guide the prediction process
4. Transdecoder.predict - which takes in ORF, BLAST Output and protein domain information to filter protein predictions to produce a protein fasta file *.pep

   To begin with the protein prediction , the directories, files, results and scripts were created The bash scripts were run sequentially (longOrfs_args.sh, blastPep_args.sh,pfamScan_args.sh,predictProteins_args.sh) to extend command line arguments to make the code extensible in provided path.

   The scripts were hard coded using $preceded by the number of arguments.

- longOrf_args.sh - finds the longest ORF in data taking trinity fasta as an input and producing an output in results in trinity_de_novo_transdecoder_dir
- blastPep.sh - aligns the proteins which are found in longest orf to swissprot database
- pfamScan_args.sh - used hmmscan tool to take query protein sequences found in the ORF's and searched them against HMM database and output the ranked list of significant matches to the sequence.
- predictProteins_args.sh - The transdecoder predicts the coding regions from the ORFS according to the ORF retention criteria which is as follows:

1. Minimum length of ORF in a transcript sequence
2. A log likelihood score
3. Longest orf identified if ORF is fully encapsulated by coordinates of another candidate orf
4. PSSM (Position specific scoring matrix) computed to refine start codon prediction

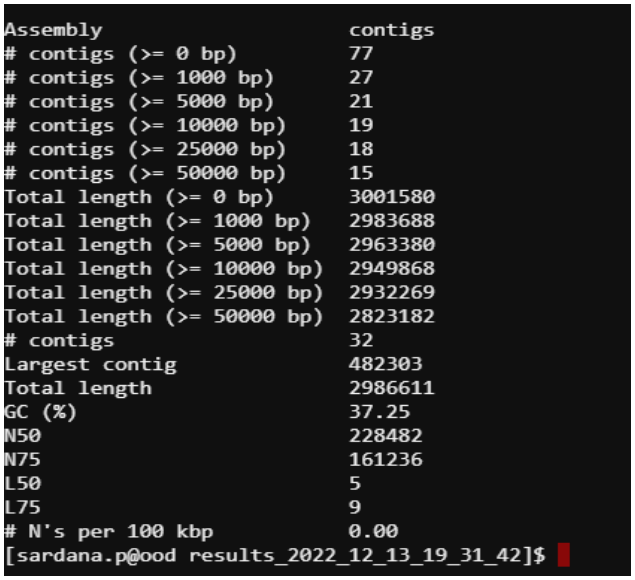
- Finally run the command sbatch_transdecoder.sh script to perform BLAST and Transdecoder.

5. **Predicting the function for those genes: -** The aligned predicted proteins were further used as an input data to align them to Swissprot using BLAST.
   Used KEGG (Kyoto Encyclopedia of genes and genomes) and gene ontology to retrieve the data for set of transcripts and combine the data into single annotation file.Associated SwissProt protein IDs with the transcripts to understand the function of the genes. To predict the function used **KEGG API** which is a programmatic way to access the data from a service like KEGG. In order to convert Swissprot to KEGG for protein and write the results in a tabular file, created a function getUniprotfromBlast to return the Uniprot ID from the blast line if evalue is below the threshold.If the evalue is above threshold, it returns

False. To return list of all kegg genes connected with Uniprot id created a function getKeggGene. After having uniprot id , to get kegg id we need to pass it to Kegg API. To execute KEGG api command used requests library. In order to decode the data into normal text used decode() string method and encoding type (UTF-8). To iterate over more than one line of output used result.iter_lines() and returned kegg genes. Similarly created a function getKeggOrthology() to return KeggOrthologyID using kegg genes and modeled it on getKegggenes() function. Created a function load kegg pathways to return the dictionary of pathID as keys and pathway description as values. Finally created a function addkeggPathways() to tie all the functions together to establish an output file in results as alignpredicted.txt.

INTERPRETATION:-

The genome assembly of the reference genome Listeria monoytogenes was performed using Spades assembler and for analysis used Quast. Quast compute the metrices and evaluates a genome assembly. The output file report.txt was analysed indicating the length of contigs , largest contig ,GC%, N50,L75,L50.



