frontiers in Microbiology

METHODS
published: 13 February 2019
doi: 10.3389/fmicb.2019.00184

Vgas: A Viral Genome Annotation System

Kai-Yue Zhang[†], *Yi-Zhou Gao*[†], *Meng-Ze Du, Shuo Liu, Chuan Dong and Feng-Biao Guo*[*]

Centre for Informational Biology, School of Life Science and Technology, University of Electronic Science and Technology of China, Chengdu, China

The in-depth study of viral genomes is of great help in many aspects, especially in the treatment of human diseases caused by viral infections. With the rapid accumulation of viral sequencing data, improved, or alternative gene-finding systems have become necessary to process and mine these data. In this article, we present Vgas, a system combining an *ab initio* method and a similarity-based method to automatically find viral genes and perform gene function annotation. Vgas was compared with existing programs, such as Prodigal, GeneMarkS, and Glimmer. Through testing 5,705 virus genomes downloaded from RefSeq, Vgas demonstrated its superiority with the highest average precision and recall (both indexes were 1% higher or more than the other programs); particularly for small virus genomes ($\leq$10 kb), it showed significantly improved performance (precision was 6% higher, and recall was 2% higher). Moreover, Vgas presents an annotation module to provide functional information for predicted genes based on BLASTp alignment. This characteristic may be specifically useful in some cases. When combining Vgas with GeneMarkS and Prodigal, better prediction results could be obtained than with each of the three individual programs, suggesting that collaborative prediction using several different software programs is an alternative for gene prediction. Vgas is freely available at http://cefg.uestc.cn/vgas/ or http://121.48.162.133/vgas/. We hope that Vgas could be an alternative virus gene finder to annotate new genomes or reannotate existing genome.

Keywords: Vgas, virus gene prediction, function annotation, novel genes, joint application of multiple programs

OPEN ACCESS

Edited by:
Andrew S. Lang,
Memorial University of Newfoundland,
Canada

Reviewed by:
Claire Bertelli,
Lausanne University Hospital (CHUV),
Switzerland
Vladislav Victorovich Khrustalev,
Belarusian State Medical University,
Belarus

****Correspondence:***
Feng-Biao Guo
fbguo@uestc.edu.cn

[†]*These authors have contributed equally to this work*

Specialty section:
This article was submitted to Virology, a section of the journal Frontiers in Microbiology

Received: *07 July 2018*
Accepted: *23 January 2019*
Published: *13 February 2019*

Citation:
Zhang K-Y, Gao Y-Z, Du M-Z, Liu S, Dong C and Guo F-B (2019) Vgas: A Viral Genome Annotation System. Front. Microbiol. 10:184. doi: 10.3389/fmicb.2019.00184

INTRODUCTION

Because of the tremendous value of in-depth studies of viral genomes for the treatment of human infectious diseases caused by viral infections, many viroinformatics resources, including web servers and databases, have been developed (Sharma et al., 2015). The number of sequenced viral genomes stored in the RefSeq database has increased more than five times from the year 2000 to 2016 with the rapid development of sequencing technologies (Brister et al., 2015). For the investigation of viral genomes, the first and most important step is to annotate genes accurately. Although wet experiments likely represent the most accurate way to annotate viral genes, the experiments are often time-consuming, and involve huge costs to deal with such enormous data. Furthermore, wet experiments may miss some genes that are expressed only in some specific conditions with the limitation of laboratory techniques. Therefore, computational methods for viral gene prediction are needed to serve as assistance and reference instruments for experimental results. Currently, there are two major groups of computational methods to achieve relatively accurate viral gene prediction: the similarity-based methods and the *ab initio* methods. Z-curve is a type of widely applied theory in gene identification (Dong et al., 2016; Guo et al., 2017). Based on the Z-curve method, we developed ZCURVE_V in 2006, an *ab initio* gene finding software program for viruses, which has helped many researchers study virus genes over the past few years (Li et al., 2013; Huang et al., 2014; Mahony et al., 2015; Harrison et al., 2016).

