BIOINFORMATICS ORIGINAL PAPER

Vol. 28 no. 17 2012, pages 2223–2230
doi:10.1093/bioinformatics/bts429

Genome analysis

Advance Access publication July 12, 2012

Gene and translation initiation site prediction in metagenomic sequences

Doug Hyatt[1,2,*], Philip F. LoCascio[1], Loren J. Hauser[1,2] and Edward C. Uberbacher[1,2]

[1]Computational Biology and Bioinformatics Group, Oak Ridge National Laboratory, Oak Ridge, TN 37831, USA and [2]Genome Science and Technology School, University of Tennessee, Knoxville, TN 37996, USA

Associate Editor: Martin Bishop

ABSTRACT

Motivation: Gene prediction in metagenomic sequences remains a difficult problem. Current sequencing technologies do not achieve sufficient coverage to assemble the individual genomes in a typical sample; consequently, sequencing runs produce a large number of short sequences whose exact origin is unknown. Since these sequences are usually smaller than the average length of a gene, algorithms must make predictions based on very little data.

Results: We present MetaProdigal, a metagenomic version of the gene prediction program Prodigal, that can identify genes in short, anonymous coding sequences with a high degree of accuracy. The novel value of the method consists of enhanced translation initiation site identification, ability to identify sequences that use alternate genetic codes and confidence values for each gene call. We compare the results of MetaProdigal with other methods and conclude with a discussion of future improvements.

Availability: The Prodigal software is freely available under the General Public License from http://code.google.com/p/prodigal/.

Contact: hyattpd@ornl.gov

Supplementary Information: Supplementary data are available at *Bioinformatics* online.

Received on March 20, 2012; revised on July 2, 2012; accepted on July 3, 2012

1 INTRODUCTION

Metagenomes from environmental samples can contain thousands of species and often cannot be sequenced to sufficient coverage to assemble each individual genome. Even with enough coverage, the correct binning and assembly of the various sequences still present many challenges, making it likely that at least for the immediate future, metagenomic sequencing will continue to produce a large number of small contigs.

1.1 Challenges in short, anonymous sequences

Various sequencing technologies, such as 454, Illumina and Sanger, can produce reads anywhere from 50 to >1000 bp when analyzing a typical metagenomic sample. In the first case, gene identification becomes extremely difficult; in the latter case, genes can be predicted rather well. In addition, sequencing errors, particularly the insertions and deletions common to 454, can have a profound negative impact on metagenomic gene prediction (Hoff, 2009; Rho *et al.*, 2010).

In reality, there are two problems in metagenomic gene prediction. The first problem, which we call the anonymous sequence problem, is that the genome from which the sequence was derived is unknown. The second problem, which we will refer to as the short sequence problem, is that the sequences are shorter than the length of an average gene, and therefore, many fragments contain genes that run off one or both edges of the contig. Although many methods treat these two problems together, and, indeed, the second problem does exacerbate the first, in reality, they are two separate issues. Short sequences present challenges even for draft contigs in a single genome, particularly in the identification of edge genes, and long sequences whose origin is unknown can still prove difficult to analyze as accurately as if the genome was known.

Many programs have been developed to solve these problems and identify genes in metagenomic fragments, including Metagene (Noguchi *et al.*, 2006), Metagene Annotator (Noguchi *et al.*, 2008), MetaGeneMark (Zhu *et al.*, 2010), Orphelia (Hoff *et al.*, 2008) and FragGeneScan (Rho *et al.*, 2010). Although these methods perform well, none of them specializes in identifying translation initiation sites and none of them is able to correctly identify sequences derived from the *Mycoplasma* genus, which uses an alternate genetic code that translates UGA to tryptophan (Yamao *et al.*, 1985).

1.2 The Prodigal gene prediction program

The gene prediction program Prodigal was introduced in 2007 (Hyatt *et al.*, 2010). Prodigal achieves good performance in identifying genes and translation initiation sites in finished genomes (Angelova *et al.*, 2010; de Jong *et al.*, 2010; Hyatt *et al.*, 2010). The Joint Genome Institute uses Prodigal to annotate all its draft and finished genomes for the Department of Energy. Prodigal has been downloaded over 1000 times by users from 56 different countries and is in active use at numerous institutions around the world (data provided by analytics.google.com).

Because Prodigal's training methodology already incorporates a great deal of information, including translation table, hexamer statistics, ribosomal binding site (RBS) motifs and upstream base composition, we sought to create extensions to the existing software that would handle metagenomic gene prediction, rather than to begin from nothing. Our idea was to create a variety of Prodigal training files covering all ranges of GC content, genetic codes, Eubacteria, Archaea, etc. and analyze an incoming

*To whom correspondence should be addressed.

fragment using one or more of these files. The resulting prediction would then be chosen based on the training file(s) that provided the best fit for a particular sequence.

1.3 The focus of MetaProdigal

In developing a metagenomic version of Prodigal, we chose to focus on optimizing performance for longer sequence lengths (700 bp+), in the belief that sequencing, binning and assembly technologies will rapidly improve to the point where extremely short sequences are no longer the norm. Despite this focus, we still ensured our algorithm would perform reasonably well on shorter sequences.