Assembly	contigs
# contigs (>= 0 bp)	77
# contigs (>= 1000 bp)	27
# contigs (>= 5000 bp)	21
# contigs (>= 10000 bp)	19
# contigs (>= 25000 bp)	18
# contigs (>= 50000 bp)	15
Total length (>= 0 bp)	3001580
Total length (>= 1000 bp)	2983688
Total length (>= 5000 bp)	2963380
Total length (>= 10000 bp)	2949868
Total length (>= 25000 bp)	2932269
Total length (>= 50000 bp)	2823182
# contigs	32
Largest contig	482303
Total length	2986611
GC (%)	37.25
N50	228482
N75	161236
L50	5
L75	9
# N's per 100 kbp	0.00

[sardana.p@ood results_2022_12_13_19_31_42]$

Fig 1. **Report.txt file**

Here, in the output the N50 value is 228482 which is the length of smallest contig after they have been ranked from longest to smallest and the sum of contigs length covers 50% of total size of the contig. Here the L50 value is 5 which is the smallest number of contigs whose length makes up half of the genome size.The N's per 100 kb is '0.00' which means there are no missing reads or basepairs in genome assembly. In the GC content plot above Fig 1.

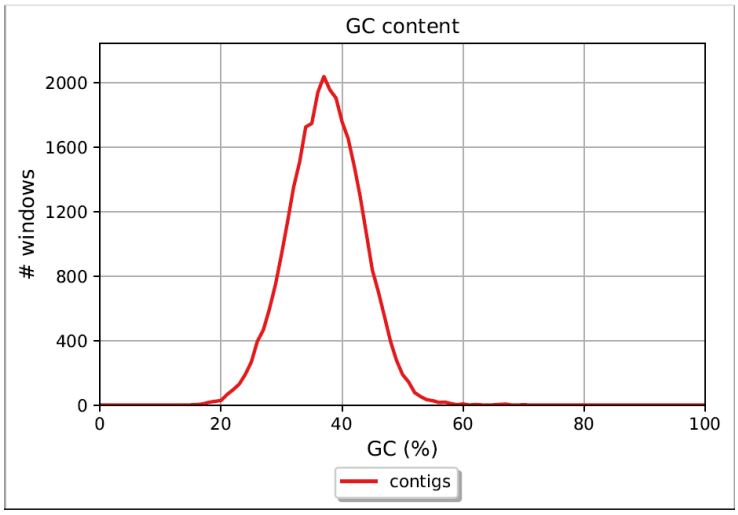


Fig2- GC content plot The GC content is 37.9% when the contig is 2000 bp which is calculated on the basis of total number of G and C nucleotides in the assembly divided by total length of the assembly.

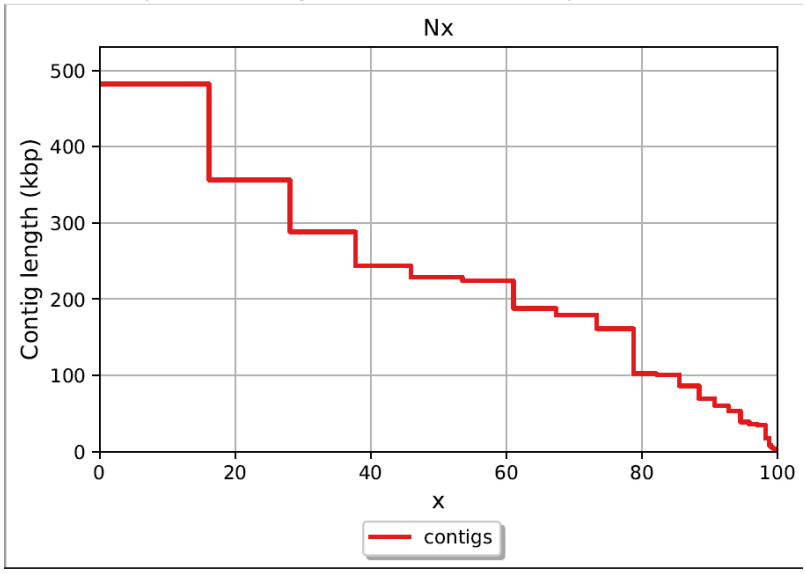


Fig 3. **Nx plot Contig length v/s contigs**

Here in the Nx plot, Fig2.the largest contig length is close to 500 kbp accounting for at least 18% of the contigs in the assembly.
According to the contigs coverage histogram graph,

