Chapter 4

AUTOMATIC ANNOTATION OF MICROBIAL GENOMES AND METAGENOMIC SEQUENCES

Victor Solovyev and Asaf Salamov

ABSTRACT

We have developed Fgenesb_annotator pipeline that provides completely automatic and comprehensive annotation of microbial genomic sequence. The pipeline identifies protein, tRNA and rRNA genes, finds potential promoters, terminators and operons. Finally potential functions are assigned to predicted proteins using comparison with a set of databases such as COG, KEGG and NR. The package provides options to work with a set of sequences such as scaffolds of bacterial genomes or short reads of DNA extracted from an environmental sample of entire bacterial community. The gene prediction algorithm is based on Markov chain models of coding regions, start of translation and termination sites. Operon models are derived using distances between ORFs, frequencies of neighboring genes in known bacterial genomes and positions of predicted promoters and terminators. The parameters of gene prediction are automatically learned on initial steps of sequence analysis, so the only input necessary for annotation of a new genome is its sequence. The pipeline is actively used in many new genome-sequencing projects and provides a superior accuracy of gene finding (comparing with the other popular bacterial gene finding software) in bacterial contigs and especially in short sequences analyzed in metagenomic projects.

Keywords: bacterial gene prediction, promoters, terminators, operons, automatic genome annotation, metagenomic sequences
**INTRODUCTION**

Recent advances in sequencing technology generated abundance of sequence data often accompanied by complete absence of any experimental characterization of the genome gene content. Therefore the accuracy of computational genome annotation defines the initial quality of scientific and practical knowledge gained from the genome sequences. The main step in the analysis of new microbial genome sequence is the identification of all its genes. The task of highly accurate gene finding is far from being trivial and requires learning specific gene-finding parameters for each class of genomes as well as to have an effective model to select protein coding open reading frames (ORFs) among numerous candidates located in six possible frames for each DNA region. Appearance of metagenomic sequencing projects that sample the genomes of multiple species and strains with shorter average sequence fragment length, higher frequency of sequencing errors, and the phylogenetic heterogeneity of the organisms in the environmental sample, presents significant additional challenges in computational gene finding [1–3]. It is not surprising that some popular gene-finding algorithms demonstrate very poor performance on these currently abundant data [3]. To overcome this problem a few new gene-finders have been developed such as GISMO [4], which is using Support Vector Machine for ORF classification and MetaGene [5], which is designed to predict genes from fragmented genomic sequences.

The bacterial genome annotation pipeline Fgenesb_annotator presented here was initially developed to provide a completely automatic and comprehensive annotation of huge amount of sequences generated by one of the first published metagenomic project [1]. The pipeline includes a suit of original algorithms for protein coding gene identification, operon assignment and promoters and terminators identification. The parameters of gene prediction are automatically trained using uncharacterized genomic sequence. Optionally, parameters from closely related genomes or a set of generic parameters for annotating short reads extracted from multiple species sample could be used. The pipeline also includes a module for rRNA genes identification and incorporates well-established public programs such as tRNAscan-SE [6] and Blast [7]. Fgenesb_annotator is widely used to annotate sequences such as scaffolds of bacterial genomes or short reads of DNA extracted from a bacterial community [8–11]. Recently the pipeline superior performance has been demonstrated on prediction of coding sequences for the assembled contigs and the unassembled reads. The Fgenesb_annotator correctly identified 10-30% more reference genes on the contigs than the combination of Critica [12] and Glimmer [13] pipeline in every data set. On unassembled reads Fgenesb_annotator correctly identified ~70% and missed ~20% of reference genes, while the Critica-Glimmer pipeline correctly predicted just 7% of genes and missed 85% [3]. The remaining reference genes were inaccurately called. In this paper we for the first time describe the Fgenesb_annotator main components, its accuracy, presentation and visualization of sequence annotation results. In general, we tried to keep design of each pipeline component simple, rather than to construct it with full sophistication achieved in some known genome annotation algorithms, as we understand that simpler models might behave more consistently and provide better performance on very heterogeneous and short sequences from metagenomics projects often lacking some features of complete genes and covering just small parts of the encoded proteins.
MATERIAL AND METHODS

