

Article

A Robust Method for Finding the Automated Best Matched Genes Based on Grouping Similar Fragments of Large-Scale References for Genome Assembly

Jaehee Jung ¹, Jong Im Kim ², Young-Sik Jeong ³ and Gangman Yi ^{3,*}

¹ Department of General Education, Hongik University, Seoul 04066, Korea; jhjung@hongik.ac.kr

² Department of Biology, Chungnam National University, Daejeon 34134, Korea; jongim@cnu.ac.kr

³ Department of Multimedia Engineering, Dongguk University, Seoul 04620, Korea; ysjeong@dongguk.edu

* Correspondence: gangman@dongguk.edu

Received: 9 August 2017; Accepted: 11 September 2017; Published: 13 September 2017

Abstract: Big data research on genomic sequence analysis has accelerated considerably with the development of next-generation sequencing. Currently, research on genomic sequencing has been conducted using various methods, ranging from the assembly of reads consisting of fragments to the annotation of genetic information using a database that contains known genome information. According to the development, most tools to analyze the new organelles' genetic information requires different input formats such as FASTA, GeneBank (GB) and tab separated files. The various data formats should be modified to satisfy the requirements of the gene annotation system after genome assembly. In addition, the currently available tools for the analysis of organelles are usually developed only for specific organisms, thus the need for gene prediction tools, which are useful for any organism, has been increased. The proposed method—termed the genome_search_plotter—is designed for the easy analysis of genome information from the related references without any file format modification. Anyone who is interested in intracellular organelles such as the nucleus, chloroplast, and mitochondria can analyze the genetic information using the assembled contig of an unknown genome and a reference model without any modification of the data from the assembled contig.

Keywords: gene; gene annotation; contig; NGS; organelle genome; gene search

1. Introduction

The last twenty years have been a remarkable era for big data on biology and medicine. The sequences of large genomes can be obtained through the use of high-throughput genome sequencing. Biologists have been able to obtain various information from genome sequence analysis with the development of NGS(Next-Generation Sequencing). However, NGS provides reads that range in length from 150 bp to 10 kb. These reads can provide limited information owing to their short lengths. Even though the length of each read is enlarged due to the use of high-throughput techniques, the assembly and annotation of whole consensus genomes are the subject of ongoing research. Many fundamental questions in genomics and biology will start to be addressed regarding the structure and function of genomes, the molecular interplay within cells and organs, and the properties of entire species and ecosystems. Therefore, studies on how to analyze other genomes and assemblies have been conducted actively, using a large number of reads [1]. The direction of such studies has been focused on the functions and nomenclature of the genome after the base sequence alignment of biological sequence data—the amount of which is rapidly increasing, according to the sequence order or phylogeny of the genome [2,3]. Methods of assembling the base sequence with regard to fragment-divided reads can be divided into two groups: those where the reference model

genome is present and those where it is not. When there is a model genome that can be referenced by tens of thousands of reads, a contig can be generated by a method of similar sequence matching or read mapping using BLAST (Basic Local Alignment Search Tool) to determine the similarity between the sequence and the model genome. In contrast, when there is no model genome, a contig can be generated using de novo assembly [4]. De novo assembly is a method of connecting reads after cutting them to a certain length according to the similarities among them [5].

Each of the generated assembled contigs is named or analyzed to obtain the sequence of the genome, which is the main objective of NGS. Another objective of NGS is to study phylogeny and evolutionary relationships using the analyzed genome information. Genome-based phylogenetic analysis can be done based on the homology analysis of the genome. That is, NGS aims to find genomes that contain similar functions and is an accurate method of assembly. A number of programs that predict gene information from the assembled contigs have been developed to analyze assembled contigs. The gene prediction can be various due to the domain [6], thus the feature of tool and domain should be investigated. GeneMarkS [7,8], Glimmer (Gene Locator and Interpolated Markov ModelER) [9], Genscan [10], GenomeScan [11], EasyGene [12,13], and AUGUSTUS [14] are some of the better-known programs. GeneMarkS [7] recognizes a coding sequence (CDS) from the inputted sequence, thereby predicting the length and direction (forward or reverse) of a given gene, and provides simple results very rapidly. Glimmer [9] is a program that predicts the gene locations of some microorganisms (especially pathogenic bacteria and archaea) and viruses based on the Markov model. GenomeScan [11] is a program that predicts the protein homology of input DNA using BLASTx and BLASTp, and provides genome location information on genes, which is useful in model organisms such as vertebrates, arabidopsis, and maize. These programs are considerably limited in their ability to obtain genetic information, since they can be applied only to specific organisms. In AUGUSTUS [14], two steps of training and testing procedures are required to use the system. It may require several training procedures to find appropriate related species. Furthermore, BLAST must be used to obtain results that provide only basic information such as the gene location, or the corresponding genes must be searched from database (NCBI, JGI, KEGG, etc.) to verify biological information such as the names and functions of genes. However, information collection can be difficult due to the long processing time by online servers due to the large size of the contig or database to be compared. Furthermore, it is difficult for biologists to analyze genes via inputs at the command line. EasyGene [12] proposed a method of searching genes automatically using the statistical hidden Markov model after creating a training model using the similarity in the Swiss-Prot database [15]. It has been applied to prokaryotes to a limited extent. Thus, it has the drawback that it is difficult to analyze eukaryotic sequences and intracellular organelles with it. Moreover, it cannot perform a comparative analysis if no accurate information about model organisms is available. Furthermore, it is not easy to analyze sequences corresponding to a specific organelle (e.g., sequences corresponding to the nucleus, mitochondria, or plastids) using only contig information, since the input formats differ depending on the characteristics of the system. However, the proposed genome_search_plotter has an advantage in that it can easily search organellar genomes using an inputted reference model.