In the present work, we updated and furthered the system based on ZCURVE_V (Guo and Zhang, 2006) by increasing the identifying variables for the classification model and adding a BLASTp searching module for gene predicting. Through these two modifications, the newly proposed Vgas system not only achieved higher prediction accuracy than ZCURVE_V but also provided functional gene annotations for predicted genes that are homologs to genes with known functions in public databases. As an application example of Vgas, 86 novel genes were found and assigned with explicit functions, while they were missed in RefSeq annotations. We believe that Vgas may help researchers to efficiently analyze unknown viral genomes.

MATERIALS AND METHODS

The Implementation Process of Vgas

The course of implementation of one inputted viral genomic sequence for Vgas processing can be divided into five successive steps (**Figure 1**). (1) Extracting all the ORFs from the genome sequence. (2) Finding the longest ORF as the seed ORF (representative of positive samples) and creating five derived ORFs (representatives of negative samples). Changing the phase position of the seed ORF will generate two derived ORFs, and changing the phase position of the complementary strand of the seed ORF can generate three additional ORFs. All of the five ORFs would be taken as representatives of negative samples. (3) Calculating the identifying variables and then distinguishing the ORFs by Euclidean distance discrimination to obtain the preliminarily predicted genes. If a candidate ORF has a closer distance with the seed ORF than all of the five artificial ORFs based on Euclidean distance, it will be predicted preliminarily as a gene; otherwise, it will not be predicted. (4) Performing a homologous search against the RefSeq database and determining the ultimately predicted genes. Because RefSeq contains all viral proteins stored in other databases, such as SwissProt, here, we only use it as a reference protein database. For some predictions that are homologous to genes with known functions (bit score > 150, e-value $< 10^{-40}$), Vgas will transfer the functions of the latter to the predictions. In detail, Vgas will divide the preliminarily predicted genes into three groups according to the results of the BLASTp search against RefSeq viral genomes. One group of genes has the highest similarity to reference genes (bit score > 125, e-value < 0.01) and will be directly considered as the ultimately predicted genes. In contrast, some genes have the lowest similarity to reference genes (bit score < 31) and will be immediately eliminated. The remaining genes with medium similarity constitute the third group and will enter the next step. (5) Dealing with overlapping genes: these retained genes will be refined according to their overlapping ratios with longer genes. Consistent with ZCURVE_V, in comparing two overlapping ORFs, if the coding potential score of the longer ORF cut down by the given value is still higher than the shorter ORF, it will be recognized as a gene, and the shorter ORF will be considered non-coding. Otherwise, both ORFs are retained as coding ORFs. After this final step, all of the predicted genes can be determined.

It should be noted that we used only the 30 viruses listed in our previous work (Guo and Zhang, 2006) to set the parameters in our software; no other viruses were involved in the algorithm creation process.

Compared with eukaryotes and bacteria, there are a few characteristic properties for viral genes, and these differences will make it necessary to modify gene-finding algorithms devised for bacteria to fit viruses. On the one hand, viruses have folds-fewer gene numbers than cellular organisms. This feature makes it very difficult to choose large enough numbers of highly reliable seed ORFs. Without a number of training samples, it would be impossible to use machine-learning and other commonly used methods to construct classifying models. In Vgas, we chose the longest ORFs as the seed ORFs and used single center Euclidean discrimination to classify genes and negative samples. On the other hand, viral genes often tend to overlap each other, which makes it somewhat difficult differentiate true genes from those ORFs located in the shadow regions of true genes. High false-positive rates will appear if we choose to consider overlapping genes or ORFs. To deal with this problem, we generated five additional negative samples in the Vgas training set.