In addition, although we acknowledge the severe impact of sequencing errors on gene prediction, it proved too difficult to integrate the handling of insertions and deletions into the Prodigal framework. We also assert that frame shifts will become increasingly less of a problem with future improvements to sequencing and assembly technologies. In the meantime, FragGeneScan (Rho *et al*., 2010) has demonstrated robust handling of insertions and deletions for those using 454.

Our algorithm provides three novel contributions: (i) the incorporation of start site information into our training files, enabling excellent recognition of translation initiation sites, particularly at longer sequence lengths; (ii) the ability to predict genes in sequences from organisms that use an alternative genetic code (*Mycoplasma*) and (iii) the provision of confidence values, which can be used to filter gene predictions (useful when dealing with small gene fragments).

2 MATERIALS AND METHODS

The first step in our algorithm was to develop a set of training files that could be used to score an anonymous coding sequence using the existing Prodigal algorithm. To generate these training files, we turned to NCBI's Refseq repository, which, as of September 2010, contained 1415 genome sequences of 500 000 bases or greater (Pruitt *et al*., 2009). The idea was to partition all microbial Refseq into a set of clusters, where each cluster could be used to create a single training file. Rather than determining the number of clusters ahead of time, we hoped to establish a dissimilarity cutoff between clusters, such that we would halt the clustering process when the distance between the closest two clusters exceeded the established dissimilarity threshold.

Before Refseq could be partitioned into clusters, we first needed to establish a distance measure between two genomes. Although various methods already existed for measuring the distance between two genomes, we decided instead to use a novel measure to calculate the distance between two genomes based on the Prodigal. The reason for choosing this method is that we wanted something computational and not biological in nature, such that we could be certain that, from Prodigal's perspective as a computer software, two genomes in the same cluster would be truly similar. We called this new measure gene prediction similarity.

2.1 Gene prediction similarity

Prodigal can examine a single genome and record its statistics in a training file, which can then be used to analyze individual sequences from that genome. Given two genomes A and B, we can train Prodigal on genome A and then use that training file to predict genes in both genomes A and B. By examining how the predictions differ, we can measure the effective difference between the two genomes.

We trained Prodigal on all 1415 microbial Refseq sequences individually. Next, for each training file, we predicted the genes in all the 1415 genomes. This resulted in 1415 × 1415, or 2 002 225, runs, each of which took about 15 s on average, for a total of about 8000 processor hours. We performed these runs on a 64-node cluster with 512 AMD Opteron processors, enabling this run to finish in a single day.

Once we had these results, we considered the diagonal of the 1415 × 1415 matrix to be the baseline, i.e. the runs where Prodigal was trained and run on the same genome. We then needed a method for measuring the similarity between two sets of gene predictions, one in which Prodigal was trained and run on genome A and one in which Prodigal was trained on a different genome (B, for example) and run on genome A. We defined this to be the gene prediction similarity B → A.

For a given baseline prediction p, and a second set of predictions p', we considered the number of correct matches M between the two predictions to be:

$$M = (m - ((a + d)/600.0)),$$

where m is the number of genes in p and p' that share a stop codon, a is the number of bases in the second prediction not contained in the first prediction for only the genes that share a stop codon and d is the number of bases in the first prediction not contained in the second prediction for only the genes that share a stop codon. The idea was to calculate the average distance between start codons and penalize 10% for every 60 bp difference (distances >600 bp in a single gene were reduced to 600 bp, i.e. we could not penalize >100% per gene). For example, if 1500 genes that share a stop codon differed by 30 bp on average in their start site predictions, then, instead of 1500 correct identifications, we counted them as 95% of 1500 or 1425. The reason for this modification was to allow differences in translation initiation site prediction to be incorporated into the clustering model.

In addition, we made one further modification that if the predictions failed to achieve a 90% perfect match in start sites among genes that shared a stop codon, we instead labeled every mismatch as only half correct. For example, if genome B correctly predicted 1500 genes (stop codons) in A, but only 1200 of those 1500 genes (80%) matched perfectly (start and stop codon), the match count M would be set to $1200 + 300 \times 0.5 = 1350$.

The idea behind this rule was to detect cases such as *Aeropyrum pernix*, which preferentially chooses TTG as its start codon (Kawabarayasi *et al*., 1999). When using another organism to predict the genes in *A. pernix*, the second organism frequently performed quite well at finding the stop codons and would even predict genes of approximately the same size, choosing a nearby ATG whenever available (because ATG is preferred in the second organism's training file). However, only ~50–60% of the genes matched perfectly. We decided to penalize heavily for this situation, because the results indicated a substantially different preference in translation between the two organisms.

Given the above information, we next needed to normalize the above value of M to be a number between 0 and 1. Therefore, we defined the 'gene prediction similarity D(A′→A)' to be the F-score, or the harmonic mean of the sensitivity (M/n) and precision (M/n'):

$$D(A' \rightarrow A) = 2M^2/(Mn + Mn'),$$

where n is the number of genes in A and n' is the number of genes in A'. The only difference between this sensitivity and precision and that described in the Prodigal paper is that we penalized matching 3′ genes for the distance between their start site predictions. It is worth noting that gene prediction similarity is not symmetrical. Although usually, A's ability to predict the genes in B is fairly close to B's ability to predict the genes in A, quite frequently one genome will predict genes quite well in its counterpart, while the opposing genome will do quite poorly on the first one.

