© 1999 Oxford University Press

Improved microbial gene identification with GLIMMER

Arthur L. Delcher[1,2,*], Douglas Harmon[1], Simon Kasif[3], Owen White[4] and Steven L. Salzberg[4]

[1]Department of Computer Science, Loyola College in Maryland, Baltimore, MD 21210, USA, [2]Celera Genomics, 45 West Gude Drive, Rockville, MD 20850, USA, [3]Department of Electrical Engineering and Computer Science, The University of Illinois at Chicago, Chicago, IL 60607-7053, USA and [4]The Institute for Genomic Research, 9712 Medical Center Drive, Rockville, MD 20850, USA

Received July 7, 1999; Revised and Accepted October 17, 1999

ABSTRACT

The GLIMMER system for microbial gene identification finds ~97–98% of all genes in a genome when compared with published annotation. This paper reports on two new results: (i) significant technical improvements to GLIMMER that improve its accuracy still further, and (ii) a comprehensive evaluation that demonstrates that the accuracy of the system is likely to be higher than previously recognized. A significant proportion of the genes missed by the system appear to be hypothetical proteins whose existence is only supported by the predictions of other programs. When the analysis is restricted to genes that have significant homology to genes in other organisms, GLIMMER misses <1% of known genes.

INTRODUCTION

Accurate microbial gene identification is becoming ever more important with the increasing rate of whole genome sequencing projects. In the past year alone, eight new bacterial and archaeal genomes have appeared, and the pace continues to accelerate. Each new genome contains thousands of new genes, all of which are deposited into public databases. These genes then become the basis for much further research into the biology of these organisms, and their sequences are used for further biological study. For work such as microarray analysis, in which specific sequences are arrayed onto a substrate and used as probes to measure expression levels, the accuracy of gene predictions is critical. The same point can be made about knockout experiments, which are an important tool to use in determining the function of the large numbers of genes whose function is unknown at the time of publication. Such hypothetical proteins typically comprise 30–40% of the genes in a newly sequenced genome.

GLIMMER 1.0 is a computational gene finder that finds 97–98% of all genes in a prokaryotic genome without any human intervention (1). The system can be quickly and easily trained using only the genome sequence of interest. The technical underpinning of the system is an interpolated Markov model (IMM), a generalization of Markov chain methods. GLIMMER 1.0 has been used as the gene finder for *Borrelia burgdorferi* (2), *Treponema pallidum* (3), *Chlamydia trachomatis* (4) and *Thermotoga maritima* (5), and the software is in use at over 100 laboratories and institutes. Below we describe the algorithm and performance results of GLIMMER 2.0, a gene finder that incorporates several technical improvements to the GLIMMER 1.0 algorithm. As a result of these improvements, GLIMMER 2.0 has slightly higher sensitivity than GLIMMER 1.0 and is much better at resolving overlapping gene calls. The latter property is especially useful for genomes such as *Deinococcus radiodurans*, which due to their high GC-content have numerous long open reading frames (ORFs) that can easily lead to predictions of genes whose boundaries overlap incorrectly.

METHODS AND ALGORITHMS

We begin by briefly reviewing Markov models in the context of DNA sequence analysis. We then describe the probabilistic model used in GLIMMER 2.0 to identify regions that are likely to be genes. We then describe how GLIMMER 2.0 resolves conflicts when overlapping genes are predicted. The complete GLIMMER 2.0 system is available from The Institute for Genomic Research at http://www.tigr.org/softlab

Markov Models

A Markov chain is a sequence of random variables X_i, where the probability distribution for each X_i depends only on the preceding k variables $X_{i-1}, ..., X_{i-k}$, for some constant k. For DNA sequence analysis, a Markov chain models the probability of a given base b as depending only on the k bases immediately prior to b in the sequence. We refer to these preceding k bases as the context of base b in the sequence. The most common type of Markov chain is a fixed-order chain, in which the entire k-base context is used at every position. For example, a fixed 5th-order Markov chain model of DNA sequences comprises $4^5 = 1024$ probability distributions, one for each possible 5mer context. Such fixed 5th-order models have proven effective at gene prediction in bacterial genomes (6,7).