Three Main Improved Features Compared With ZCURVE_V

To obtain higher sensitivity, we added 12 extra variables to ZCURVE_V; the original 33 variables are described as Equation (1):

$$\left\{\begin{array}{l} x_1, y_1, z_1 \\ x_2, y_2, z_2 \\ x_3, y_3, z_3 \end{array}\right. \quad \left\{\begin{array}{l} x_{12}^A, y_{12}^A, Z_{12}^A \\ x_{12}^C, y_{12}^C, Z_{12}^C \\ x_{12}^G, y_{12}^G, Z_{12}^G \\ x_{12}^T, y_{12}^T, Z_{12}^T \end{array}\right. \quad \left\{\begin{array}{l} x_{23}^A, y_{23}^A, z_{23}^A \\ x_{23}^C, y_{23}^C, z_{23}^C \\ x_{23}^G, y_{23}^G, z_{23}^G \\ x_{23}^T, y_{23}^T, z_{23}^T \end{array}\right. \tag{1}$$

where each of the three symbols, x, y, z, with subscripts and superscripts, denotes one variable, which can be the frequencies of the four nucleotides A, C, G, T, as in Equations (2, 3).

$$\left\{\begin{array}{l} x_i = \left(a_i + g_i\right) - \left(c_i + t_i\right), \\ y_i = \left(a_i + c_i\right) - \left(g_i + t_i\right), \\ z_i = \left(a_i + t_i\right) - \left(g_i + c_i\right), \end{array}\right. \tag{2}$$

$$\begin{array}{l} \left\{\begin{array}{l} x_k^x = \left(p_k\left(XA\right) + p_k\left(XG\right)\right) - \left(p_k\left(XC\right) + p_k\left(XT\right)\right), \\ y_k^x = \left(p_k\left(XA\right) + p_k\left(XC\right)\right) - \left(p_k\left(XG\right) + p_k\left(XT\right)\right), \\ z_k^x = \left(p_k\left(XA\right) + p_k\left(XT\right)\right) - \left(p_k\left(XG\right) + p_k\left(XT\right)\right), \end{array}\right. \\ X = A, C, G, T \qquad k = 12, 23, 31 \end{array} \tag{3}$$

In the above Equation (2), i denotes the codon position of one of the four mononucleotides located at a gene or negative sample,

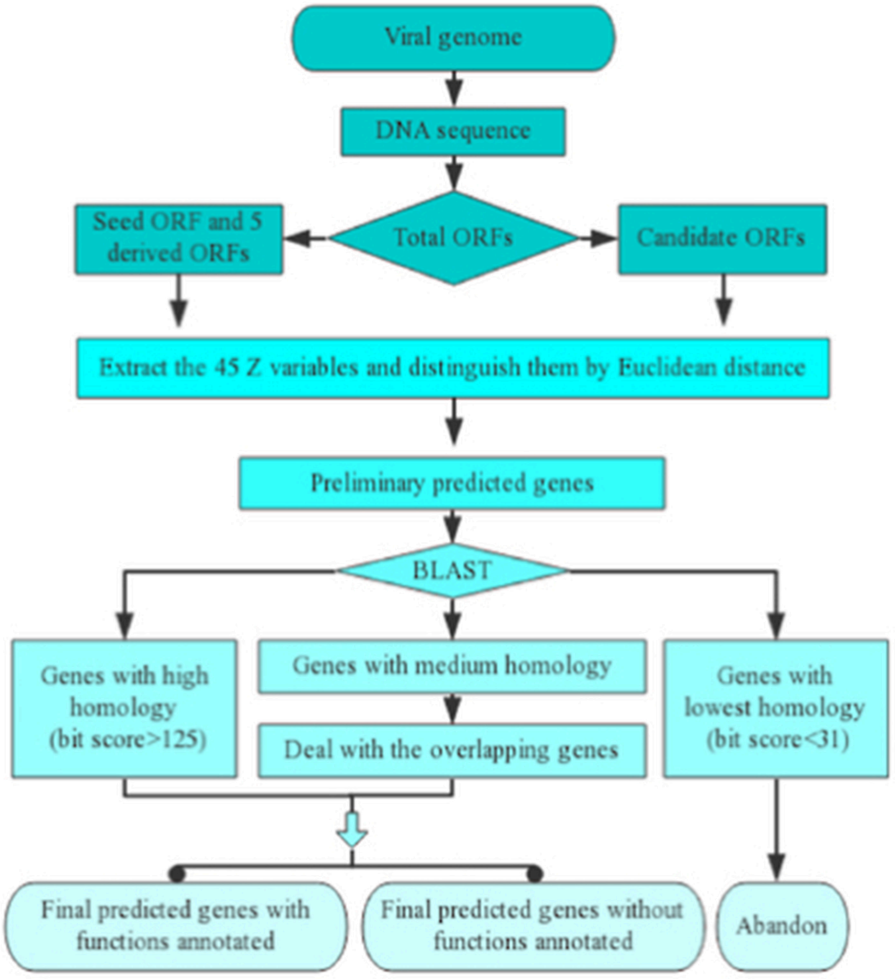
Viral genome
DNA sequence
Seed ORF and 5 derived ORFs
Total ORFs
Candidate ORFs
Extract the 45 Z variables and distinguish them by Euclidean distance
Preliminary predicted genes
BLAST
Genes with high homology (bit score>125)
Genes with medium homology
Deal with the overlapping genes
Genes with lowest homology (bit score<31)
Final predicted genes with functions annotated
Final predicted genes without functions annotated
Abandon