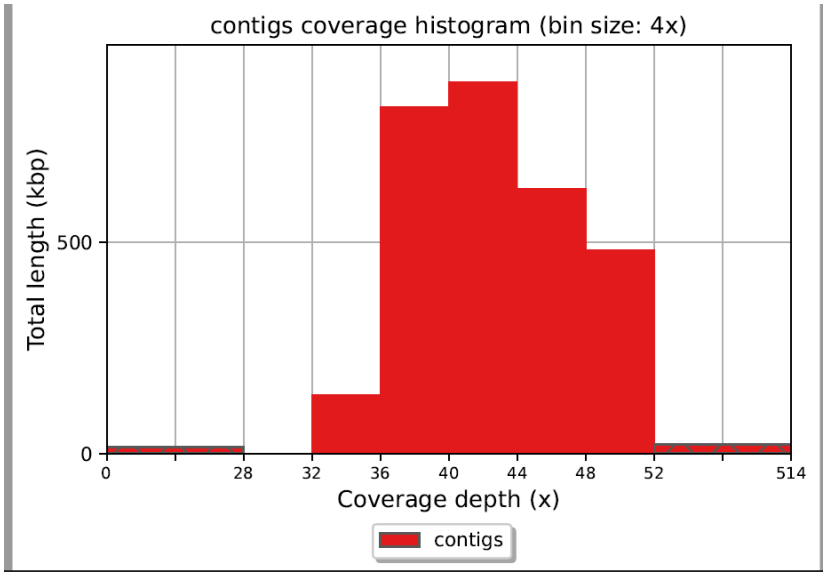


Fig 4. **Contigs coverage histogram Total length v/s contigs**

The coverage depth is the number of times each nucleotide is sequenced in an experiment which maps the read length. In the above graph the coverage depth at bin size 4x is quite high and the total length of contigs covering more than 1000 kbp.
According to the cumulative index plot graph below,

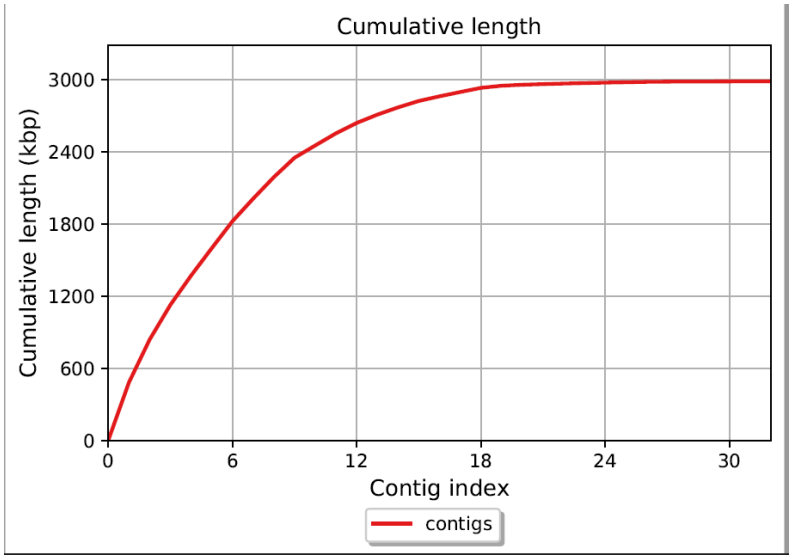


Fig 5. **Cumulative index plot**
the cumulative length of assemblies are plotted for contigs and indexed from longest to shortest. The cumulative length of the first x contigs are plotted as a function of x for the assembly using a log-transformed x-axis.

The transcriptome assembly of Listeria monocytogenes was performed using Trinity assembler and the trinity_de_novo_stats suggest that total trinity genes

account for 2475 genes and percentage GC is 38.96. The total assembled bases appears to be 2243068.