Learning Parameters and Prediction of Protein-Coding Genes

Prediction of protein-coding genes depends on a set of parameters describing a bacterial gene model. Many of them such as, for example, codon usage are specific for each new genome. The Fgenesb algorithm includes software module which iteratively learns these parameters using a given genomic sequence. It starts with a compilation of all relatively long ORFs (> 200 bp) that serve as an initial gene set for calculating organism-specific gene finding parameters. A few algorithms that use long ORFs (or ORFs with a significant match to a protein from a different organism) for deriving models of coding sequences have been described earlier [14-16]. In the subsequent iterations we assign each potential ORF a score which reflects a joint probability to observe various features associated with protein-coding sequences such as coding content of a reading frame, oligonucleotide composition of sequence regions surrounding start and stop codons and the closeness of ORF’s length to the length distribution of ‘real’ genes. Protein coding sequences are modeled by 3-periodic fifth-order Markov-chains using 5th-order Markov transition probabilities computed for hexamers ending at each of the three codon positions. Non-coding sequences are also modeled using 5th-order Markov transition matrix computed for hexamers ending at any non-coding nucleotide. Markov models or Hidden Markov models using Markov transition probabilities of second or fifth-order have earlier been implemented in many bacterial gene-finding algorithms [15–17]. Start of coding region including ribosome-binding site (RBS) is modeled by second order Markov model using second-order Markov transition probabilities computed for triplets ending at nucleotides in the upstream of the start codon sequence [-20 - +1] (here +1 is the position of the 1st nucleotide of start codon). To derive the log-likelihood ratios (start/nonstart probabilities) we also computed second-order Markov non-start transition matrix using sequences upstream of non-start triplets ATG, GTG, TTG and TGT. These probabilities have been averaged over windows of 5 bp long to account for variable location of RBS in 5′-gene region. The RBS weight matrix was applied in one of earlier works on bacterial gene finding to improve gene identification [19]. To model the stop codon we used second-order Markov transition matrix computed for triplets ending at the positions +1 to +3 after the stop codon. To derive the log-likelihood ratios (stop/nonstop probabilities) we also computed second-order Markov non-start transition matrix computed for triplets ending at the positions +1 to +3 of TAA, TAG and TGA non-stop codon triplets. We also exploit a distribution of the ORF lengths that is accounted in combined scoring of potential protein-coding sequences.

If an ORF’s start and end positions are L1 and L2 (excluding the stop codon), then the coding score of the ORF (S) is defined as

$$S = \sum_{i=L1}^{L2-5} llh(i) + \sum_{i=L1-20}^{L1} llw(i) + \log(P(L2 - L1 + 1)) + \sum_{i=L2+1}^{L2+3} llwm(i),$$

where \(llh(i)\) is the log-likelihood ratio for the probability of generating nucleotide \(Xi\) in sequence position \(i\) by Markov model of coding region to the probability of generating nucleotide \(Xi\) by Markov model of non-coding region computed using corresponding Markov
transition matrices of the fifth-order described above; \(ll\ww(i)\) is the log-likelihood ratio for the probability of generating nucleotide \(X_i\) in sequence position \(i\) using Markov model of start region to the probability of generating nucleotide \(X_i\) by Markov model of non-start region computed using corresponding Markov transition matrices of the second-order described above; \(P(L_2-L_1+1)\) is the probability of a coding region to have \(L_2-L_1+1\) length; and \(ll\wm(i)\) is the log-likelihood ratio for the probability of generating nucleotide \(X_i\) in sequence position \(i\) using Markov model of the stop codon region to the probability of generating nucleotide \(X_i\) by Markov model of non-stop region computed using corresponding Markov transition matrices of the second-order described above.