FIGURE 1 | Implementation process of Vgas for one inputted viral genomic sequence. As shown, there are five successive steps.

with a total of three codon positions. The lowercase *a*, *c*, *g*, and *t* with subscripts denote the frequency of each mononucleotide occurring at a given codon position. At each codon position, the sum of *a*, *c*, *g*, and *t* will be 1. Obviously, Equation (2) describes 9 total variables, which were transformed from the 12 frequency values of the four mononucleotides at three codon positions.

In the above Equation (3), *k* denotes the codon positions of one of 16 dinucleotides located at a gene or negative sample. At that time, we only considered short-distance association; codon position 31 was not involved because it corresponds to dinucleotides spanning two codons. The lowercase p denotes the frequency of one dinucleotide at a given codon position. There are a total of 32 frequency values for dinucleotides at 12 and 23 codon positions. With a similar transformation as used for Equation (2), these 32 values change to 24 variables.

In this work, we added another 12 variables, which are derived from 16 frequency values of dinucleotides at 31 codon positions, as in Equation (4):

$$t_i \begin{cases} x_{31}^{A}, y_{31}^{A}, z_{31}^{A} \\ x_{31}^{C}, y_{31}^{C}, z_{31}^{C} \\ x_{31}^{G}, y_{31}^{G}, z_{31}^{G} \\ x_{31}^{T}, y_{31}^{T}, z_{31}^{T} \end{cases} \tag{4}$$

We have now adopted 45 identifying variables instead of the 33 variables in ZCURVE_V 1.0 to represent a gene or negative sample sequence.

As the second improvement, we use five negative samples, which were all derived from the seed ORF. As an improvement over ZCURVE_V, two negative samples are derived via changing the phase position of the seed ORF. In addition, we obtained another three negative samples by changing the phase position of the complementary strand sequence of the seed ORF. In other words, the seed ORF contains six frames; the frame with the correct start and direction constitute the positive sample, whereas the other five frames are taken as negative samples. When calculating variable values, we excluded stop codons for all ORF sequences and positive/negative samples because five negative samples derived from the longest ORF would not adopt standard start and stop codons.

Third, we also utilize the BLASTp (Camacho et al., 2009) searching method to eliminate some false-positive predictions of hypothetical proteins and to assign functions to genes that are homologous to annotated genes with known functions. The E-value thresholds of the two operations are very different because one operation involves transferring functional information, while the other only involves deciding the coding potential. Please also note that we provide an option for predicting translation start sites. For this reason, we use the subprogram GS-finder to assign start sites for predicted genes. It has rather reliable predictions of translation start sites and correctly assigned 90% of 5 experimental sets of termini from *E. coli* and *B. subtilis* (Ou et al., 2004).