Table 1. Sample gene prediction similarities for *Escherichia coli K12*

Genome	NG	3′*M*	5′*M*	XB	*M*	Sn	Pr	GPS
E. coli K12	4313	4313	4313	0.0	4313.0	1.00	1.00	1.000
E. coli S88	4315	4307	4287	2.5	4304.5	0.99	0.99	0.998
S. enterica	4309	4290	4241	7.2	4282.8	0.99	0.99	0.993
Brucella melitensis	4197	4159	3991	27.2	4131.8	0.96	0.98	0.970
Helicobacter pylori	4036	4010	3746	39.7	3970.3	0.92	0.98	0.952
C. difficile	3707	3669	3379	40.1	3628.1	0.84	0.98	0.910
Aquifex aeolicus	3904	3829	3146	78.6	3487.5*	0.81	0.89	0.851
A. pernix	3282	3128	1330	399.6	2229.0*	0.52	0.68	0.598
M. bovis	3520	2459	2090	185.0	2274.5*	0.53	0.65	0.587

Table 1 shows an example of gene prediction similarity calculations between *Escherichia coli K12* and a variety of organisms. Prodigal was trained on each of these organisms and then run on *E. coli*, and the gene prediction similarity was calculated using the previously described formula. 'NG' indicates the number of genes predicted by the second training file. '3′*M*' and '5′*M*' indicate the number of genes that match a stop codon and start codon in the *E. coli* predictions. 'XB' indicates the $(d' + d')/600$ term in the match equation and indicates the number of genes we are penalizing from the final result. The next column, 'M,' represents the number of matches, which we then divide by 4313 (the number of genes in the *E. coli* prediction) to get the sensitivity and by the number in the first column (NG) to get the precision. The final gene prediction similarity is then the harmonic mean of Sn and Pr. Note that in the cases marked with an asterisk, we applied the alternative formula described above for calculating *M*, since <90% of the starts were correct (i.e. $5'M/3'M < 0.9$).

Observing Table 1, we can see that *E. coli S88* produced gene predictions extremely close to the original, which is to be expected for the same species. The highly similar *Salmonella enterica* also performed extremely well. The two Archaea proved to be quite distant, especially *A. pernix* with its TTG start motif. Finally, *Mycoplasma bovis* performed the worst of the entries in this table, due to using a completely different genetic code. *Clostridium difficile* proved interesting, as it failed to predict many real genes (~15%) in *E. coli*, but the genes it did predict were mostly correct (98% Sp).

2.2 Complete-linkage clustering of Refseq

Having obtained a reliable distance measure, we built the 1415 × 1415 distance matrix for all sequences above 500 000 bp in microbial Refseq. Note that some of these sequences were chromosomes belonging to the same genome, but we kept these separate because we found it interesting to examine gene prediction similarities within multiple chromosomes in a single genome. The distances in this matrix could be used for a variety of purposes beyond the scope of this article, such as building phylogenetic trees or establishing cutoffs to delineate species, genus and family boundaries.

Next, we clustered the sequences using an algorithm similar to complete-linkage clustering, in which, at each step, the two clusters are merged whose farthest neighbors are the closest (Massaro, 2005). We chose this method to avoid the problem of population bias in Genbank, where more strains of one organism (for example, *E. coli*) have been sequenced than another. This makes merging clusters of different sizes using various weighted average distance methods difficult.

When calculating the distance between two clusters, we examined the new cluster that would be created by merging them. For each point in the potential new cluster, we located the point farthest away from it, i.e. the one with the lowest gene prediction similarity, which corresponded to the sequence least recognized by the initial data point. We then chose the data point that had the 'best' such distance, which can be roughly thought of as the central-most point in the merged cluster. We label this sequence as the 'recognizer' of the cluster. An example cluster using these concepts is illustrated in Figure 1, in which *Pseudomonas aeruginosa* would be chosen as the recognizer for the cluster based on its worst gene prediction similarity being better than that of the other two organisms. At each step of the clustering algorithm, the two closest clusters were merged, until only one cluster comprising all 1415 sequences remained.

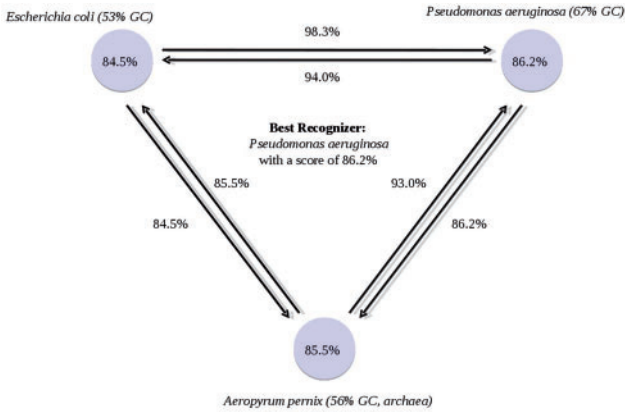

Fig. 1. Example of best worst distance and recognizer in cluster

After the clustering was completed, we examined the similarity cutoffs and found that the score dropped below 95% when going from 51 to 50 clusters. Therefore, with 50 training files, a gene prediction similarity of 95% or better would be guaranteed across all microbial GenBank. We then trained MetaProdigal on the sequences in these 50 clusters, resulting in 50 training files that could be used to recognize anonymous coding sequences. A detailed list of the 50 clusters (with their best recognizer) is provided in Supplementary Table S1.