Ideally, larger values for k are always preferable. Unfortunately, because the training data available for building models is limited, we must limit k. In most collections of DNA coding

[*]To whom correspondence should be addressed at: Department of Computer Science, Loyola College in Maryland, Baltimore, MD 21210, USA. Tel: +1 410 617 2740; Fax: +1 410 617 2157; Email: delcher@cs.loyola.edu

sequences, however, there is substantial variability in the frequency of occurrence of different *k*mers.

IMMs are a generalization of fixed-order Markov chains that combine contexts of different lengths to compute the probability of base *b*. Our formulation allows each context to have a weight based in part on its frequency; this allows the IMM to be sensitive to how common a particular oligomer is in a given genome. In particular, rare *k*mers should not be used for prediction; the IMM will ignore these in favor of shorter Markov chains. On the other hand, some long *k*mers may occur very frequently, and for those the IMM can give the longer context more weight and make a better prediction. These weights define an interpolated probability distribution that incorporates information from multiple Markov chains. An IMM can emulate a fixed *k*th-order chain simply by setting all weights to zero except for those associated with *k*.

Details of how to construct an IMM for sequence data have been described previously (1). For coding regions, GLIMMER 1.0 builds three separate IMMs, one for each codon position. [This is known as a 3-periodic Markov model (6).] These IMMs include 0–8th order Markov chains, as well as weights computed for every oligomer of eight bases or less that appears in the training data. These weights and Markov models are interpolated to produce a score for each base in any potential coding sequence. The logs of these scores are summed to score each coding region.

The interpolated context model

Interpolated context models (ICMs) are a further extension of IMMs. For a given context $C = b_1 b_2 \ldots b_k$ of length *k*, the IMM in GLIMMER 1.0 computes a probability distribution for b_{k+1} using as many of the bases immediately preceding b_{k+1} as the training data set allows. The ICM is more flexible and can select any of the bases in *C* (not just those adjacent to b_{k+1}) to determine the probability of b_{k+1}. In general, from a given context, the ICM will choose approximately the same number of bases as the IMM. Our motivation for choosing bases other than those at the end of the context is the fact that in coding regions the significance of a given base depends strongly on its position in a codon; e.g. the nucleotide in the third codon position is sometimes irrelevant to the amino acid translation.

The criterion employed by the ICM to select which bases of a context C to use is mutual information. The mutual information between a given pair of discrete random variables *X* and *Y* is defined to be:

$$I(X;Y) = \sum_i \sum_j \mathrm{P}(x_i, y_j) \log \left(\frac{\mathrm{P}(x_i)\mathrm{P}(y_j)}{\mathrm{P}(x_i, y_j)} \right)$$

where x_i and y_j are the values taken by random variables *X* and *Y* respectively, and $\mathrm{P}(x_i, y_j)$ is the joint probability of x_i and y_j together.

To construct an ICM with context length *k* from a training set *T* of DNA sequences, we begin by considering all windows (i.e. oligomers) of length *k*+1 that occur in *T*. We let random variable X_1 be the distribution of bases in the first position of those windows; X_2 be the distribution of bases in the second position; and so on through X_{k+1}. We then calculate the mutual information values $I(X_1; X_{k+1})$, $I(X_2; X_{k+1})$, …, $I(X_k; X_{k+1})$, and choose the maximum. Suppose that maximum is $I(X_j; X_{k+1})$. We then partition our set of windows into four subsets based on the nucleotide that occurs in position *j* in the window.