At each iteration step the algorithm uses gene finding parameters produced on the previous stage and selects ORFs with the scores higher than some predefined threshold, sorts them in the descending order by score and accounts any selected ORF as a protein-coding gene if it overlaps with no more than 2 previous higher scoring genes and the fraction of the overlap sequence does not exceed 50% of the ORF length. Iteration stops when at least 99% of predicted genes between two successive iterations are the same. Optionally, instead of learning gene-finding parameters from a given sequence, the parameters from closely related genomes could be used. Also, we provide generic parameters to annotate unknown bacteria, archaea or their mixture sequences. These parameters have been used to annotate short sequences extracted from the environmental samples [1].

Prediction of Potential Promoter Regions

To identify significant features of bacterial promoter regions we searched for conserved sequences in a set of known promoters from E.coli genome, which has the largest number of experimentally verified promoters. This set was used earlier in developing promoter prediction algorithm and described in Gordon et al. [20]. Five relatively conserved sequence motifs (represented by their weight matrices) have been selected to describe a bacterial promoter model. Two most conserved motifs correspond to the well characterized –10 and –35 sequence elements of promoters regulated by sigma70 factors. The third motif (upstream of –35 box) with length 7 bp is searched in the area (–60 to –40); the fourth motif (downstream of –10 block) with length 7 bp is searched in the area (–11 to +10) and the fifth motif (between –35 and –10 boxes) has length 5 bp and is searched in the area (–31 to –22) of potential promoter sequences. We applied a linear discriminant analysis to derive the recognition function for discrimination between promoter and non-promoter sequences using as the ‘negative’ set a collection of sequences from the inner regions of protein-coding ORFs. In addition to the above-described five motifs, we also considered the following features: the distance between –10 and –35 elements and the ratio of densities of octa-nucleotides overrepresented in known bacterial transcription factor binding sites relative to their occurrence in the coding regions. We used bacterial functional sites collected in the DPInteract database [21]. The last feature was calculated similar to the one used in the eukaryotic promoter recognition programs such as PromoterScan [22] and TSSW [23]. The constructed linear discriminant function (LDF) demonstrated sensitivity 83% and specificity 84% in recognition of promoter and non-promoter sequences not included in the learning set.

To reduce the rate of false positive predictions we restrict prediction of promoters to the upstream regions of predicted ORFs in the annotation pipeline. Besides of its own value, the
predicted promoters and terminators (described in the next section) help to identify more accurately the exact boundaries of operons.

**Prediction of Potential Terminators**

Currently we implemented only prediction of rho-independent terminators as they have well-known distinguished characteristics. Based on statistical analysis of E.coli rho-independent terminators, d'Aubenton-Carafa et al. [24] suggested a model for such terminators consisting of a hairpin followed by a T-rich region. More recently Ermolaeva et al. [25] and Lesnik et al. [26] refined the model testing it on the complete genome sequence of E.coli. To build terminator recognition function we used the terminator features described in Lesnik et al. [26] work. The terminator model includes the following main components: a) a variable-length hairpin; b) a variable-length spacer and c) a 13-nt T-rich region. The hairpin is a basic part of terminator and has the following structural constraints: a) its loop consists of 3-10 bases and b) there are 3 types of stems: 1) perfect stem of 4-18 nt; 2) stem of 9-18 nt with internal loop no longer than 20% of the stem length and 3) stem of 7-18 nt with 1-5 nt bulges in either 3’ or 5’ side of the stem. Also the hairpin should have at least 4 GC/CG or GT/TG pairs.