Of the 50 genomes selected by the clustering process, 35 were bacteria and 15 were Archaea. Three of the bacteria were from the *Mycoplasma* genus, which uses translation table 4, while the remaining 47 genomes used the standard translation table. Thirty-two of the chosen genomes used Shine-Dalgarno RBS motifs (Shine and Dalgarno, 1975), and 18 genomes (many of them are Cyanobacteria, Chlorobi or Archaea) did not. GC content of the 50 genomes ranged from 29.3 to 69.8%. The five largest clusters consisted of 262, 232, 184, 130 and 98 genomes, respectively, accounting for 64% of the 1415 sequences in GenBank, with the remaining 45 clusters covering the other 36%. Despite the top five clusters being very large, the average gene prediction similarities of their recognizers was over 98.5%. The 50 genomes roughly subdivided into

3% GC intervals, with a Shine-Dalgarno-using bacterium, a non-Shine-Dalgarno bacterium (such as a Cyanobacteria or Chlorobi) and an Archaeum at each interval.

One can see from the supplementary table that many clusters are devoted to small numbers of 'unusual' genomes, with a relatively small number of clusters covering the common organisms such as *E. coli*, *Pseudomonas*, etc. In fact, 26 of the 50 clusters contained five or fewer genomes. An open question would be if devoting so many training files for recognizing such a small number of genomes is worthwhile. An alternative approach would delve into more detail on the larger clusters, splitting them further. As noted in the previous paragraph, however, recognition of even the largest clusters was already at 98.5%, so it is questionable if one could really get a significant improvement by doing so.

2.3 Using the Prodigal training files for metagenomic analysis

By using the 50 training files, an input sequence can be scored with the standard Prodigal dynamic programming algorithm for finished genomes (Hyatt *et al.*, 2010). Since the Prodigal dynamic programming function returns a numerical score, the algorithm can run an input sequence through each of the 50 training files and output only the best result. However, this approach presents two drawbacks: (i) increased computation time (50 times a normal Prodigal run) and (ii) increased false-positive rate at shorter sequence lengths due to sampling multiple training files.

To address the first drawback, MetaProdigal calculates the GC content for an incoming fragment and runs only on the training files within a given range of GC content relative to the fragment GC (a configurable parameter). To address the second drawback, we implemented a series of penalties for each gene in a sequence based on the length of the input sequence, the number of training files used to score the sequence and the length of the gene being scored. The principle of these penalties is similar to that of a Bonferroni correction (Bonferroni, 1935), in which a score is corrected based on the number of tested hypotheses (in this case, each test is a training file). Such a correction was only necessary in shorter sequences (<500 bp), where a lack of sufficient information resulted in greater volatility when using multiple training models to score a gene.

One novel contribution of our algorithm is the calculation of confidence scores for each gene. Since the MetaProdigal score represents the log of the likelihood of this gene to be real versus background (i.e. a gene 1000× more likely to be real than false would have a score of ln(1000)), the score can be converted to a percent value between 0 and 100 exclusive using the logistic function

$$C = e^s/(1 + e^s),$$

where C is the confidence value and s is the Prodigal score for that gene. We will examine the performance of this confidence score in the Results.

The algorithm for the MetaProdigal is illustrated in Figure 2. A sequence arrives on standard input, the lower and upper GC content bounds for the fragment are established, and the full dynamic programming is performed using only training files trained on genomes with GC content in the specified range. The highest scoring set of gene models is selected and output to the user, along with confidence scores for each gene and detailed information about the training file used (genetic code used, Shine-Dalgarno preferences, etc.). In addition, as in the regular version of Prodigal, protein translations, DNA sequences and detailed information about every potential start site in the sequence can be output on request.

As a result of running a full dynamic programming algorithm multiple times, which admittedly is complete overkill on short fragments, MetaProdigal is somewhat slow compared with existing programs like MetaGene Annotator and MetaGeneMark (Noguchi *et al.*, 2008; Zhu *et al.*, 2010). However, the finished genome version only took ~15–20 sec to analyze a typical 4*M* bp genome on a single processor, so, even running on five to six training files per sequence, the metagenomic version can analyze 4*M* worth of data in about 100 s. A 1 GB sample could be analyzed in 7 h on a single processor at this rate, which, in our experience, is an acceptable turnaround time, especially given the ease by which the sample could be divided and run on multiple processors.

```
FOR each sequence in sample
   READ sequence
   CALCULATE GC content of the sequence
   DETERMINE low GC and high GC bounds from fragment GC
   FOR each metagenomic training file
      IF training file GC less than low GC bound and greater than high GC bound
         DO prodigal dynamic programming using training file
         RECORD dynamic programming score if highest observed
         INCREMENT count of number of training files used
      ENDIF
   ENDFOR
   DO Bonferroni correction to scores based on training file count
   IF corrected score still greater than 0
     OUTPUT gene models from highest scoring training file
   ENDIF
ENDFOR
```

Fig. 2. Pseudocode description of the MetaProdigal algorithm

3 RESULTS

Assessing the performance of metagenomic gene prediction tools remains a difficult task, due to the lack of experimentally verified gene sets. Tools such as Metagene Annotator, MetaGeneMark, Orphelia and FragGeneScan, have compared their predicted results with GenBank annotations (Hoff *et al.*, 2008; Noguchi *et al.*, 2008; Rho *et al.*, 2010; Zhu *et al.*, 2010). By using this method, complete genomes from Refseq are sampled to a certain level of coverage at various fragment sizes (either with or without simulated errors), and the predicted results are compared with the positions of the GenBank-annotated genes in the fragments.