Three Indexes Used to Evaluate Prediction Results

We used the following three indexes to evaluate our work:

$$precision = \frac{TP}{TP + FP} \tag{5}$$

$$recall = \frac{TP}{TP + FN} \tag{6}$$

$$F - score = \frac{2 * precision * recall}{precision + recall} \tag{7}$$

where the TP denotes the number of genes that were correctly predicted by the program, FP denotes the number of ORFs that were wrongly predicted as genes by the program, and FN denotes the number of genes that should be found by the program but were missed. If the translation terminal position of a prediction is consistent with the record in the database, we assume this annotated gene have been correctly predicted regardless of their overlapping ratio. Obviously, the higher these three indexes, the better the program performed. F-score, as a comprehensive index combining precision and recall, is a main standard reference for evaluating the performance of the software's prediction algorithm.

Construction of the Benchmark Datasets

We used the RefSeq annotations of 5,705 viruses as the major benchmark to evaluate the prediction performance of four gene-finding programs. These viruses could be classified into five groups: ssDNA (830 viruses), dsDNA (2028), ssRNA (1629), dsRNA (809), and unknown (409); the viruses can subdivided into hundreds of families, with *Siphoviridae* (687) representing the largest family of our dataset. All strains of one virus will be tested if several strains are available. RefSeq's annotation may have bias to varied extents. However, no other datasets could be taken as benchmarks for large-scale measurements. To compensate for the drawback of RefSeq, we additionally constructed another benchmark of 100 viruses that have higher quality annotations. UniProtKB provides protein function annotations, and all entries correspond to validated or curated proteins. Based on this database, we could rank our 5,705 viruses in descending order of proportion of curated protein fraction. We then classified all viruses into 100 groups according to genome size. For viruses in the 100 groups, we chose those with the highest curated protein fraction as the representative of each group.

RESULTS AND DISCUSSION

The Comparison Results Based on Different Test Sets

To perform an objective evaluation, we tested 5,705 viral genomes downloaded from RefSeq[1] (in March 2017). During the

[1]NCBI FTP Site for Viruses. Available online at: ftp://ftp.ncbi.nlm.nih.gov/genomes/Viruses/

TABLE 1 | The average prediction performance levels of Vgas, ZCUVE_V, Prodigal, and GeneMarkS for different genome sizes.

	Software		Vgas (%)	ZCURVE_V (%)	Prodigal (%)	GeneMarkS (%)	Combined (%)
	All genomes (5705)	Precision	88.60	87.70	87.63	86.68	92.19
		Recall	92.22	81.06	91.10	87.97	93.68
		F-score	88.91	81.68	87.68	85.33	91.80
	Phages (1418)	Precision	80.87	86.16	93.61	92.12	93.14
		Recall	92.27	80.66	94.71	94.75	95.13
		F-score	85.60	82.87	93.84	93.07	93.80
Other viruses	Small genomes	Precision	91.86	88.29	85.24	84.38	92.25
	(3514)	Recall	92.95	82.31	90.50	85.56	93.65
	(≤10 kb)	F-score	90.76	81.70	85.60	82.20	91.51
	Medium genomes	Precision	88.23	88.34	86.63	86.60	89.21
	(459)	Recall	87.89	75.09	86.95	85.62	90.97
	(10–30 kb)	F-score	85.78	77.47	84.78	84.17	88.51
	Large genomes	Precision	87.67	91.07	88.81	88.00	91.55
	(314)	Recall	90.00	77.53	87.60	87.81	91.49
	(>30 kb)	F-score	87.72	82.26	87.45	87.22	90.89

TABLE 2 | The average prediction performance levels of Vgas, ZCUVE_V, Prodigal and GeneMarkS for different genome types.

Software		Vgas (%)	ZCURVE_V (%)	Prodigal (%)	GeneMarkS (%)	Combined (%)
ssDNA (830)	Precision	87.96	78.42	74.20	74.01	90.39
	Recall	90.90	79.76	88.68	85.77	91.65
	F-score	87.83	75.81	78.30	76.69	89.72
dsDNA (2028)	Precision	83.12	87.05	90.26	89.00	92.29
	Recall	88.75	75.62	89.97	90.16	92.53
	F-score	84.20	78.92	89.20	88.65	91.65
ssRNA (1629)	Precision	90.13	88.18	85.29	83.92	89.27
	Recall	95.30	83.87	92.74	86.18	95.37
	F-score	91.13	82.65	86.81	82.37	90.64
dsRNA (809)	Precision	98.67	98.28	97.87	97.87	98.72
	Recall	98.71	95.68	96.36	94.87	98.66
	F-score	98.26	95.92	96.37	95.32	98.35
Unknown (409)	Precision	91.09	89.50	90.84	89.78	94.02
	Recall	86.86	70.09	84.71	79.07	86.98
	F-score	87.04	74.90	85.17	78.48	88.42