These features were used as the components of linear function that we trained to discriminate terminators and potential hairpins within ORFs of E.coli genome. The LDF included the following seven features of bacterial terminators: 1) – 3) number of CG/GC, AT/TA and GT/TG pairs in the stem, respectively; 4) number of bases in outer loop; 5) the length of possible bulge; 6) the length of possible internal loop; 7) the length of spacer between the hairpin and T-tail region. We tested the performance of our discriminant function on a set of genomic sequences with known factor independent terminators in B. subtilis genome. We can detect 91% of these terminators with specificity of 92% that is comparable to the best-known methods tested on similar data [27].

In the annotation pipeline we restrict prediction of terminators to the downstream regions of predicted genes, provided that they followed a sufficiently long intergenic sequence (at least 100 bp).

**Prediction of Operons**

Prediction of operons is based on several characteristics of bacterial operons such as the short distances between the adjacent genes, frequent conservation of adjacent gene pairs in other genomes and the location of predicted promoters and terminators. Analyzing a set of experimentally characterized operons in E. coli, Salgado et al. [28] have found that genes in the same operon have much shorter intergenic distances than genes at boundaries of transcription units. Therefore they derived log likelihoods of gene pairs to be in the same operon as a function of distance. In another study Ermolaeva et al. [29] applied the frequency of appearance of the neighbor genes in multiple genomes to assign genes to the same operon. Some approaches used expression data [30] or functional class of genes [31] to improve operon prediction. While it is definitely the relevant features, we try to exploit operon
In our approach to identify operons we developed a complex scoring system for potential operons and single gene transcription units (TUs) including multiple lines of evidences. We learned a lot from some successful operon prediction approaches that combine similar characteristics [31-32]. At the same time we developed our own original implementation of many important components of the method. First, we compute a log-likelihood score for each pair of neighboring genes to be in operon based on the known intergenic distance distribution in operons [33], essentially the same way as in the method of Salgado et al. [28]. As a ‘negative’ data set we used distances between adjacent genes in the same strand at the transcription unit boundaries. Taking into account the relative scarcity of experimental data, we computed ‘distance’ likelihoods only in intervals of 50 bp for distances between 10 bp and 500 bp (10-50, 50-100,…,450-500) and separately for distances < -10, distance > 500 bp and distance between -10 and 10 bp.

As the scores accounting for predicted promoters and terminators we used their corresponding LDF values, which are also log-likelihoods in nature, although in a different scale.

The total score of an operon comprised of N genes is calculated as \( \text{score}_o = \text{score}_d + k \times (\text{score}_p + \text{score}_t) \). Here the \( \text{score}_d \) is the sum of the log-likelihoods for distances of all pairs of neighboring genes in the operon. The \( \text{score}_p \) is the score of predicted promoter upstream the first gene in the operon, and the \( \text{score}_t \) is the score of predicted terminator downstream of the last gene in the operon. If the promoter or terminator is not predicted, their scores are set to 0. The score of a single gene (single transcriptional unit) is based on combined scores of adjacent predicted promoter and terminator: \( \text{score}_{TU} = b \times (\text{score}_p + \text{score}_t) \). The coefficients \( k \) and \( b \) were chosen by optimizing the prediction performance on the E.coli training set. We also check localization of all pairs of predicted neighboring genes in the representative set of microbial genomes comprising the COG database [34] and calculated P-values for the frequencies of their adjacent occurrence in the considered genomes by chance. The conservation of a gene pair is used to improve the operon identification in the simple procedure: if P-value is less than 0.001, then the correspondent distance log-likelihood for a given pair will be calculated as if the distance between genes is 0, independent from the actual distance. Applying the scoring scheme described above, the algorithm for predicting operons and single transcription units takes the following steps:

a) the predicted gene set is subdivided on ‘directons’ (uninterrupted strings of genes in the same strand);
b) for each ‘directon’ we compute scores for all possible assignments of operons and single TUs to its N genes;
c) by dynamic programming procedure we find a parse (a variant of gene assignment to operons and single transcriptional units) with the maximal total score for the considered ‘directon’. This assignment will represent the operon prediction results.
Prediction of RNA Genes

Using BLASTN alignment of our collection of bacterial and/or archaean rRNAs to genomic sequences we identify location of potential ribosomal RNA genes. The database of rRNAs was produced from various public repositories and includes small (16S or SSU), large (23S or LSU) and 5S RNA known genes. tRNA genes are predicted using tRNAscan-SE program [6]. Genomic sequence of predicted tRNA and ribosomal RNA genes is subsequently masked to avoid false prediction of protein-coding genes in these regions.