Unfortunately, this methodology has one significant drawback. Since the gene calls in Refseq have not been experimentally verified, it is likely some of them are incorrect. Error rates have been shown to be greater in genomes containing high GC content (Angelova *et al.*, 2010). In addition, some translation initiation site predictions are likely to be incorrect as well, which could have an impact on gene predictions as fragment sizes become smaller. Nonetheless, the chosen genomes represent a good cross section across bacteria, Archaea and all levels of GC content. Therefore, we decided to analyze the results of MetaProdigal on the 50 genome set from the MetaGeneMark publication, according to the same standards described previously (Zhu *et al.*, 2010).

3.1 Gene prediction performance on an errorless simulated dataset

In the first analysis, we measured the performance of several methods at locating genes in an errorless simulated dataset. In this dataset, we did not consider the performance of programs on identifying translation initiation sites, because the error rate of start site predictions in the Genbank files is likely too high to make such a test meaningful (we examine start site performance separately in the next section). Each of the 50 sequences in the MetaGeneMark set was randomly sampled to 5× coverage in four different fragment sizes: 150, 300, 700 and 1200 bp. Sequencing errors were not considered for this analysis. In addition, one genome was added to the 50 Refseq sequences, namely that of *Mycoplasma leachii*. Since ~2% of the finished genomes in GenBank are *Mycoplasma* (Benson *et al.*, 2011), adding one *Mycoplasma* to a set of 50 sequences seemed like a reasonable

Table 2. Performance on 51 genome sequences from Refseq

Category	MetaProdigal (%)	MetaGeneMark (%)	MetaGene Annotator (%)	Prodigal finished (%)	Combined (MP + MGmk) (%)
1200 bp sensitivity	**95.5**	95.2	94.9	95.9	93.5
1200 bp precision	**95.4**	94.0	93.6	95.8	96.9
1200 bp *F*-score	**95.4**	94.6	94.2	95.8	95.3
700 bp sensitivity	**95.1**	94.6	94.7	95.5	93.1
700 bp precision	**95.0**	94.1	92.9	95.9	96.9
700 bp *F*-score	**95.0**	94.3	93.8	95.7	95.0
300 bp sensitivity	**94.5**	93.6	94.1	95.0	91.8
300 bp precision	93.5	**94.1**	91.1	96.1	96.5
300 bp *F*-score	**94.0**	93.8	92.6	95.5	94.1
150 bp sensitivity	**92.5**	91.0	91.7	94.0	88.4
150 bp precision	90.0	**92.6**	88.1	94.9	95.1
150 bp *F*-score	91.2	**91.8**	89.9	94.4	91.6

addition. This genome was added to the set to demonstrate how MetaProdigal can distinguish between genetic code 4 (used by *Mycoplasma*) and genetic code 11 (the standard microbial code) and achieve good performance on both types of genomes. Neither MetaGeneMark nor MetaGene Annotator possesses this capability, and both programs performed poorly on the *M. leachii* genome. A complete list of the 51 sequences used to evaluate the algorithms can be found in Supplementary Table S2.

As described above, genes <60 bp, whether partial or complete, were not considered. The programs were only evaluated on their ability to predict genes of 60 bp or more in length. Regardless of the amount of coding present in a fragment, only the stop codon (or correct frame in the absence of a stop codon) and 60 bp of shared coding were required; the start codon was not required to match the predicted start in the Genbank file. For this analysis, we created a special version of MetaProdigal in which we excluded the 51 genomes in our test set from the clustering and training process; however, for the release version, these genomes were added back in to the training process. Table 2 shows the results of four methods: finished genome Prodigal, MetaProdigal, MetaGeneMark and MetaGene Annotator. The finished genome version of Prodigal (labeled as Prodigal_Finished in Table 3) was run to observe how closely the metagenomic version could match a version trained on the actual genome. In this analysis, the sensitivity, precision and *F*-score were calculated separately for each of the 51 sequences and then averaged together to produce the numbers in Table 2. We define sensitivity to be TP/(TP + FN), precision to be TP/(TP + FP) and *F*-score to be the harmonic mean of the precision and sensitivity, or $2pr/(p+r)$.