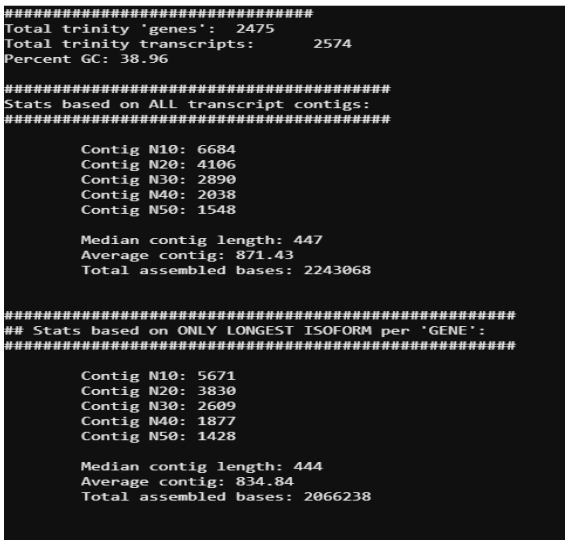
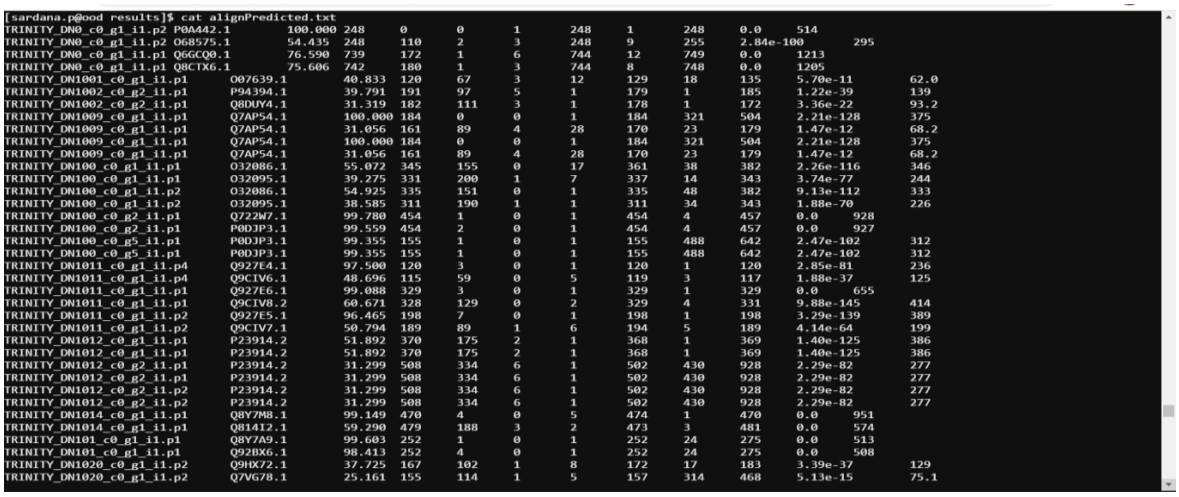


Fig 6 **Trinity stats.txt**

The output file N10 through N50 values are computed on the basis of assembled contigs and longest isoform of contigs. Here the N50 values of transcript contigs and longest isoform per gene are 1548 and 1428 respectively. The assembly program produces too many transcript isoforms due to N50 values being exaggerated based on longer transcripts. this , the Nx values are computed on the basis of single longest isoform per gene .This indicates Nx values based on single longest isoform per gene are lower than assembled contigs.