process of constructing and training the Vgas prediction model, every virus involved was considered as an independent test set. We ran Vgas for each virus to obtain gene prediction results. In the BLASTp search procedure of the Vgas algorithm, we removed the query species itself from the reference database to avoid its unfair influence on the prediction.

We used the same dataset to test four other software programs as a comparison: ZCURVE_V, Prodigal (meta option) (Hyatt et al., 2012), GeneMarkS (Borodovsky and Lomsadze, 2014), and Glimmer (Delcher et al., 1999). The same indexes were used to measure the results. It should be noted that 1,041 viruses out of 5,705 viruses are so small that Glimmer could not achieve any gene prediction; therefore, we excluded this program from the assessment and only compared the performances of the other three programs.

In the practical process of bacterial gene annotation, several programs are often used to obtain collaborative predictions, which will usually be better than the predictions derived from individual programs (McHardy et al., 2004). In our previous work (Guo and Zhang, 2006), we also demonstrated that better results could be obtained for virus genomes when we jointly used ZCUVE_V and GeneMark. Here, we again test this strategy by combining Vgas, Prodigal, and GeneMarkS after running each of the programs individually. In the case of joint applications, if more than two software programs found the same gene, this gene was kept in the final result; otherwise (only one program predicted this gene), the gene was removed. As shown in **Table 1**, the general results support Vgas as the best prediction program, as it showed the highest values for all three measuring indexes among the three programs. Consistent with our inference, the joint strategy provided better predictions than only Vgas. We then divided all 5,705 viruses into four groups: phage, small genomes ($\leq$10 kb), medium genomes (10–30 kb) and large genomes (>30 kb). In this case, Vgas' performance fell short for phages but still achieved the best index values for the other three groups. Especially in the small virus group, Vgas greatly surpassed the other two programs, with 6% higher precision, 2% higher recall, and a 5% higher F-score. Given that this group occupies most (3,514/5,705 = 61.6%) of the virus size profile, Vgas should be the preferred choice for practical annotation, particularly in the case of small-size viruses. Additionally, it is most sensible if Vgas is used jointly with Prodigal and/or GeneMarkS.

We also classified the dataset of 5,705 viruses into five groups including ssDNA (830 viruses), dsDNA (2028), ssRNA (1629), dsRNA (809), and unknown (409). The group of unknown represents those viruses that have not yet been classified into any of above other four groups in GenBank. As shown in **Table 2**, we can see that Vgas achieved the highest F-scores in the group for ssDNA (87.83%), ssRNA (91.13%), dsRNA (98.26%), and unknown (87.04%) among the four software programs; in ssDNA in particular, the performance of Vgas was markedly better than that of the other three. For the dsDNA group, although Vgas did not perform the best, the results were acceptable, and its performance was greatly improved over the original ZCURVE_V, with an F-score of 84.20% compared with 78.92%.

For further confirmation of Vgas' performance, we repeated the above test based on a curated dataset with 100 viruses, instead of RefSeq. As shown in **Table 3**, Vgas obtained the highest F-score of 83.6%, higher than Prodigal and GeneMarkS (79.64 and 78.57%, respectively). Additionally, collaborative prediction was better than the use of any one software program.

Comparison of the Annotation Function of Vgas With Other Software Programs

Some software programs are widely used for annotating prokaryotic genomes, such as Prokka (Seemann, 2014) and RAST (Aziz et al., 2008). The former uses Prodigal 2.6 and the latter uses Glimmer3 as their *ab initio* gene-finding programs. Hence, their performance for finding viral genes will be generally worse than Vgas, as shown in **Tables 1–3**. Like the two annotating

TABLE 3 | The average performance levels of Vgas, GeneMarkS and Prodigal for the curated viruses set.