In the longer fragment lengths, it is clear that MetaProdigal performs very closely to a version trained on the actual genomes (0.4 difference in *F*-score at 1200 bp, 0.7 at 700 bp, 1.5 at 300 bp and 3.2 at 150 bp). The implication is that for longer sequence lengths, such as 2000 bp, MetaProdigal identifies genes nearly as well as if it had actually been trained on the full reference genome. At the 700 and 1200 bp sequence lengths, MetaProdigal outperforms MetaGeneMark and MetaGene Annotator both in sensitivity and precision. At 300 and 150 bp, MetaProdigal still has better sensitivity and precision than MetaGene Annotator, but MetaGeneMark achieves a lower false-positive rate and better overall accuracy at 150 bp. Based on our results, it appears that MetaProdigal performs better at sequence lengths 250–300 bp and above, whereas MetaGeneMark, due to a lower false-positive rate, achieves a better *F*-score at lengths <250 bp.

We view the two programs (MetaProdigal and MetaGene Mark) as quite complementary on smaller sequences, however, as MetaProdigal seems to preserve sensitivity as sequence lengths grow shorter, whereas MetaGeneMark sacrifices sensitivity to preserve precision. At all four sequence lengths, overall accuracy was within 1% between MetaProdigal and MetaGeneMark, which may well lie in the margin of error based on incorrectly called genes in the Refseq annotations. However, it is likely that both programs perform better than MetaGene Annotator at identifying genes.

In the particular case of *M. leachii*, the genome we added to the MetaGeneMark set, the MetaProdigal achieved 95.3% sensitivity and 94.3% precision even in the 150 bp fragments, whereas MetaGeneMark and MetaGene Annotator managed only 78.1 and 83.6% sensitivity, respectively. In 1200 bp fragments, MetaGeneMark and MetaGene locate most of the stop codons, but, because *Mycoplasma* translates TGA, they often only find the 3′-end of the gene and split the true gene into many smaller genes. Therefore, the precision in 1200 bp fragments for MetaGeneMark and MetaGene was only 69 and 66%, respectively, whereas MetaProdigal had 98% sensitivity and 97.3% precision MetaProdigal can distinguish anonymous coding sequences using the *Mycoplasma* genetic code without sacrificing performance on the sequences that use the standard genetic code, which is a novel capability of the program compared with other methods.

Recent publications have considered combining gene prediction methods for better results (Yok and Rosen, 2011). Although examining more elaborate methods of combining MetaProdigal and MetaGeneMark gene predictions is beyond the scope of this article, however, we did benchmark the performance of the intersection of the gene sets predicted by each program. These data are presented in the 'Combined' column in Table 2. Although

Table 3. Performance on 2443 experimentally verified genes and start sites

Length (bp)	Type	% Total	MetaProdigal (%)	MetaGeneMark (%)	MetaGene Annotator (%)	MGA + Meta TISA (%)	Prodigal finished (%)
3000	Internal	77.4	**93.5**	86.3	87.6	93.3	96.3
3000	External	23.6	**99.8**	98.3	94.2	83.4	99.8
1200	Internal	56.9	**91.6**	85.3	86.7	90.5	95.0
1200	External	43.1	**99.8**	98.8	94.1	82.9	99.8
700	Internal	42.4	**89.5**	83.5	85.6	86.7	93.7
700	External	57.6	**99.8**	99.2	94.0	82.8	99.8
300	Internal	21.7	**82.1**	77.1	80.1	71.0	88.2
300	External	78.3	**99.7**	99.0	93.8	81.3	99.8
150	Internal	9.4	66.0	60.4	**66.5**	26.9	75.2
150	External	90.6	**99.6**	98.9	94.0	80.0	99.8

Table 4. Prodigal confidence estimations for 51 genome sequences from Refseq

Fragment length (bp)	Confidence (%)	Real genes	False genes	% Real	% False	Sn	Pr	F-score
700	**100**	854 622	6727	**99.2**	0.8	60.0	99.2	79.6
700	**90–99**	441 336	23 011	**95.0**	5.0	90.9	97.8	94.3
700	**80–89**	35 547	10 648	**76.9**	23.1	93.4	97.1	95.2
700	**70–79**	19 307	9512	**67.0**	33.0	94.8	96.4	95.6
700	**60–69**	10 200	7311	**58.2**	41.8	95.5	96.0	95.8
700	**50–59**	5796	6212	**48.3**	51.7	95.9	95.6	95.8
300	**100**	772 854	5061	**99.3**	0.7	29.7	99.3	64.5
300	**90–99**	1 552 693	73 289	**95.5**	4.5	89.4	96.7	93.1
300	**80–89**	78 026	33 410	**70.0**	30.0	92.4	95.6	94.0
300	**70–79**	41 307	32 911	**55.7**	44.3	94.0	94.4	94.2
300	**60–69**	15 789	17 670	**47.2**	52.8	94.6	93.8	94.2
300	**50–59**	7373	11 673	**38.7**	61.3	94.8	93.4	94.1

sensitivity dropped, the precision of the predictions improved dramatically, exceeding even that of the finished genome version of Prodigal. Even at 150 bp, the precision of the set of genes predicted by both MetaGeneMark and MetaProdigal remained above 95%. These data highlight the advantages of using multiple methods to obtain a set of high confidence gene models. Even though MetaProdigal's performance is similar to MetaGeneMark's individually, the inclusion of another method still provides substantial value.