To annotate genes within the genome used **blast**, **Transdecoder** and **hmmscan** to identify protein sequences similarity and protein predictions using input data. Reviewing the Transdecoder.predict *pep creates and output protein fasta file to filter the protein predictions taking input in ORF, BLAST and protein domain information. The fasta file predict the likely coding regions within the transcript sequences. TransDecoder identifies coding sequences based on the following criteria: 1. Minimum length open reading frame (ORF) found in transcript sequence. 2. Log likelihood score similar to what is computed by the gene 3. Coding score which is greatest when the ORF is scored in the 1st reading frame as compared to scores in other 5 reading frames. 3. The longer orf is reported when the candidate ORF is found encapsulated by coordinates of another candidate orf. The total predicted proteins were found to be 2313 from Trinity.fasta.transdecoder.pep fasta file. On the basis of reviewing the blast output alignpredicted.txt of shell script alignPredicted_args.sh, the output above



```
[sardana.p@ood results]$ cat alignPredicted.txt
TRINITY_DN0_c0_g1_i1.p2 P0A442.1        100.000 248     0       0       1       248     1       248     0.0     514
TRINITY_DN0_c0_g1_i1.p2 O68575.1        54.435  248     110     2       3       248     9       255     2.84e-100       295
TRINITY_DN0_c0_g1_i1.p1 Q6GCQ0.1        76.590  739     172     1       6       744     12      749     0.0     1213
TRINITY_DN0_c0_g1_i1.p1 Q8CTX6.1        75.606  742     180     1       3       744     8       748     0.0     1205
TRINITY_DN1001_c0_g1_i1.p1      O07639.1        40.833  120     67      3       12      129     18      135     5.70e-11        62.0
TRINITY_DN1002_c0_g2_i1.p1      P94394.1        39.791  191     97      5       1       179     1       185     1.22e-39        139
TRINITY_DN1002_c0_g2_i1.p1      Q8DUY4.1        31.319  182     111     3       1       178     1       172     3.36e-22        93.2
TRINITY_DN1009_c0_g1_i1.p1      Q7AP54.1        100.000 184     0       0       1       184     321     504     2.21e-128       375
TRINITY_DN1009_c0_g1_i1.p1      Q7AP54.1        31.056  161     89      4       28      170     23      179     1.47e-12        68.2
TRINITY_DN1009_c0_g1_i1.p1      Q7AP54.1        100.000 184     0       0       1       184     321     504     2.21e-128       375
TRINITY_DN1009_c0_g1_i1.p1      Q7AP54.1        31.056  161     89      4       28      170     23      179     1.47e-12        68.2
TRINITY_DN100_c0_g1_i1.p1       O32086.1        55.072  345     155     0       17      361     38      382     2.26e-116       346
TRINITY_DN100_c0_g1_i1.p1       O32095.1        39.275  331     200     1       7       337     14      343     3.74e-77        244
TRINITY_DN100_c0_g1_i1.p2       O32086.1        54.925  335     151     0       1       335     48      382     9.13e-112       333
TRINITY_DN100_c0_g1_i1.p2       O32095.1        38.585  311     190     1       1       311     34      343     1.88e-70        226
TRINITY_DN100_c0_g2_i1.p1       Q722W7.1        99.780  454     1       0       1       454     4       457     0.0     928
TRINITY_DN100_c0_g2_i1.p1       P0DJP3.1        99.559  454     2       0       1       454     4       457     0.0     927
TRINITY_DN100_c0_g5_i1.p1       P0DJP3.1        99.355  155     1       0       1       155     488     642     2.47e-102       312
TRINITY_DN100_c0_g5_i1.p1       P0DJP3.1        99.355  155     1       0       1       155     488     642     2.47e-102       312
TRINITY_DN1011_c0_g1_i1.p4      Q927E4.1        97.500  120     3       0       1       120     1       120     2.85e-81        236
TRINITY_DN1011_c0_g1_i1.p4      Q9CIV6.1        48.696  115     59      0       5       119     3       117     1.88e-37        125
TRINITY_DN1011_c0_g1_i1.p1      Q927E6.1        99.088  329     3       0       1       329     1       329     0.0     655
TRINITY_DN1011_c0_g1_i1.p1      Q9CIV8.2        60.671  328     129     0       2       329     4       331     9.88e-145       414
TRINITY_DN1011_c0_g1_i1.p2      Q927E5.1        96.465  198     7       0       1       198     1       198     3.29e-139       389
TRINITY_DN1011_c0_g1_i1.p2      Q9CIV7.1        50.794  189     89      1       6       194     5       189     4.14e-64        199
TRINITY_DN1012_c0_g1_i1.p1      P23914.2        51.892  370     175     2       1       368     1       369     1.40e-125       386
TRINITY_DN1012_c0_g1_i1.p1      P23914.2        51.892  370     175     2       1       368     1       369     1.40e-125       386
TRINITY_DN1012_c0_g2_i1.p1      P23914.2        31.299  508     334     6       1       502     430     928     2.29e-82        277
TRINITY_DN1012_c0_g2_i1.p1      P23914.2        31.299  508     334     6       1       502     430     928     2.29e-82        277
TRINITY_DN1012_c0_g2_i1.p2      P23914.2        31.299  508     334     6       1       502     430     928     2.29e-82        277
TRINITY_DN1012_c0_g2_i1.p2      P23914.2        31.299  508     334     6       1       502     430     928     2.29e-82        277
TRINITY_DN1014_c0_g1_i1.p1      Q8Y7M8.1        99.149  470     4       0       5       474     1       470     0.0     951
TRINITY_DN1014_c0_g1_i1.p1      Q81412.1        59.290  479     188     3       2       473     3       481     0.0     574
TRINITY_DN101_c0_g1_i1.p1       Q8Y7A9.1        99.603  252     1       0       1       252     24      275     0.0     513
TRINITY_DN101_c0_g1_i1.p1       Q92BX6.1        98.413  252     4       0       1       252     24      275     0.0     508
TRINITY_DN1020_c0_g1_i1.p2      Q9HX72.1        37.725  167     102     1       8       172     17      183     3.39e-37        129
TRINITY_DN1020_c0_g1_i1.p2      Q7VG78.1        25.161  155     114     1       5       157     314     468     5.13e-15        75.1
```