As regards reference sequence matching, MUMmer [16] is a well-known program for this purpose. MUMmer is characterized by the fact that it displays two sequences in a diagram by aligning the entire genome rapidly [17]. It has advantages in that the sequence alignment of the whole genome can be done using PROmer and MUMmer; and that structural differences such as repeats, reversals, insertions, or deletions between two sequences can be analyzed using mummerplot. However, although this program performs a comparison of the similarity between two sequences of entire inputs with diagrams, it cannot be used to predict a genome that has not yet been analyzed. In other words, it has a drawback in that cannot predict a genome by sequence location, as the analysis of a similar sequence is done based on gene information with diagrams.

Thus, the present paper aims to separate the sequences corresponding to the preferred intracellular organelles (nucleus, mitochondria, or plastids) [18] from the assembled contigs to facilitate their

analysis, since the existing methods of genome analysis have difficulty separating a specific genome from mixed sequences, and an analysis of the entire genome sequence provides analysis results on all the subcellular organelles (nucleus, mitochondria, and plastids).

2. Implementation

As shown in Figure 1, there are several steps to searching the best-matched genome in previous approaches. The first step to define the function is finding open reading frames (ORFs) or RNA sequences by BLAST. Based on the results, the biologist also runs another program such as a tRNA scan [19] or a gene annotation program. At this point, since the input and output formats are different among the currently available systems, another annoying job such as format changing must also be executed. For example, if the biologists want to draw a gene map [20], the input format should be changed to gene bank format. However, the suggested system has a pipeline for genome searching; thus, it is unnecessary to do other extra jobs such as format changing and inputting the query for each different system.