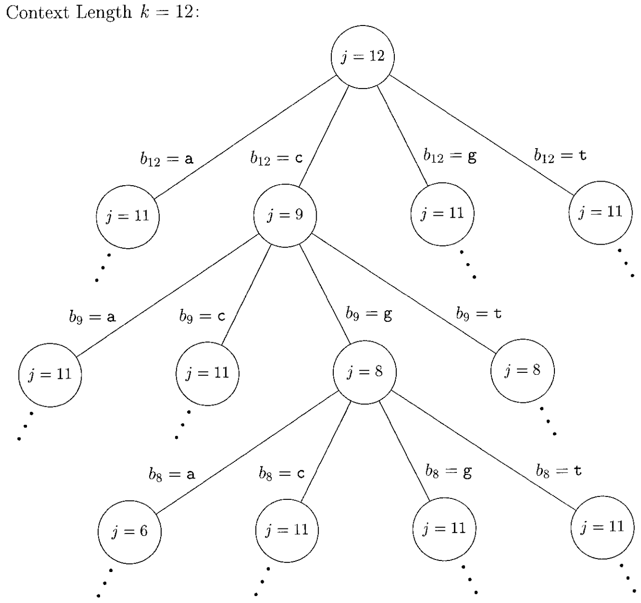


Figure 1. Sample ICM decomposition tree. The root position 12 has maximum mutual information with the final base position 13. Each child of the root represents the subset of windows with the indicated nucleotide value at position 12, and indicates the maximum mutual information position for that subset. Each node is similarly decomposed into children. Note that children of a single node may represent different base positions.

The same procedure can now be performed again for each of the four sets of windows. Within each set, the position that has the highest mutual information with the base at position *k*+1 is chosen. The four nucleotide values at that position induce a further partitioning of the current set of windows into four subsets.

This process can be viewed as constructing a tree of positions within context strings. A sample portion of such a tree is shown in Figure 1. The construction is terminated when the tree depth reaches a predetermined limit, or when the size of a set of windows becomes too small to be useful to estimate the probability of the last base position.

Each node in the ICM decomposition tree represents a set of windows that provide a probability distribution for the final base position. The root node, which includes all possible windows, represents a 0th-order Markov model. All other nodes give a probability distribution for the final base position, conditional on a specific set of bases occurring at the positions indicated on the path to the root from that node.

Note that the IMM used in GLIMMER 1.0 is a special case of this ICM, namely the case where the base chosen at each level of the tree is the last available base in the context window. Thus, when the nearest positions to base b_{k+1} provide the strongest evidence for its value, the ICM automatically chooses them and the result is identical to the IMM. But when other bases provide stronger evidence, as is often the case, the ICM will choose them instead.

The interpolation mechanism used in the ICM is identical to that used in GLIMMER 1.0. It takes a weighted sum of two probability distributions, where the weights are determined by the number of training instances used to construct the distribution and its statistical significance as measured by a χ^2 test. The only difference is that the ICM interpolation is naturally viewed as interpolating between the distributions at a parent and child node in the tree, while the IMM interpolation is always between distributions obtained using different numbers of bases at the end of the context window.

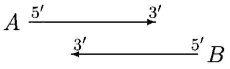
The interpolated context model presented here is essentially a probabilistic decision tree, i.e. a sparse probability distribution expressed as a decision tree. The tree construction is identical to constructing classification trees using information gain as the splitting criteria (8). Classification trees associate a class label with each leaf node of the tree. The labels in our case are the four nucleotide values, and our interpolated context model determines a probability distribution for the base to be predicted given the context in which it occurs. Probabilistic decision trees have been designed for other applications (9–11). In computational biology probabilistic decision trees have been used for modeling splice site junctions (12) and exon modeling (13).

Resolving overlapping genes

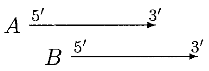
In developing GLIMMER 2.0, a conscious effort was made to reduce the number of false negative gene predictions at the expense of a slight increase in the number of false positive predictions. Upon close examination of GLIMMER 1.0s output, we learned that occasionally a gene was discarded because its start codon was positioned too far in the 5′ direction, resulting in substantial overlap with another gene. GLIMMER 2.0 solves this problem by incorporating additional rules to resolve such overlaps.