Annotating Predicted Proteins

Predicted protein sequences are annotated by comparison with COG and other user selected protein databases such as KEGG [35], eggNOG (automatic extension of COG) [36] or NR [37]. In the output file we report the best significant Blastp hits (with E-value < e-10) in each selected database. Finding significant COGs hits gives some clue about possible functional class of the predicted gene. KEGG hits might provide information on gene participation in a specific metabolic pathway. NCBI’s NR database is one of the most comprehensive data sets, while many of its proteins sequences lack any functional description. The NR can be checked if no significant similarities are observed in the three previously mentioned databases.

Table 1. Performance of three bacterial gene finders on “difficult” genes

<table>
<thead>
<tr>
<th></th>
<th>Sn exact (%)</th>
<th>Sn exact+overlapping (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>123set:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Glimmer</td>
<td>57.0</td>
<td>91.1</td>
</tr>
<tr>
<td>GeneMarkS</td>
<td>82.9</td>
<td>91.9</td>
</tr>
<tr>
<td>FgenesB</td>
<td>89.3</td>
<td>98.4</td>
</tr>
<tr>
<td><strong>72set:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Glimmer</td>
<td>57.0</td>
<td>91.7</td>
</tr>
<tr>
<td>GeneMarkS</td>
<td>88.9</td>
<td>94.4</td>
</tr>
<tr>
<td>FgenesB</td>
<td>91.5</td>
<td>98.6</td>
</tr>
<tr>
<td><strong>51set:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Glimmer</td>
<td>51.0</td>
<td>88.2</td>
</tr>
<tr>
<td>GeneMarkS</td>
<td>90.2</td>
<td>94.1</td>
</tr>
<tr>
<td>FgenesB</td>
<td>92.0</td>
<td>98.0</td>
</tr>
<tr>
<td><strong>B.subtilis genome:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Glimmer</td>
<td>62.4</td>
<td>98.1</td>
</tr>
<tr>
<td>GeneMarkS</td>
<td>83.2</td>
<td>96.7</td>
</tr>
<tr>
<td>FgenesB</td>
<td>87.9</td>
<td>98.7</td>
</tr>
</tbody>
</table>
RESULTS

Accuracy of Protein Coding Gene Prediction by Fgenesb

Fgenesb gene prediction engine is one of the most accurate prokaryotic gene finders. Its performance has been estimated on various datasets. We have computed the accuracy estimate on a set of difficult for prediction short genes that was used previously for evaluating several other bacterial gene finders. First set (51set) has 51 genes with at least 10 strong similarities to known proteins. The 72set has 72 genes with at least two strong similarities, and 123set has 123 genes with at least one known protein homolog. The prediction accuracy on these three sets and on the B.subtilis genome (annotated in GenBank) for GeneMarkS, Glimmer (calculated in [38]) and for Fgenesb gene prediction algorithm is presented in Table 1.

The Fgenesb predictor used E.coli gene-finding parameters for three sets mentioned above and automatically learned the parameters from B.subtilis genomic sequence for its genome annotation. We can observe that Fgenesb demonstrates superior accuracy comparing with the other two popular gene finders. Recently the pipeline performance has been tested on prediction of coding sequences for the representative set of assembled contigs (which are the typical results of bacterial sequencing) and as well as for the unassembled reads. The other pipeline in that study was a combination of Critica and Glimmer (called here CG pipeline). The Fgenesb correctly identified 10-30% more reference genes on the contig sequences than the CG pipeline in every analyzed data set. The accuracy of the gene calls on unassembled reads (where many species are usually represented) was also evaluated. Fgenesb correctly identified ~70% and missed ~20% of reference genes on unassembled reads in all data sets. The CG pipeline exhibited significantly more poor results (~7% accurately predicted and 85% missed genes) [3]. Both pipelines predicted 7-10% of genes inaccurately.