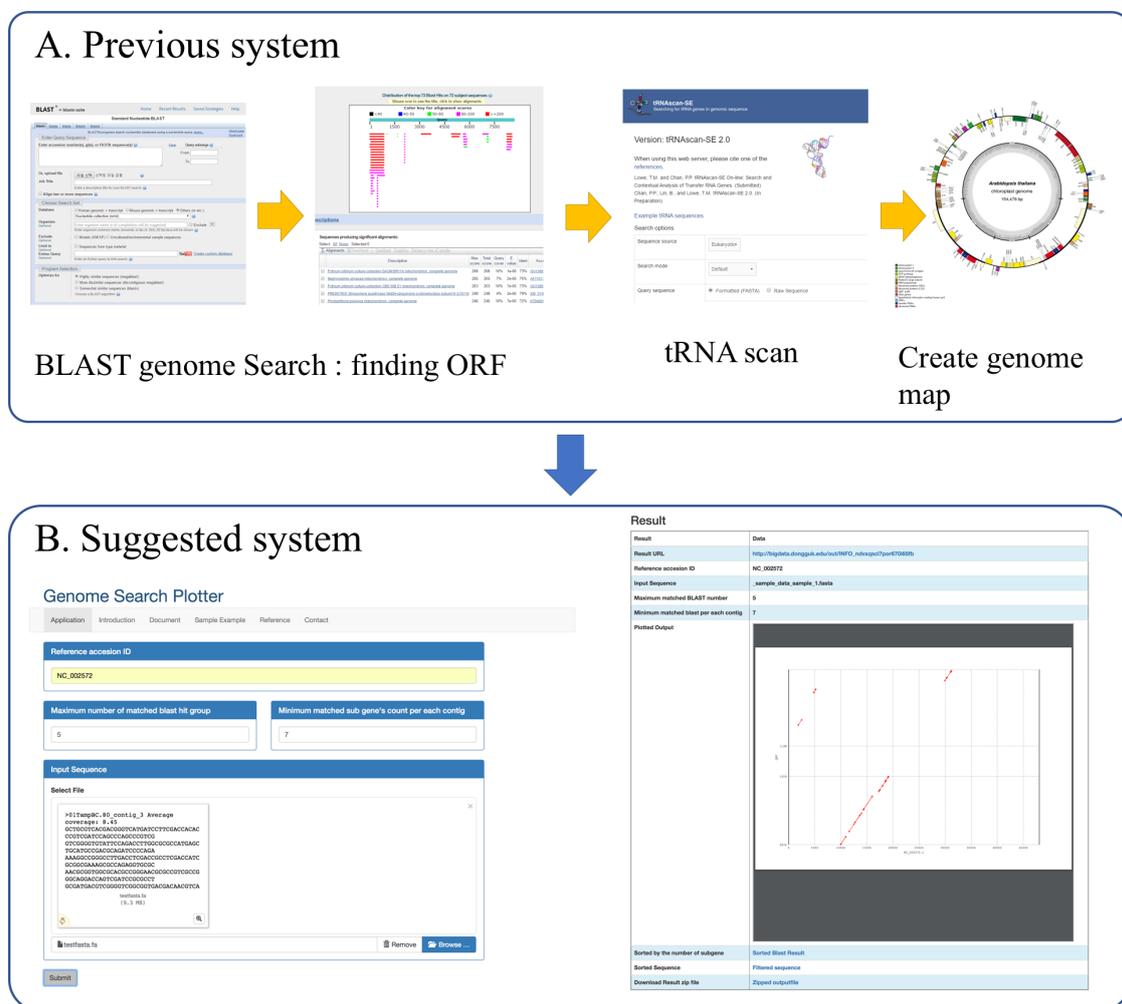


Figure 1. Overview of system: In previous systems, in order to create a genome map for unknown sequences, biologists search for open reading frames (ORFs) and find RNA sequences manually. Each step requires different input types; thus, the data format should be changed depending on the system. However, the suggested system—genome_search_plotter—provides genome matched plotter and other supplementaries without any modification of format.

2.1. Inputs

The proposed program was designed with a web-based interface for convenient use by biologists (The accessible URL is http://bigdata.dongguk.edu/genome_search_plotter). It uploads contigs obtained by assembly and provides the results (Figure 2). The mandatory inputs include the following three user-defined values: The first user-defined value is the reference accession ID, which is used to extract the whole amino acid and nucleotide sequence of CDS regions from the NCBI GenBank database. A method of sampling the amino acid sequence extracts features named as CDS regions in the NCBI GenBank database, corresponding to a DNA section where protein expression occurs. CDS refers to a sequence after the transcription and splicing, where all intron regions are removed and only the exon remains.

Figure 2. Genome Search Plotter: The program performs gene matching based on the inputs of query sequences, a reference accession number, and two user-defined option values.

The second user-defined value refers to the maximum number of groups that are most closely matched according to the BLAST result. As shown in Figure 3A, BLAST hits can be created between some groups based on the query-matched start and end positions. If two different BLAST results are overlapped on the end position and the start position, these results are assigned to the same group. On the other hand, when there are no overlapping positions between two BLAST results, they are assigned to different groups. In Figure 3A, four separated groups can be seen. Therefore, the maximum number of groups by BLAST is represented by N . When the second user-defined value (N ; i.e., the maximum number of groups from the BLAST results) is set to 5 by the user's input in Figure 2, the number of searched groups is 4 in Figure 3. Thus, all candidate regions can be set up to all sub-genes in the hit genes. That is, all groups whose number of groups searched is less than N are regarded as candidate groups in which there is a possibility of finding similar genes. By contrast, if the number of searched groups is larger than N , the groups are ordered by smallest E-value. If the value of N is not set and there are many matched fragments from the BLAST results, the query sequence is matched to find the best hit recursively; thus, the overall result can be skewed by only one query.

The next step is counting the sub-genes in all N groups. In Figure 3B, similar genes connected to the first user-defined value N are all searched based on a single query contig, and they are called sub-genes. Genes are set only when the number of connected sub-genes is larger than the minimum value of the third user-defined value. For example, in Figure 3, 1 sub-gene from the first group (green), 2 sub-genes from the second group (blue), 3 sub-genes can be found from the third group (yellow), and 2 sub-genes from the fourth group (red); thus, the total number of sub-genes for this contig is 8.

If the third user-defined value k is set below 8 (as shown in Figure 2), then the analyzed contigs can be set as a candidate gene. However, when k is set above 8, the analyzed contigs cannot be set as the matched candidate after filtering. Genes that are predicted based on the query sequence can be found with the minimum number of each of the sub-genes connected within the group and with the number of groups of the maximally matched genes, as shown in Figure 3A.