In GLIMMER 1.0, when two potential genes *A* and *B* overlap, the overlap region is scored. If *A* is longer than *B*, and if *A* scores higher on the overlap region, and if moving *B*’s start site will not resolve the overlap, then *B* is rejected.

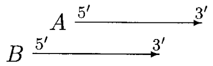
In GLIMMER 2.0, when potential genes *A* and *B* overlap, the overlap region is scored just as in GLIMMER 1.0. The system attempts to move the locations of the start codons much more aggressively, as follows. Suppose gene *A* scores higher, now four different orientations are considered:



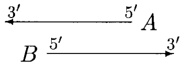
In this case, postponing the start site of either *A* or *B* does not remove the overlap. If *A* is significantly longer than *B* (as determined by a program parameter), then *B* is rejected. Otherwise, both *A* and *B* are called genes, with an annotation that there was a doubtful overlap.



Only moving the start of *B* can resolve the overlap. If it can be moved, then it is. If not, and if *B* is significantly shorter than *A*, then *B* is rejected. Otherwise, both are listed as genes, with a note indicating the overlap. Moving a start codon works as follows: the system shortens the predicted gene by shifting the start location to the next available start codon. If this does not resolve the overlap, it moves the start codon again. This process continues as long as the resulting gene is longer than the minimum gene length (an easily adjustable parameter).



Only moving the start of *A* can resolve the overlap. Since *A* scores higher, we only try to move it if the overlap is a relatively small fraction of *A*’s length. If adjusting *A* is not successful, *B* is rejected.



Both starts can move. We first move the start of *B* until the overlap region scores higher for *B*. Then we move the start of *A* until it scores higher. Then *B* again, and so on, until either the overlap is eliminated or no further moves can be made.

An additional step is taken by GLIMMER 2.0 to help find genes that previously were missed because the score from the independent probability model was too high. The independent probability model is used by both versions of the system to compete against the IMMs used to score all six reading frames; its purpose is to serve as a model of non-coding DNA. In order to be called a gene, an ORF must score higher than the independent model as well as the other five reading frames. Genes that were missed due to high scores from this independent model will fall in between the genes predicted by GLIMMER 1.0. For a target ORF in such regions, GLIMMER 2.0 considers the scores on subsequences of that ORF as compared to other overlapping ORFs. If these subsequences receive sufficiently high scores, and if the ORF scores relatively high in relation to the independent model (even though it did not exceed the normal score threshold to be called a gene), then it is added to the list of prospective genes.

The process of evaluating overlaps in GLIMMER 2.0 is performed in an iterative fashion in order to avoid rejecting genes unnecessarily. For example, in the case where ORF *A* causes ORF *B* to be rejected, and *B* in turn causes *C* to be rejected, we wish to reject only *B* and not both *B* and *C*. Thus, we perform the rejection phase in multiple stages, first discarding *B* and then checking again for overlaps.

COMPUTATIONAL METHODS

We analyzed 10 completed microbial genomes: *Haemophilus influenzae* (14), *Mycoplasma genitalium* (15), *Methanococcus jannaschii* (16), *Helicobacter pylori* (17), *Escherichia coli* (18), *Bacillus subtilis* (19), *Archaeoglobus fulgidus* (20), *B.burgdorferi* (2), *T.pallidum* (3) and *T.maritima* (5). On each of the genomes, we ran both GLIMMER 1.0 and GLIMMER 2.0. All parameters were the defaults, although adjusting these default settings will improve performance on selected genomes. The training data was identical in every case in order to ensure a fair comparison.

The method of training was as follows: using only the genome itself as input, we extracted all ORFs longer than 500 bp from each genome. From these long ORFs, only those that did not overlap other long ORFs were retained; this produces a set of ORFs that are highly likely to be coding. (The programs to perform this extraction are included in the

Glimmer package; total runtime is <1 min on a standard desktop PC.) For all genomes in this study, this set contains more than enough data to train the system accurately.