To make another comprehensive test we downloaded sequences and annotations of complete genomes from the IMG database [39]. 216 genomes have been selected that have percent of “unusual” annotated start codons (i.e. not atg, gtg, tgt, ctg, att, atc, or ata) and stop codons (not tag, taa, tga) less than 0.5%. The accuracy data for Fgenesb, Metagene [5] and latest versions of GeneMarkS [38] and Glimmer [40] are presented in Table 2.

We can observe that Fgenesb, Metagene and GenMarkS ab initio gene predictors have comparable performance on the annotated complete genomes finding ~96% of annotated genes. At the same time we observe that just ~76% or less of start codons predictions coincide with the annotations provided by research groups. Most of these annotations have no experimental verification, therefore we can not exclude that some start codon assignments are not correct and this problem is waiting of some solution in the future. The Glimmer (standalone and web server version) [40] shows lower specificity than Fgenesb, Metagene and GenMark that is in agreement with the other studies [4, 41].

One more test of gene predictors we made for artificial shotgun sequences (700 bp fragments from a set of 216 bacterial genomes). We extracted these sequences with real genes covering only part of each sequence (by its 5’- or 3’-fragment) to make gene prediction more difficult. Correct gene assignment for these sequences was extremely high for all three gene finders (Table 3), while Metagene shows slightly lower specificity than Fgenesb and GeneMark.
Table 2. Gene prediction accuracies for 216 bacterial complete genomes

<table>
<thead>
<tr>
<th></th>
<th>Sn (%) exact+overlapping</th>
<th>Sn (%) exact</th>
<th>Sp (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fgenesb</td>
<td>96.0</td>
<td>76.8</td>
<td>91.2</td>
</tr>
<tr>
<td>GeneMarkS</td>
<td>96.5</td>
<td>73.5</td>
<td>90.7</td>
</tr>
<tr>
<td>Metagene*</td>
<td>95.9</td>
<td>76.8</td>
<td>93.7</td>
</tr>
<tr>
<td>Glimmer</td>
<td>95.2</td>
<td>73.3</td>
<td>88.5</td>
</tr>
</tbody>
</table>

*) Metagene uses a set of regression functions for estimation the coding scores, which have been computed using annotations of 170 bacterial genomes (many of them are among our test set sequences).

Table 3. Gene prediction accuracies for short bacterial sequences

<table>
<thead>
<tr>
<th></th>
<th>(Sn+Sp)/2 (%)</th>
<th>Sn (%)</th>
<th>Sp (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fgenesb</td>
<td>95.55</td>
<td>98.5</td>
<td>93.5</td>
</tr>
<tr>
<td>GeneMark</td>
<td>94.05</td>
<td>98.8</td>
<td>91.3</td>
</tr>
<tr>
<td>Metagene</td>
<td>91.65</td>
<td>97.6</td>
<td>82.9</td>
</tr>
</tbody>
</table>

Our results presented in Table 2 and 3 are in close agreement with the performance analysis of Metagene and GenMark tested on different data sets earlier [5].

All software components of Fgenesb work extremely fast (processing a bacterial genome sequence requires only a few minutes). The time consuming stage is annotation by Blastp, which for *E.coli* genome takes around 12 hours using a single processor. Running the pipeline on multiple processors with the corresponding Blastp options would speed up the annotation proportionally.