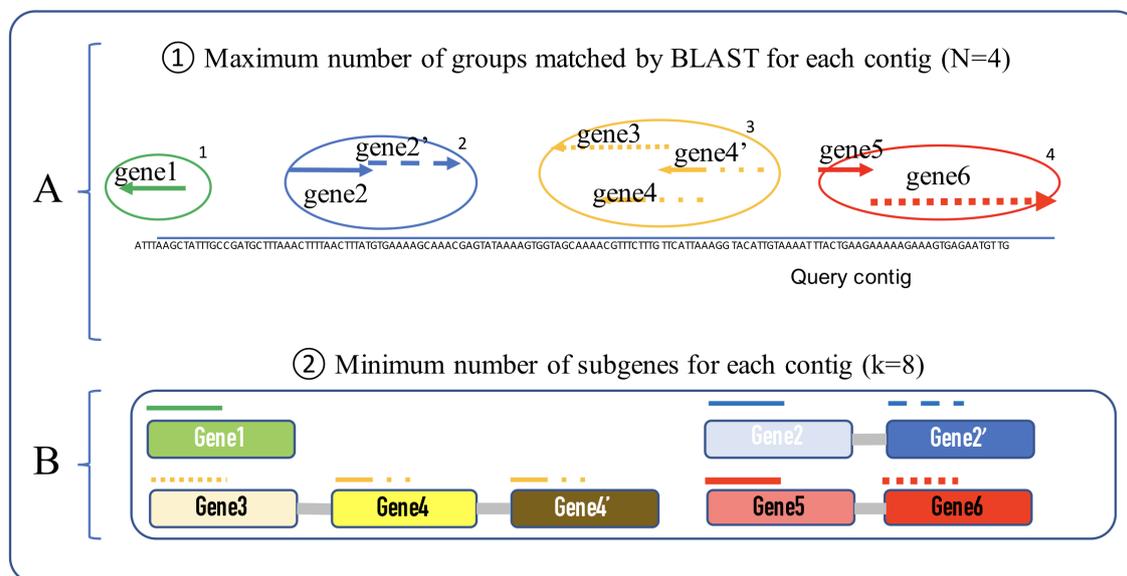


Figure 3. User-defined values (N, k): N is the maximum number of matched groups for each query, and k is the minimum number of sub-genes for each group.

2.2. Process

Whenever an analysis request is inputted, a new individual ID is assigned and a job space is provided to perform the task. First, BLAST is performed with the reference model, in which the inputted assembled query contigs are compared with the amino acid reference sequences. Grouping is done based on a comparison of the start and end positions matched with a comparison of the inputted user-defined values after performing BLAST, thereby selecting only as many groups as inputted value for the maximum number of groups. The selected groups are then ordered based on the E -value of BLAST in ascending order. After BLAST and the grouping comparison, an alignment is done based on the number of matched sub-genes in each of the queries in descending order. Here, filtering and naming of the query sequence are conducted only when the number of sub-genes is larger than the inputted value for the minimum number of sub-genes k , and gene information is provided. The whole process for obtaining the results is explained in Figure 4.

$$GID_{start}^i \leq \forall s_j^i : i \leq GID_{end}^i$$

i : number of matched groups

j : number of sub-genes for each group

GID^i : gene group ID

GID_{start}^i : gene group start position

GID_{end}^i : gene group end position.