Next, the IMM training was conducted using the original Glimmer 1.0 program and the new, tree-structured ICMs for Glimmer 2.0. These models were then used to identify genes in the complete genome. For all genomes, ranging in size from 0.5 to 4.7 Mb, training Glimmer 1.0 or Glimmer 2.0 takes <1 min on a Pentium 400 PC running the Linux operating system. The gene finding step takes an additional 1 min or less.

The results of the comparison are summarized in Tables 1–4. In all 10 genomes, there are only 12 confirmed annotated genes that Glimmer 1.0 found that Glimmer 2.0 did not. In all these results, we have not discounted gene predictions that fall into known ribosomal RNA or tRNA regions. Since such regions are easy to identify independently of Glimmer, this step should be a routine part of any annotation process.

Table 1. A comparison of the number of genes correctly found by Glimmer 1.0 and Glimmer 2.0 for 10 complete genomes

		Glimmer 1.0		Glimmer 2.0	
Organism	Genes annotated	Annotated genes found	Additional genes found	Annotated genes found	Additional genes found
H. influenzæ	1738	1715 (98.7%)	234 (13.5%)	1720 (99.0%)	242 (13.9%)
M. genitalium	483	479 (99.2%)	78 (16.1%)	480 (99.4%)	82 (17.0%)
M. jannaschii	1727	1715 (99.3%)	210 (12.2%)	1721 (99.7%)	218 (12.6%)
H. pylori	1590	1545 (97.2%)	293 (18.4%)	1550 (97.5%)	322 (20.3%)
E. coli	4269	4099 (96.0%)	757 (17.7%)	4158 (97.4%)	868 (20.3%)
B. subtilis	4100	4006 (97.7%)	917 (22.4%)	4030 (98.3%)	1022 (24.9%)
A. fulgidus	2437	2385 (97.9%)	274 (11.2%)	2404 (98.6%)	341 (14.0%)
B. burgdorferi	849	845 (99.5%)	67 (7.9%)	843 (99.3%)	62 (7.3%)
T. pallidum	1039	1012 (97.4%)	180 (17.3%)	1014 (97.6%)	250 (24.1%)
T. maritima	1877	1849 (98.5%)	190 (10.1%)	1854 (98.8%)	208 (11.1%)

Table 2. The number of genes with database matches found by Glimmer 1.0 and Glimmer 2.0 for 10 complete genomes

Organism	Genes annotated	Genes with database match	Genes found by Glimmer 1.0	Genes found by Glimmer 2.0
H. influenzæ	1738	1501	1495 (99.6%)	1496 (99.7%)
M. genitalium	483	478	475 (99.4%)	476 (99.6%)
M. jannaschii	1727	1259	1255 (99.7%)	1256 (99.8%)
H. pylori	1590	1092	1083 (99.2%)	1084 (99.3%)
E. coli	4269	2656	2618 (98.6%)	2632 (99.1%)
B. subtilis	4100	1249	1229 (98.4%)	1231 (98.6%)
A. fulgidus	2437	1799	1778 (98.8%)	1786 (99.3%)
B. burgdorferi	849	601	599 (99.7%)	600 (99.8%)
T. pallidum	1039	755	744 (98.5%)	747 (98.9%)
T. maritima	1877	1504	1488 (98.9%)	1493 (99.3%)

Database matches include genes that match genes with unknown function, known as ‘conserved hypotheticals’, as well as genes whose function is known. (Thanks to Alain Viari for testing Glimmer on *B.subtilis*. The 1249 genes listed in the third column for *B.subtilis* were selected according to an even stricter criterion than having a database match; these are the genes that already had been documented in the literature prior to the completion of the *B.subtilis* genome project.)