3.2 Translation initiation site prediction performance on an experimentally verified gene set

Start site identification in metagenomic sequences has not been studied much in the literature, although programs such as MetaTISA have been built to address this problem (Hu *et al.*, 2009). Although the primary focus remains finding the genes themselves, it is still desirable to locate as many translation initiation sites correctly as possible. The problem is complicated by the fact that some organisms use Shine-Dalgarno ribosomal binding site (RBS) motifs, whereas others, such as Cyanobacteria and Chlorobi, do not appear to use RBS motifs at all (Hyatt *et al.*, 2010). Regardless of the presence or absence of an RBS motif, one of the 50 training files used in the metagenomic version of Prodigal will likely assign that start site a positive score, because both SD and non-SD organisms are included.

To assess start site performance, we took the dataset from the Prodigal publication, containing 2443 genes (Aivaliotis *et al.*, 2009; Hyatt *et al.*, 2010; Rudd, 2000). The genomes were randomly sampled in five fragment sizes: 150, 300, 700, 1200 and 3000 bp, with the restriction that the fragment must contain at least 60 bp of 1 of the 2443 experimentally verified genes. We added the longer fragment size to illustrate the continuing increase in start site accuracy as more information becomes available. Again, for this analysis, we used a special version of MetaProdigal that had not been trained on any of the genomes in this dataset; however, we did include these genomes in the training process for the final release version (resulting in much higher performance on some of the Archaea). The performance on this dataset is given in Table 3. The results of the regular version of Prodigal are again shown for comparison (in the 'Prodigal Finished' column) as a best achievable result for the metagenomic program.

Accuracy in start site prediction was defined to be the percentage of start sites correctly identified from the successfully located genes, i.e. we did not penalize a program for being less sensitive at finding genes overall. For the start site accuracy, we divided the start sites into two categories: those where the start site was present in the fragment ('Internal') and those where the correct start site lay beyond the edge of the fragment ('External'). The '% Total' column indicates the percentage of the total start sites that belong to that category. At shorter sequence lengths, the majority of start sites are not present in the contig (external), making it most important not to incorrectly predict a start site near the edge of the sequence. At longer sequence lengths, many more start sites are contained within the fragment (internal), and the RBS motifs and so on become more important.

Prodigal outperformed its nearest competitor, MetaGene Annotator, on internal starts by 5.9% in 3000 bp fragments, 4.9% in 1200 bp fragments and 3.9% in 700 bp fragments. The gap shrinks at the smaller fragment lengths due to less likelihood of upstream information to aid in start prediction, and due to the fact MetaGene calls many more internal starts than the other programs. On start sites external to the contig, Prodigal achieved near perfect results, falsely calling a start site within the fragment only 0.2–0.4% of the time, regardless of fragment length. MetaGene Annotator outperformed MetaGenemark on internal starts, perhaps due to the specific RBS routines added in the annotator version (Noguchi *et al.*, 2008). However, MetaGene Annotator, regardless of fragment length, incorrectly truncated many genes (5–6%) prematurely, calling an internal start site instead of allowing the gene correctly to run off the edge. MetaGeneMark does not experience this problem, although, interestingly, at longer sequence lengths, it begins to truncate more genes prematurely as well (1.7% at 3000 bp).

We also compared MetaProdigal with the start correction program MetaTISA (Hu *et al.*, 2009), which was run as a post-processing step to MetaGene Annotator. Although MetaTISA accurately binned most of the fragments and scored starts with the requisite amount of upstream bases (50 nt) about the same as MetaProdigal, it moved many starts away from the edges of contigs to incorrect starts farther downstream in the contigs. In addition, rather than correcting MetaGene's truncation problem, MetaTISA exacerbated it by taking many more genes that ran off the edges of the contigs and instead predicting false starts for these genes internal to the contigs. A modification to MetaTISA to leave starts near the edge of fragments unchanged, as well as not to truncate genes that run off edges of contigs, would result in a dramatic improvement in its performance.

These results suggest that the simplest change programs could make to improve their start site predictions to implement large penalties for calling a start site near the edge of a fragment when it is possible that the true start site lies beyond the edge. It is worth noting that starts are not present in the contig 90% of the time at 150 bp fragments. Even in 3000 bp fragments, the correct start was not present in the contig in 23.6% of the cases. This highlights the importance of not prematurely truncating genes by calling starts near the edges of contigs, especially in smaller fragment sizes.

3.3 Evaluating confidence measures for gene predictions

Using the confidence measures described in Section 2, we can subdivide our results into confidence intervals and examine how sensitivity and precision change if we only consider high confidence genes, medium confidence genes, etc. Table 4 shows the results of this analysis for 300 bp and 700 bp fragments based on the MetaGeneMark dataset described in Section 3.1. In this table, the sensitivity (Sn), precision (Pr) and *F*-score correspond to the performance of the algorithm if only genes of that confidence level or higher were accepted. For example, Prodigal could achieve a 99.2% precision by accepting only genes with 100% confidence in 700 bp fragments, but it would fail to identify 40% of real genes with this stringent a restriction.