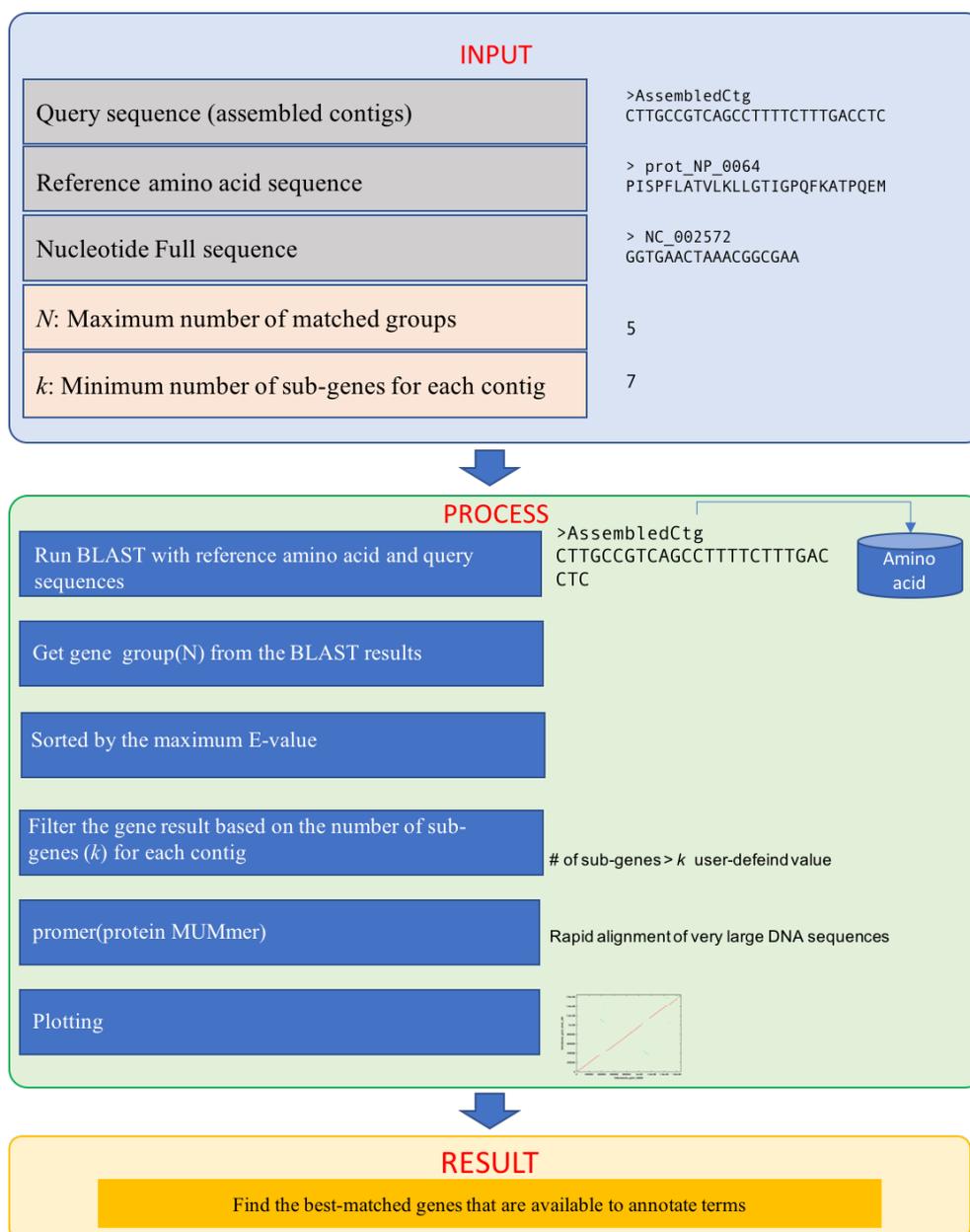


Figure 4. The running process of the genome_search_plotter: Three sequences and two user-defined values are required as inputs. The reference sequences including the amino acid and whole genome sequences are automatically downloaded from the NCBI GenBank database. The system executes BLAST with the amino acid reference sequence and the query sequence, and the genome_search_plotter groups genes that are overlapped by the start position or the end position of other genes. The genome_search_plotter creates sequence files that find the best-matched genes after filtering steps, and generate a result plot.

GID^i refers to the group's ID and s^i refers to a sub-gene that belongs to the i -th group. Regardless of backward or forward matching between the start and end positions, all sub-genes that belong to the i -th group must be located between the start and end positions in the i -th group. N refers to the maximum number of BLAST hit groups that should be searched and set by a user; it is the maximum number of i .

$$\begin{aligned} ||s_j^i|| &\geq k, i \leq N \\ \forall s_j^i: s_j^i &\subseteq GID^i \end{aligned}$$

where s^i refers to a sub-gene that belongs to the i -th gene ID group. Up to the maximum of N groups, in the case where the total number of sub-genes that belong to each group exceeds the user-defined k , the most optimally-matched reference contig can be found with regard to the query sequence, and the number of sub-genes in the group that does not exceed k is filtered and ignored.

3. Results

As the assembly query sequences are inputted into the system, results in PDF format and several supplementary files are generated. The results are shown as a graph depicting matches with the reference genome on the X-axis and the query sequences uploaded by the user on the Y-axis. Each line on the plot indicates that a query contig is matched with the reference sequences. As an example of visualizing matched contigs in Figure 5, the red-colored lines refer to forward matching with the reference sequence, and the blue-colored lines refer to backward matching with the reference sequence.