Table 3. Differences between the length and GC-content of genes that are conserved in other organisms versus ‘hypothetical’ genes

Organism	Conserved genes			Hypothetical genes		
	Number	%GC	Avg len	Number	%GC	Avg len
H. influenzæ	501	39.0	992	237	37.5	502
M. genitalium	478	31.6	1099	5	32.5	453
M. jannaschii	1259	32.8	915	468	29.4	662
A. fulgidus	1799	50.0	907	638	47.0	616
B. subtilis	1249	44.8	1118	2851	44.0	790
E. coli	2656	52.5	1074	1628	50.6	749
H. pylori	1092	40.3	1081	498	37.2	674
B. burgdorferi	601	29.7	1073	248	26.2	818
T. pallidum	755	52.3	1121	284	54.3	766
T. maritima	1504	46.8	1011	373	44.5	706
Averages	1289	42.0	1039	723	40.3	673

The disproportionately small number of conserved genes for *B.subtilis* reflects the fact that this set includes only those genes that were identified experimentally prior to the completion of the genome sequence.

Table 4. Numbers of genes confirmed by database matches found exclusively by Glimmer 1.0, by Glimmer 2.0, and by both systems

Organism	Glimmer 1.0 Only		Glimmer 2.0 Only		Found by both
	Matched	Additional	Matched	Additional	
H. influenzæ	1	48	2	60	1494
M. genitalium	0	12	1	16	475
M. jannaschii	0	27	1	40	1255
H. pylori	1	40	2	73	1082
E. coli	1	176	15	332	2617
B. subtilis	4	163	6	290	1225
A. fulgidus	0	54	8	132	1778
B. burgdorferi	1	24	2	16	598
T. pallidum	2	77	5	146	742
T. maritima	2	65	7	83	1486

The columns labeled ‘Additional’ show how many additional genes are uniquely predicted by each of the two systems respectively. Thus for *H.influenzae*, Glimmer 1.0 predicts 49 genes that Glimmer 2.0 does not, one of which has database homology. Likewise, Glimmer 2.0 predicts 62 genes that Glimmer 1.0 does not, two of which have database matches. They agree on 1494 (out of 1501) gene predictions with database homology.

A second set of experiments was designed to find the true accuracy of Glimmer. In the original study (1), Glimmer 1.0’s gene calls were compared to the published annotation for several completed genomes. The results of this study showed that Glimmer 1.0 was able to find 97–98% of annotated genes fully automatically, using neither database searches nor human intervention; however, published annotation is not 100% accurate. Therefore the question remains open as to how accurate these predictions really are. This second experiment is an attempt to answer that question more precisely.

In order to measure accuracy more precisely, we extracted a subset of genes from the published annotation for each genome. These subsets include only those genes that have significant homology to known proteins, as indicated in the published annotation. Many of these genes have a functional assignment, but some are homologous to other genes of unknown function (these are sometimes annotated as ‘conserved hypothetical’ proteins). We included the latter in the experiment because the existence of homology itself is very strong evidence that the sequence encodes a protein. Except for the use of only a subset of annotated genes, all other details of the experiments were the same as for Table 1. The results of this second comparison are summarized in Table 2.

The results make it clear that Glimmer is more accurate on genes confirmed by sequence homology than it is on the remaining genes. For Glimmer 1.0, sensitivity ranges from 98.4 to 99.7%, with an average of 99.1%. For Glimmer 2.0, the range is 98.6–99.8%, with an average of 99.3%. In contrast, Glimmer 1.0's average accuracy on the complete set of annotated genes for all 10 genomes is 98.1%, and Glimmer 2.0's average on those genes is 98.6%.

Table 3 contains a summary of how the 'confirmed' (or conserved) genes differ from the hypothetical genes in the 10 genomes used in this study. On average, the hypothetical genes are considerably shorter and have ~2% lower GC-content. These data are consistent with the hypothesis that these hypothetical genes contain a significant number of non-coding regions that were mistakenly annotated as coding. (For example, the presence of stop codons alone lowers the average GC-content of non-coding regions.) Most hypothetical gene annotations are based primarily on the predictions of computational systems. The fact that Glimmer is more accurate on conserved genes is suggestive that the hypothetical predicted genes missed by Glimmer are the result of simple disagreement between two computational gene finders.