Table 4. The *E.coli* operon prediction accuracy for different components of the method

<table>
<thead>
<tr>
<th>Prediction component</th>
<th>Genes correctly assigned to be in operons</th>
<th>Genes correctly assigned to belong to single TUs</th>
<th>Exact operons</th>
<th>Partial operons</th>
</tr>
</thead>
<tbody>
<tr>
<td>By distance log likelihoods</td>
<td>1005 (90.1%)</td>
<td>154 (77.0%)</td>
<td>125 (47.8%)</td>
<td>247 (92.5%)</td>
</tr>
<tr>
<td>Adding simple rule for pair conservativeness</td>
<td>1019 (91.4%)</td>
<td>151 (75.5%)</td>
<td>124 (46.8%)</td>
<td>253 (94.8%)</td>
</tr>
<tr>
<td>Adding promoter and terminator scores</td>
<td>1023 (91.7%)</td>
<td>151 (75.8%)</td>
<td>141 (52.8%)</td>
<td>257 (96.3%)</td>
</tr>
</tbody>
</table>
Testing Operon Prediction Performance

The accuracy of operon predictions was tested on the experimentally validated set of 200 single TUs and 267 operons comprising 1121 genes of E.coli genome [42]. The results of the testing are presented in Table 4.

“Exact operons” is the number of exactly predicted operons (exact assignment of all operon genes). “Partial operons” is the number of operons with at least one pair of genes assigned correctly.

It can be observed (Table 4) that although adding conservation information increased the accuracy of assigning genes to operons, it slightly decreased the accuracy of predicting the correct operon boundaries. As we used the length distribution of intergenic distances within operons derived from the annotation of E.coli genome, the test set was not independent on some part of learning data. To evaluate the Fgenesb operon prediction accuracy using unseen data, we tested our approach on two sets of data from different bacteria. One test we performed with experimentally verified 251 operons of Bacillus subtilis having sizes from 2 to 31 genes. Fgenesb_annotator annotating the bacteria sequence predicted exactly 64% of them and 92% of these operons were predicted partially. For another test 22 experimentally determined operons of Staphylococcus aureus Mu50 (NC_002758) have been investigated [43]. The bacterial sequence for Fgenesb annotation was downloaded from the NCBI (NC_002758.1). The following results have been achieved: 15 out of 22 operons (78%) were predicted exactly. In average 88.7% of genes in an operon were predicted correctly. These figures are close to the results reported by Wang et al. [44] whose consensus approach predicted 75% out of the considered 36 operons exactly.

The Fgensb_annotator Output

For each genomic sequence (complete genome, scaffold, read, etc.), program lists locations of predicted protein coding, rRNAs, tRNAs genes, operons, promoters and terminators. A fragment of annotation is presented in Figure 1. Predicted protein coding regions are labeled as CDS. They are numbered by their order in the sequence and marked whether they are transcribed as a single transcription unit (Tu) or as an operon (Op). If the predicted proteins have homologs in the COG or other user selected protein database we provide the short name of the best homolog after ## separator. For example:

4 Op 2 + CDS 2737 - 3744 871 ## COG0673 Predicted dehydrogenases ...

This line is the description for predicted gene number 5 of the 4th operon with coordinates 2737 - 3744 in the ‘+’ strand and it is the second gene in the operon; 871 is the BLAST score and COG0673 is the ID of the best homolog in the COG database. At the end of annotation file we provide protein sequences of predicted genes in fasta format with the full name of the best homolog and some similarity characteristics extracted from the Blastp output. For example, for the gene considered above we have the following protein description:
Contig 1 GENE 5 2737 - 3744 871 335 aa, chain + # L109882 COG0673 #
Gene_name: MviM # Func_class: R General function prediction only # Function: Predicted dehydrogenases and related proteins # Organism: Lactococcus lactis
## # L109882 COG0673 #
MIKLGLVGTHWITAQFAQAVLETGKYEIAIYSRNKENAKKFSVKIQQNKA$LYDNF$
DDFID$DVIQVYLYASPNSFHFEHAQIAKHDVDVI$VEKPSFSNPQFRTIDLLKLHPKIR$
LFEAARNYHPFNKVQSTVKNLDDLQGANLIVYAHYSSRYDEYLAKPDNPPNVPFT$
DFS$GGALYDLGVYLDALGWF$YPEKIDYKAQLKNGIDAFGW$KLKYSKFSVGI$F$
SKVFT$TAPTEIFGLKHTIEIDSPSELNQIA$vingHKQFV$ADGSRNP$NF$AD$ARV$
LND$R$EKNQ$SE$YE$KWLKIA$TQINQQLFDLRKSAGI$DPAD$Q$