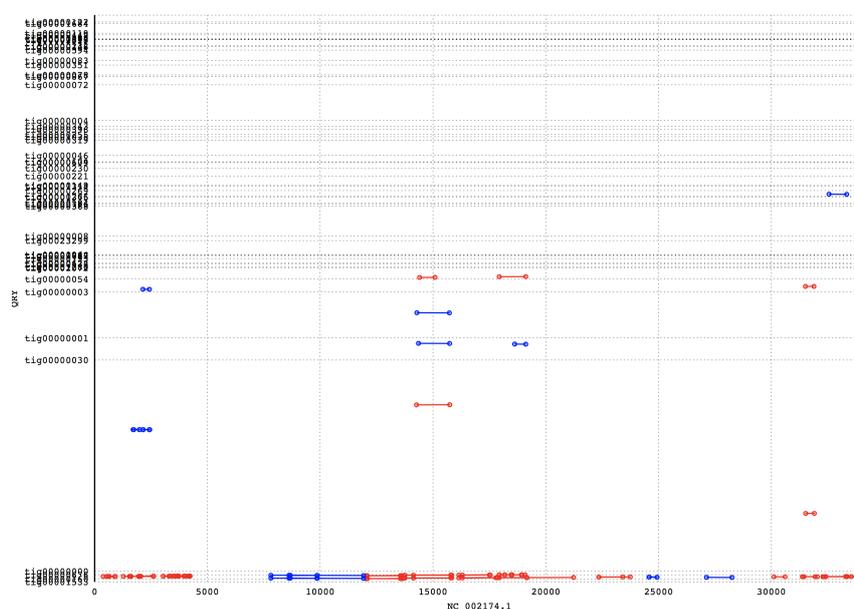


Figure 5. Example plot of contigs that are optimally matched: The Y-axis shows the query sequences and the X-axis shows the reference sequence. The red-colored lines show forward matching with the reference sequence, and the blue-colored lines show backward matching with the reference sequence.

Based on the inputted query options, not only a plotted PDF file but also sorted query sequences and BLAST results are supported (Figure 6). In the sorted query sequences file, sequences are sorted by the number of matched sub-genes, and the sequences that do not meet the minimum value of k are filtered out. BLAST results are provided as the BLAST output, which is grouped by the start and end positions for each contig. As depicted in Figure 6, the detailed information such as gene name and BLAST score value of matched genes is provided in the result file. However, information using the file is not well-suited to check whether or not assembled contigs of query are well matched to references. Positions in blue-colored box of Figure 6 represent solid-lines of different color according to sequence directions in Figure 7. The lines in Figure 7 represent matched regions of query and references. The actual running time for sample data (input sequence of 22,266 bp and reference sequence of 97,626 bp) was about 20 s. Figure 7 shows the results page as outputted by the online version of genome_search_plotter.

```

DB = lcl|NC_002572.1_prot_NP_066470.1_17 [gene=nad4L] [protein=NADH]
0 ACCTTTCATTAAATCGATAAATCAATCGAATCGTTCCTCAACTATAATAAATCAACCAATAGCTAGTCCAAATGCTGATTCTCCGCTGC
-Pos = [6244,6149] Len = 96, Score = 55.8 bits (133), Expect = 6e-12, Method: Compositional matrix adjust.
Identities = 27/32 (84%), Positives = 31/32 (97%), Gaps = 0/32 (0%), Frame = -1
Query 6244 AAESAIGLAILVIYRVRTIAIEFFIDLKMG 6149
AAESAIGLAILVIYR+RTI 1+FI+L+KG
Sbjct 72 AAESAIGLAILVIYRIRGTIRIDIFINLIK 103

DB = lcl|NC_002572.1_prot_NP_066496.1_43 [gene=nad11] [protein=NADH]
0 TATCAAGGTAATTGATATGTAATGAACCTTCAATTGACCGAATAAAATTCGGTTACACAACCAAGGAATAAGCCAGCAGCACTTAAGATC
+Pos = [2760,2849] Len = 90, Score = 25.0 bits (53), Expect = 1.0, Method: Compositional matrix adjust.
Identities = 11/34 (32%), Positives = 21/34 (62%), Gaps = 4/34 (12%), Frame = +3
Query 2760 YQNLICNELSIDRIKFGYTT---KVNKPALKI 2849
++ + + E D+I+FG+ T ++NKP LK+
Sbjct 248 FKNDFNVEEISDKIRFGFDLTKRQRLNKP LKL 281

DB = lcl|NC_002572.1_prot_NP_066477.1_24 [gene=atp6] [protein=ATP]
0 ATAACAAGGCTATAACTACTATAAATATACAAGAGGAGTGAATATACACTACCATCAGGGAATAAACACACGAAATCTAACGAGCCCAATAGTA
1 CCCAT
-Pos = [3340,3236] Len = 105, Score = 21.6 bits (44), Expect = 8.8, Method: Compositional matrix adjust.
Identities = 12/35 (34%), Positives = 19/35 (54%), Gaps = 0/35 (0%), Frame = -1
Query 3340 MGTGYLVRFSLCLFPDGSVYFTPLVYLMISIAVY 3236
+ +GL FSL L P + PL+ + II+ V+
Sbjct 132 INKHGFFSFLPFGAPLALAPLLVPIEISYVF 166

DB = lcl|NC_002572.1_prot_NP_066454.1_1 [gene=atp8] [protein=ATP synthase]
0 TGTAAATAAACCAACCATTCTTCATATCTGCTCATCATTTAAAGCATGTATCACAGAACCGCACTTAAGAAT
+Pos = [5020,5094] Len = 75, Score = 21.2 bits (43), Expect = 9.7, Method: Compositional matrix adjust.
Identities = 6/25 (24%), Positives = 15/25 (60%), Gaps = 0/25 (0%), Frame = +1
Query 5020 CNKPPILLSCSFKAKITEPALKN 5094
CNK + + + + + K + + + KN
Sbjct 108 CNKNYIFLFTINNYKNFVDFDKIKN 132

DB = lcl|NC_002572.1_prot_NP_066473.1_20 [gene=nad1] [protein=NADH]
0 TTCACCTAAGAAAAATAGCTGAACCCATTGCTGCATATTCAGTATTATCCAGAAACAAGTTCAGCTCAGCTCAGGTAATCAACGGAGCCCGA
1 TTTGTTCTGCTAGCGGAAACAAGAACATCAAAACAAGGGAATAATGGTATACAATA
-Pos = [6956,6795] Len = 162, Score = 100 bits (248), Expect(2) = 5e-34, Method: Compositional matrix adjust.
Identities = 47/54 (87%), Positives = 52/54 (96%), Gaps = 0/54 (0%), Frame = -3
Query 6956 YCIPLPLFLMFFVSALAEINRPFDLPEAEALVSGNYTEYAMGFALFFLGE 6795
Y IPLPF+F+MFF+SALAEINRPFDLPEAEALVSG+NTEY+AM FALFFLGE
Sbjct 184 YIIPLPVFMFFISALAEINRPFDLPEAEALVSGNTEYSAMAFALFFLGE 237
    
```