At the longer sequence lengths, Prodigal's confidence score corresponds very well to the actual performance. For example, at 700 bp, 99.2% of genes with a 100% confidence score were true positives and 95% of genes with a confidence score of 90–99.99% were true positives. At the smaller sequence lengths (150 and 300 bp), however, the comparison worsens, and only 38.7% of genes in the 50–59% confidence interval were actually true positives, according to the Refseq annotations of our dataset. This suggests further room for improvement in the scoring function of the algorithm, particularly in our Bonferroni modifications to the scores (Bonferroni, 1935). Perhaps, the algorithm should eliminate more of the lower scoring genes or add more rules to penalize our score based on the fragment or gene length. However, we were reluctant to make changes based on a single dataset, because that could be considered to be fitting to the test set data. Examining these lower scoring genes on a larger dataset to see whether they should be kept is a worthwhile goal for future versions. Regardless of the actual performance, the confidence estimation gives researchers a valuable tool for deciding whether to retain or eliminate a given gene model. We believe this % confidence measure to be a significant improvement over a numerical score, the meaning of which can be often difficult to understand or apply to practical problems.

4 CONCLUSION

We built an open source heuristic *ab initio* algorithm for metagenomic gene prediction using Prodigal. The program can analyze fragments independently and thereby achieve full speedup through utilization of multiple processors. Although we understand the problems posed by sequencing errors, we chose to focus instead on other problems that have received less attention, such as translation initiation site identification, handling of alternate genetic codes and providing filtering mechanisms for scores based on confidence. In future versions, we hope to address sequencing errors in more detail and provide further improvements to the program's performance at smaller fragment lengths.

ACKNOWLEDGEMENTS

We thank Michael D. Galloway for help with the AMD Opteron cluster. We also thank Dr. Jillian Banfield and Brian C. Thomas for helpful discussions and feedback on the metagenomic version of Prodigal.

Funding: Genomic Science Program, US Department of Energy, Office of Science, Biological and Environmental Research, a part of the Plant Microbial Interfaces Scientific Focus Area (http://pmi.ornl.gov/), the BioEnergy Science Center, which is a US Department of Energy Bioenergy Research Center supported by the Office of Biological and Environmental Research in the DOE Office of Science and Oak Ridge National Laboratory is managed by UT Battelle, LLC, for the DOE (DE-AC05-00OR22725).

Conflict of Interest: none declared.

REFERENCES

Aivaliotis,M. *et al.* (2007) Large-scale identification of N-terminal peptides in the halophilic archaea *Halobacterium salinarum* and *Natronomonas pharaonis*. *J. Proteome Res.*, **6**, 2195–2204.

Angelova,M. *et al.* (2010) Computational methods for gene finding in prokaryotes. *ICT Innovations*, 11–20.

Benson,D.A. *et al.* (2011) GenBank. *Nucleic Acids Res.*, **39**, D32–D37.

Bonferroni,C.E. (1935) Il calcolo delle assicurazioni su gruppi di teste. *Studi in Onore del Professore Salvatore Ortu Carboni*, 13–60.

Hoff,K. *et al.* (2008) Gene prediction in metagenomic fragments: a large scale machine learning approach. *BMC Bioinformatics*, **9**, 217.

Hoff,K. (2009) The effect of sequencing errors on metagenomic gene prediction. *BMC Genomics*, **10**, 520.

Hu,G.-Q. *et al.* (2009) MetaTISA: metagenomic translation initiation site annotator for improving gene start prediction. *Bioinformatics*, **25**, 1843–1845.

Hyatt,D. *et al.* (2010) Prodigal: prokaryotic gene recognition and translation initiation site identification. *BMC Bioinformatics*, **11**, 119.

de Jong,A. *et al.* (2010) BAGEL2: mining for bacteriocins in genomic data. *Nucleic Acids Res.*, **38**, W647–W651.

Kawarabayasi,Y. *et al.* (1999) Complete genome sequence of an aerobic hyperthermophilic crenarchaeon, Aeropyrum pernix K1. *DNA Res.*, **6**, 83–101, 145–152.

Massaro,J. (2005) Clustering, Complete Linkage. *Enc. Biostatistics*.

Noguchi,H. *et al.* (2008) MetaGeneAnnotator: detecting species-specific patterns of ribosomal binding site for precise gene prediction in anonymous prokaryotic and phage genomes. *DNA Res.*, **15**, 387–396.

Noguchi,H. *et al.* (2006) MetaGene: prokaryotic gene finding from environmental genome shotgun sequences. *Nucleic Acids Res.*, **34**, 5623–5630.

Pruitt,K.D. *et al.* (2009) NCBI Reference sequences: current status, policy and new initiatives. *Nucleic Acids Res.*, **37**, D32–D36.

Rho,M. *et al.* (2010) FragGeneScan: predicting genes in short and error-prone reads. *Nucleic Acids Res.*, **38**, e191.

Rudd,K.E. (2000) EcoGene: a genome sequence database for *Escherichia coli* K-12. *Nucleic Acids Res.*, **28**, 60–64.

Shine,J. and Dalgarno,L. (1975) Determinant of cistron specificity in bacterial ribosomes. *Nature*, **254**, 34–38.

Yamao,F. *et al.* (1985) UGA is read as tryptophan in *Mycoplasma capricolum*. *Proc. Natl. Acad. Sci. USA*, **82**, 2306–2309.

Yok,N. and Roden,G. (2011) Combining gene prediction methods to improve metagenomic gene annotation. *BMC Bioinformatics*, **12**, 20.

Zhu,W. *et al.* (2010) Ab initio gene identification in metagenomic sequences. *Nucleic Acids Res.*, **38**, e132.