In each of the 10 genomes, Glimmer 2.0 found more conserved genes than Glimmer 1.0. Usually the number was very small, only 1–5 genes for eight of the genomes. However, the set of conserved genes found by Glimmer 2.0 was not a strict superset of those found by Glimmer 1.0. We intersected the two sets and compared them in order to identify which genes were found by both systems and which were found exclusively by one or the other. These results are shown in Table 4. As the table shows, for each genome there are 0–4 genes found by Glimmer 1.0 and missed by Glimmer 2.0. There are three genomes, *M.genitalium*, *M.jannaschii* and *A.fulgidus*, in which all conserved genes found by Glimmer 1.0 are found also by Glimmer 2.0. Typically, genes found by Glimmer 1.0 but not found by Glimmer 2.0 are relatively short and score just below the minimum scoring threshold. For example, in *B.burgdorferi* the gene found by Glimmer 1.0 and not by Glimmer 2.0 is a 74-amino-acid ribosomal protein S14 (BB0491). The Glimmer 2.0 score for this gene was 88, just below the default threshold value of 90. Such genes could be included in Glimmer 2.0's predictions with suitable parameter adjustments, although at a cost of additional false-positive predictions.

In order to demonstrate that Glimmer 2.0 has a higher sensitivity than alternative gene-finding methods, we analyzed a recently sequenced genome, *Mycobacterium tuberculosis* strain H37Rv (21), for which Glimmer 2.0 was not among the computational methods used for annotation. Table 5 summarizes the genes that were found by Glimmer 2.0 but missed in the original annotation, and that have detectable homology to a coding region from another organism. For each of the 13 genes identified, the table lists the function and identifier of the best hit found by a BLAST search. Eleven of the genes occur in intergenic regions in the published annotation of the complete genome, and the remaining two (those whose closest homologs are P17996 and Q02541) have relatively small overlaps with coding sequences annotated as hypothetical. Glimmer 1.0 finds 11 of these 13 genes, missing those homologous to P17996 and Q02541.

It is worth noting too that the false-positive rate appears to be higher for Glimmer 2.0, as reflected in the fact that the number of additional genes (not confirmed by database matches) predicted by Glimmer 2.0 is higher in nine of the 10 genomes. Because of its revised rules to resolve overlapping ORFs, Glimmer 2.0 generally makes more gene predictions than Glimmer 1.0 when all parameters are set identically as in the above-described results. To verify that the additional annotated matches found by Glimmer 2.0 are not attributable merely to the greater number of predictions, we compared the two systems with Glimmer 1.0's parameters set so that the total additional gene predictions for all 10 genomes matched Glimmer 2.0. Specifically, we raised the overlap-length parameter, which is the maximum number of DNA bases by which two ORFs can overlap and both still be predicted as genes. The results are shown in Table 6. With this adjustment Glimmer 2.0 still finds 99 more annotated genes than Glimmer 1.0, indicating that its predictions are in fact more accurate than Glimmer 1.0. The parameters of either system can be adjusted to reduce the number of additional genes, at the cost of missing some true genes.

Table 5. Genes in *M.tuberculosis* found automatically by Glimmer 2.0 with homology to protein sequences from other organisms