Figure 6. Example of supplementary BLAST result: The BLAST results represent matched genes from references. This supplementary data enables genes to be checked manually.

Result

Result	Data
Result URL	http://bigdata.dongguk.edu/out/INFO_23890wagpc2al47do91
Reference accession ID	NC_002572
Input Sequence	_sample_data_sample_1.fasta
Maximum matched BLAST number	5
Minimum matched blast per each contig	7
Plotted Output	
Sorted by the number of subgene	Sorted Blast Result
Sorted Sequence	Filtered sequence
Download Result zip file	Zipped outputfile

Figure 7. Results analysis: The genome_search_plotter provides a dedicated URL so that users can re-visit or distribute the results via an easily accessible online address. The system generates the results as separate files, as well as the graph plotted as a PDF file that can be downloaded. The filtered sequence file is sorted by the number of sub-genes, and the sorted sequence file is also downloadable.

4. Conclusions

A `genome_search_plotter` is a program designed to generate the gene information of a contig generated by an assembler after NGS. The previous approaches to finding the gene information required biologists to make different input files depending on the systems used. Other plotting programs require such tasks as BLAST or secondary processing with regard to the input and output values for analysis. In these steps, the required input formats also vary depending on the programs used. However, the `genome_search_plotter` has the advantage of enabling the easy acquisition of the gene information of a genome using a reference amino acid sequence, the nucleotide sequence, and the assembled contig. The suggested `genome_search_plotter` does not require re-analysis of the input and output values, and can perform an analysis of the intracellular organelles by using several user-defined values and the inputted reference sequence. The system was implemented by considering not only the input of reference sequences from the NCBI GenBank database, but also by uploading the reference sequences directly, by allowing a degree of freedom of the sequence input that can be referenced. Regarding areas of future study, faster program execution by the effective use of a multi-core computational system will be implemented and the `genome_search_plotter` will be compared with existing methods in terms of its utilities for various comparative analyses.

Detailed genome information such as matched CDS, RNA, and referred position can be acquired after constructing the potential genome using `genome_search_plotter`. For future work, we would like to provide an updated method that can support gene annotation functions on organelle genomes. Moreover, the reference sequences in the current systems are only available to put the NCBI accession number. As a future work, the user-defined reference sequences can be uploaded in order to find the most related species.

Acknowledgments: This work was supported by the National Research Foundation of Korea's (NRF): Basic Science Research Program, which is funded by the Ministry of Education (NRF-2016R1C1B1007929, NRF-2017R1D1A1A09000631, NRF-2016R1D1A1A09919318, NRF-2015R1D1A1A01057899) and the Hongik/Dongguk University Research Fund of 2016.