Here is the Fasta formatted sequence, where the name is actually presented in one line that contains a short annotation from the COG database. The end of name line contains characteristics of similarity of the predicted protein and its best homolog: the start and the end of similarity region (1, 335) in the predicted protein; the start and the end of similarity region (1, 326) in the homolog; length of homologous protein (326); the Blastp score (198), identity (37.0%) and expected value ($6e^{-51}$).

For the other predictions (rRNA and tRNA genes, promoters, terminators) we provide one line description, for example: LSU_RRNA 884415 - 887254 98.0 # Leuconostoc oenos S60377, with positions of the feature and the score value that was used in the feature identification.

**CONCLUSION**

We have developed an automatic annotation pipeline that includes the following main components:

- Automatic learning gene finding parameters for new bacterial genomes or using pre-learned parameters from the related organisms;
- Generic Bacterial, Archaeabacterial or mixture parameters can be used for mixture of short sequences from an environmental sample;
- Identification of tRNA and rRNA genes;
- Highly accurate Markov chains-based gene prediction;
- Prediction of promoters and terminators;
- Operon prediction based on distances between predicted genes and frequencies of different genes neighboring each other in known bacterial genomes, as well as accounting promoter and terminator predictions;
- Annotation of predicted genes by homology with COG, KEGG, NR and other databases.
Figure 1. Example of Fgenesb output. The fragment of B.subtilis automatic annotation.
Figure 2. Presentation of Fgenesb annotation in Bacterial Genome Explorer of a complete bacterial genome (A) and a set of metagenomic sequences (B).
Victor Solovyev and Asaf Salamov

Figure 3. Example of visualization of FgenesB annotation by CGView.

The Fgenesb_annotator package also includes Absplit program that separates archaeabacterial and eubacterial sequences to annotate them with the bacterial class specific gene-finding parameters. The final annotation can be presented in the GenBank format to be readable by visualization software such as Artemis [45] or Softberry Bacterial Genome Explorer in server and local versions (Figure 2) or converted into the format that is required by SEQIN software for submission of new sequences with their annotations to the GenBank. The package also includes converters to visualize the gene predictions (Figure 3) by CGView software [46].

The Fgenesb and the other programs of bacterial sequence analysis are available to run at the Bioinformatics web server of Royal Holloway (University of London) http://mendel.cs.rhul.ac.uk/mendel.php?topic=fgen and Softberry Inc. web server: http://www.softberry.com/berry.phtml?topic=index&group=programs&subgroup=gfindbor in combination with a hundred other bioinformatics software modules within the MolQuest package developed to run on Windows, MAC and Linux computers (http://www.molquest.com). The Fgenesb_annotator has been applied in dozens of published
studies for annotation of whole bacterial genomes, as well as short sequences extracted from environmental samples (for examples, 47–49). The operons identified by Fgenesb_annotator have been used as initial operon models that were further experimentally investigated and improved [50-51]. Other pipeline components such as Bprom (promoter predictor) and Bterm (terminator predictor) also have been actively used in many studies of functional characterization of bacterial sequences [52–58].

REFERENCES


Automatic Annotation of Microbial Genomes and Metagenomic Sequences