is the query id which is the target, subject id, % identity, query length, alignment length, mismatches, gap opens, evalue, and bit score. The maximum score is the highest alignment score between the query sequence and database segments which is inversely proportional to the e- value. The larger the bit score, less likely to obtain by chance than is a smaller bit score. E value is a measure of likeliness that sequence similarity is not by random chance. The percentage identity the similarity of query to the aligned sequences. The total score is the sum of alignment scores of all sequences from same database. The percent query coverage is the percent of the query length included in aligned segments.

The pipeline can effectively use such tools hhmscan, Transdecoder and BLAST if we we want to adapt the pipeline to work on another denovo genome assembly.

Finally created to predict the coding function of proteins KEGG API was used to access the data from KEGG.

CONCLUSION:-

From the above pipeline, we determined the assembly of source genome by working on distinct computational metrices like N50, L50, NA59, genome fraction, GC%,total contig length,total transcripts etc Thereafter, we annotated the genes within the genome and predicted the protein function for the genes. We can implement the new strategies in pipeline by investigating on the properties of genome we are studying such as genomic size, repeats,GC –content etc To extract high quality of DNA , various intrinsic properties can be studied effectively starting with checking the DNA quality requirements and use the best quality molecular weight. Also choosing appropriate sequencing technology will influence the cost and success of the assembly to a large extent.

References:-

Dominguez Del Angel V, Hjerde E, Sterck L, Capella-Gutierrez S, Notredame C, Vinnere Pettersson O, Amselem J, Bouri L, Bocs S, Klopp C, Gibrat JF, Vlasova A, Leskosek BL, Soler L, Binzer-Panchal M, Lantz H. Ten steps to get started in Genome Assembly and Annotation. F1000Res. 2018 Feb 5;7:ELIXIR-148. doi: 10.12688/f1000research.13598.1. PMID: 29568489; PMCID: PMC5850084.

Haas, B. J., Papanicolaou, A., Yassour, M., Grabherr, M., Blood, P. D., Bowden, J., Couger, M. B., Eccles, D., Li, B., Lieber, M., MacManes, M. D., Ott, M., Orvis, J., Pochet, N., Strozzi, F., Weeks, N., Westerman, R., William, T., Dewey, C. N., … Regev, A. (2013). De novo transcript sequence reconstruction from RNA-seq using the Trinity platform for reference generation and analysis. Nature Protocols, 8(8), 1494–1512

Martin, J., Wang, Z. Next-generation transcriptome assembly. *Nat Rev Genet* 12, 671–682 (2011). https://doi.org/10.1038/nrg3068

Venket Raghavan, Louis Kraft, Fantin Mesny, Linda Rigerte, A simple guide to *de novo* transcriptome assembly and annotation, *Briefings in Bioinformatics*, Volume 23, Issue 2, March 2022, bbab563, https://doi.org/10.1093/bib/bbab563

Tallon, Luke & Liu, Xinyue & Bennuru, Sasisekhar & Chibucos, Marcus & Godinez, Alvaro & Ott, Sandra & Zhao, Xuechu & Sadzewicz, Lisa & Fraser, Claire & Nutman, Thomas & Dunning Hotopp, Julie. (2014). Single molecule sequencing and genome assembly of a clinical specimen of Loa loa, the causative agent of loiasis. BMC genomics. 15. 788. 10.1186/1471-2164-15-788.

Lischer, H.E.L., Shimizu, K.K. Reference-guided *de novo* assembly approach improves genome reconstruction for related species. *BMC Bioinformatics* 18, 474 (2017). https://doi.org/10.1186/s12859-017-1911-6

Gurevich A, Saveliev V, Vyahhi N, Tesler G. QUAST: quality assessment tool for genome assemblies. Bioinformatics. 2013 Apr 15;29(8):1072-5. doi: 10.1093/bioinformatics/btt086. Epub 2013 Feb 19. PMID: 23422339; PMCID: PMC3624806.