Author Contributions: Jaehee Jung conducted research for the related works, performed the experiments, and wrote the manuscript. Jong Im Kim found the biological problems and performed experiments. Young-Sik Jeong improved the performance of the proposed method and revised the manuscript. Gangman Yi supervised the main idea and provided a review, comments, assessment, etc.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Song, H.J.; Lee, J.; Graf, L.; Rho, M.; Qiu, H.; Bhattacharya, D.; Yoon, H.S. A novice's guide to analyzing NGS-derived organelle and metagenome data. *Algae* **2016**, *31*, 137–154.
2. Snel, B.; Bork, P.; Huynen, M.A. Genome phylogeny based on gene content. *Nat. Genet.* **1999**, *21*, 108–110.
3. Yu, N.; Yu, Z.; Li, B.; Gu, F.; Pan, Y. A Comprehensive Review of Emerging Computational Methods for Gene Identification. *J. Inf. Process. Syst.* **2016**, *12*, 1–34.
4. Zerbino, D.R.; Birney, E. Velvet: Algorithms for de novo short read assembly using de Bruijn graphs. *Genome Res.* **2008**, *18*, 821–829.
5. Miller, J.R.; Koren, S.; Sutton, G. Assembly Algorithms for Next-Generation Sequencing Data. *Genomics* **2010**, *95*, 315–327.
6. Parikesit, A.A.; Steiner, L.; Stadler, P.F.; Prohaska, S.J. Pitfalls of Ascertainment Biases in Genome Annotations—Computing Comparable Protein Domain Distributions in Eukarya. *Malays. J. Fundam. Appl. Sci.* **2014**, *10*, 64–73.
7. Besemer, J.; Lomsadze, A.; Borodovsky, M. GeneMarkS: A self-training method for prediction of gene starts in microbial genomes. Implications for finding sequence motifs in regulatory regions. *Nucleic Acids Res.* **2001**, *29*, 2607–2618.
8. Lukashin, A.V.; Borodovsky, M. GeneMark.hmm: New solutions for gene finding. *Nucleic Acids Res.* **1998**, *26*, 1107–1115.

9. Delcher, A.L.; Bratke, K.A.; Powers, E.C.; Salzberg, S.L. Identifying bacterial genes and endosymbiont DNA with Glimmer. *Bioinformatics* **2007**, *23*, 673–679.
10. Burge, C.B.; Karlin, S. Finding the genes in genomic DNA. *Curr. Opin. Struct. Biol.* **1998**, *3*, 346–354.
11. Yeh, R.F.; Lim, L.P.; Burge, C.B. Computational Inference of Homologous Gene Structures in the Human Genome. *Genome Res.* **2001**, *11*, 803–816.
12. Larsen, T.S.; Krogh, A. EasyGene—A prokaryotic gene finder that ranks ORFs by statistical significance. *BMC Bioinform.* **2003**, *4*, 21.
13. Nielsen, P.; Krogh, A. Large-scale prokaryotic gene prediction and comparison to genome annotation. *Bioinformatics* **2005**, *21*, 4322–4329.
14. Hoff, K.J.; Stanke, M. WebAUGUSTUS—A web service for training AUGUSTUS and predicting genes in eukaryotes. *Nucleic Acids Res.* **2013**, *41*, W123–W128.
15. Bairoch, A.; Apweiler, R. The SWISS-PROT protein sequence database and its supplement TrEMBL in 2000. *Nucleic Acids Res.* **2000**, *28*, 45–48.
16. Kurtz, S.; Phillippy, A.; Delcher, A.L.; Smoot, M.; Shumway, M.; Antonescu, C.; Salzberg, S.L. Versatile and open software for comparing large genomes. *Genome Biol.* **2004**, *5*, R12.
17. Delcher, A.L.; Phillippy, A.; Carlton, J.; Salzberg, S.L. Fast algorithms for large-scale genome alignment and comparison. *Nucleic Acids Res.* **2002**, *30*, 2478–2483.
18. Kim, J.I.; Yoon, H.S.; Yi, G.; Kim, H.S.; Yih, W.; Shin, W. The Plastid Genome of the Cryptomonad *Teleaulax amphioxeia*. *PLoS ONE* **2015**, *10*, e0129284.
19. Lowe, T.M.; Eddy, S.R. tRNAscan-SE: A program for improved detection of transfer RNA genes in genomic sequence. *Nucleic Acids Res.* **1997**, *25*, 955–964.
20. Lohse, M.; Drechsel, O.; Kahlau, S.; Bock, R. OrganellarGenomeDRAW—A suite of tools for generating physical maps of plastid and mitochondrial genomes and visualizing expression data sets. *Nucleic Acids Res.* **2013**, *41*, W575–W581.



© 2017 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).