Software	Vgas (%)	GeneMark (%)	Prodigal (%)	Combined (%)
Precision	79.08	73.00	73.28	78.96
Recall	93.82	90.00	93.15	94.53
F-score	83.60	78.57	79.64	84.24

systems, Vgas can also assign functions to genes that are highly similar to experimentally known genes referenced in public databases. After we checked a few widely studied viruses, such as HIV and HBV, it was found that Vgas could assign functional information for more genes than Prokka and RAST. Although these two systems could be used for virus gene annotation, they should be principally devised for bacteria. Therefore, their reference genome databases may not contain gene sequences from sequenced viruses. We believe that this deficiency may result in the inconsistent performance of the two systems compared with Vgas. In a nutshell, Vgas is more suitable for viral gene annotation.

The Analysis of Predicted Novel Genes

Here, we comprehensively analyzed additional predicted genes by Vgas for 5,705 viruses. After performing BLASTp against the GenBank and RefSeq databases, we found that 86 identified predictions were highly similar to previously annotated genes (bit score > 150, *e*-value $< 10^{-40}$), and the functions of these genes' products had been experimentally validated or reliably inferred. Information for all of the 86 newly predicted genes and their most similar genes from the RefSeq database are listed in **Supplementary Table 1**. Among them, 46 do not have any overlapping nucleotides with other genes in the same genome. Most of the remaining genes have sequence-overlapping ratios of <10, indicating that they are predicted to be coding genes based on their coding potentials rather than falling in the shadow regions of longer genuine genes. Meanwhile, these genes are homologs of functionally validated genes, and it is believed that all or at least the majority of them truly encode proteins. Note that these 86 genes were newly identified by us and have not been previously reported in RefSeq. Furthermore, GeneMarkS missed 52 of the newly predicted genes, and Prodigal missed nine of them. Our identification of these genes would provide help for studies comparing related viruses. These results illustrate that Vgas could be used to find novel genes from previously annotated viral genomes, enriching the gene set.

CONCLUSION

Vgas, as an improved version of ZCURVE_V, combines an *ab initio* method and a similarity-based method to automatically find viral genes and annotate the gene functions. Systematic tests illustrated that the program was competitive with extant programs, such as Prodigal and GeneMarkS. Vgas can also be jointly used with other programs to improve the performance of single gene finders. As an application example of the new system, 86 novel genes were identified and assigned explicit functions when we validated our program on 5,705 test viruses. We hope that Vgas could be an alternative virus gene finder to annotate new genomes or to reannotate extant genomes.

AUTHOR CONTRIBUTIONS

F-BG designed and coordinated this project, and revised the manuscript. Y-ZG programmed Vgas, and K-YZ checked the results. K-YZ and Y-ZG drafted the manuscript. CD improved the Vgas algorithm. M-ZD and SL took part in the data analyses. All the authors read and approved this manuscript.

FUNDING

This work was supported by the National Natural Science Foundation of China [31871335], the Fundamental Research Funds for the Central Universities of China [ZYGX2016J117], and the scientific platform improvement project of UESTC.

ACKNOWLEDGMENTS

We would like to thank Prof. Chun-Ting Zhang for inspiring discussions and De-Chang Yang for giving us some useful suggestions about our web page.

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fmicb.2019.00184/full#supplementary-material

Supplementary Table 1 | Details of the 86 novel genes identified by Vgas that were missed in RefSeq annotations.

REFERENCES

Aziz, R. K., Bartels, D., Best, A. A., DeJongh, M., Disz, T., Edwards, R. A., et al. (2008). The RAST Server: rapid annotations using subsystems technology. *BMC Genomics* 9:75. doi: 10.1186/1471-2164-9-75

Borodovsky, M., and Lomsadze, A. (2014). Gene identification in prokaryotic genomes, phages, metagenomes, and EST sequences with GeneMarkS suite. *Curr. Protoc. Microbiol.* 32, Unit 1E 7. doi: 10.1002/9780471729259.mc01e07s32

Brister, J. R., Ako-Adjei, D., Bao, Y., and Blinkova, O. (2015). NCBI viral genomes resource. *Nucleic Acids Res.* 43, D571–577. doi: 10.1093/nar/gku1207

Camacho, C., Coulouris, G., Avagyan, V., Ma, N., Papadopoulos, J., Bealer, K., et al. (2009). BLAST+: architecture and applications. *BMC Bioinformatics* 10:421. doi: 10.1186/1471-2105-10-421

Delcher, A. L., Harmon, D., Kasif, S., White, O., and Salzberg, S. L. (1999). Improved microbial gene identification with GLIMMER. *Nucleic Acids Res.* 27, 4636–4641.

Dong, C., Yuan, Y. Z., Zhang, F. Z., Hua, H. L., Ye, Y. N., Labena, A. A., et al. (2016). Combining pseudo dinucleotide composition with the Z curve method to improve the accuracy of predicting DNA elements: a case study in recombination spots. *Mol. Biosyst.* 12, 2893–2900. doi: 10.1039/c6mb00374e

Guo, F. B., Dong, C., Hua, H. L., Liu, S., Luo, H., Zhang, H. W., et al. (2017). Accurate prediction of human essential genes using only nucleotide composition and association information. *Bioinformatics* 33, 1758–1764. doi: 10.1093/bioinformatics/btx055

Guo, F. B., and Zhang, C. T. (2006). ZCURVE_V: a new self-training system for recognizing protein-coding genes in viral and phage genomes. *BMC Bioinformatics* 7:9. doi: 10.1186/1471-2105-7-9

Harrison, R. L., Rowley, D. L., and Funk, C. J. (2016). The Complete Genome sequence of plodia interpunctella granulovirus: evidence for horizontal gene transfer and discovery of an unusual inhibitor-of-apoptosis gene. *PLoS ONE* 11:e0160389. doi: 10.1371/journal.pone.0160389

Huang, H., Dong, Y., Yang, Z. L., Luo, H., Zhang, X., and Gao, F. (2014). Complete sequence of pABTJ2, a plasmid from *Acinetobacter baumanni*i MDR-TJ, carrying many phage-like elements. *Genomics Proteomics Bioinformatics* 12, 172–177. doi: 10.1016/j.gpb.2014.05.001

Hyatt, D., LoCascio, P. F., Hauser, L. J., and Uberbacher, E. C. (2012). Gene and translation initiation site prediction in metagenomic sequences. *Bioinformatics* 28, 2223–2230. doi: 10.1093/bioinformatics/bts429

Li, W., Zhang, J., Chen, Z., Zhang, Q., Zhang, L., Du, P., et al. (2013). The genome of VP3, a T7-like phage used for the typing of *Vibrio cholera*e. *Arch. Virol.* 158, 1865–1876. doi: 10.1007/s00705-013-1676-9

Mahony, J., Randazzo, W., Neve, H., Settanni, L., and van Sinderen, D. (2015). Lactococcal 949 group phages recognize a carbohydrate receptor on the host cell surface. *Appl. Environ. Microbiol.* 81, 3299–3305. doi: 10.1128/AEM.00143-15

McHardy, A. C., Goesmann, A., Puhler, A., and Meyer, F. (2004). Development of joint application strategies for two microbial gene finders. *Bioinformatics* 20, 1622–1631. doi: 10.1093/bioinformatics/bth137

Ou, H. Y., Guo, F. B., and Zhang, C. T. (2004). GS-Finder: a program to find bacterial gene start sites with a self-training method. *Int. J. Biochem. Cell Biol.* 36, 535–544. doi: 10.1016/j.biocel.2003.08.013

Seemann, T. (2014). Prokka: rapid prokaryotic genome annotation. *Bioinformatics* 30, 2068–2069. doi: 10.1093/bioinformatics/btu153

Sharma, D., Priyadarshini, P., and Vrati, S. (2015). Unraveling the web of viroinformatics: computational tools and databases in virus research. *J. Virol.* 89, 1489–1501. doi: 10.1128/JVI.02027-14

Conflict of Interest Statement: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.