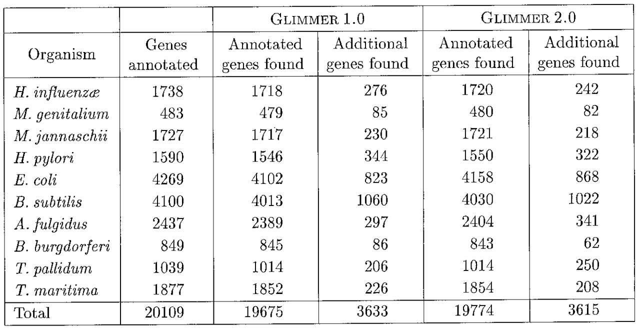
Start	Stop	Length	Accession	Function (Top BLAST Hit in GenBank)	E-Value
591109	591342	234	S72921	B2168_C1_172 protein [*Mycobacterium leprae*]	7e-33
731710	731874	165	P23375	ribosomal protein L33 [*B. stearothermophilus*]	3e-12
1056706	1057068	363	P17996	alpha-antigen A, extracellular [*M. bovis*]	4e-10
1264312	1264551	240	U10895	PcaF [*Pseudomonas putida*]	7e-10
1678940	1679167	228	Q23381	probable methylmalonyl-coA mutase precursor [*C. elegans*]	0.002
1699864	1700223	360	C55239	cpsB 5′-region hypothetical protein [*E. coli*]	2e-09
1999194	1999421	228	L09108	IS401 transposase subunit [*Pseudomonas cepacia*]	8e-13
2943374	2943598	225	P46711	triosephosphate isomerase tpiA [*M. leprae*]	1e-07
3289702	3290229	528	P15026	istB protein (IS21) [*E. coli* plasmid R68.45]	8e-12
3325931	3326098	168	S30383	morphine 6-dehydrogenase [*Pseudomona putida*]	1e-06
3357425	3357225	201	Q02541	CopS [*Pseudomonas syringae*]	4e-08
3568440	3568721	282	S72603	B1937_F2_68 protein [*Mycobacterium leprae*]	3e-20
3571332	3571583	252	Q05266	Mycobacterium phage L5	0.004

All but two (homologous to P15026 and Q02541) of the listed genes are intergenic with respect to the currently published annotation for *M.tuberculosis*. The first three columns list the location of the predicted start and stop codons and the length in base pairs; if Start > Stop then the coding sequence is on the reverse strand. The last three columns give the GenBank accession number, the function of the top hit found by BLAST (23), and the E-value given by BLAST for that hit. (The E-value is the number of homologous sequences expected by chance.)

CONCLUSION

In this paper we have described several technical improvements made in the Glimmer 2.0 gene-finding system and argued that the system is more accurate than previously recognized. Glimmer 2.0 also can be an effective gene finder for eukaryotic genomes, especially those with a high gene density as is found in some parasites. For example, it is being used as the main gene finder for the parasite *Trypanosoma brucei*, the agent that causes African sleeping sickness, which currently is being sequenced at The Institute for Genomic Research. This parasite has few or no introns and a gene density estimated at 50%. The IMM scoring method in GLIMMER 1.0 has also been used to create a eukaryotic gene finder, GLIMMERM, that has been quite successful in finding genes in the genome of *Plasmodium falciparum*, the malaria parasite (22).

Table 6. GLIMMER 1.0 accuracy versus GLIMMER 2.0 accuracy with overlap-length parameter of GLIMMER 1.0 raised to 51



		GLIMMER 1.0		GLIMMER 2.0	
Organism	Genes annotated	Annotated genes found	Additional genes found	Annotated genes found	Additional genes found
H. influenzæ	1738	1718	276	1720	242
M. genitalium	483	479	85	480	82
M. jannaschii	1727	1717	230	1721	218
H. pylori	1590	1546	344	1550	322
E. coli	4269	4102	823	4158	868
B. subtilis	4100	4013	1060	4030	1022
A. fulgidus	2437	2389	297	2404	341
B. burgdorferi	849	845	86	843	62
T. pallidum	1039	1014	206	1014	250
T. maritima	1877	1852	226	1854	208
Total	20109	19675	3633	19774	3615

The value 51 was chosen to make the total number of additional genes found by GLIMMER 1.0 as close as possible to the corresponding number for GLIMMER 2.0. GLIMMER 2.0 still finds significantly more annotated genes than GLIMMER 1.0.

ACKNOWLEDGEMENTS

A.L.D. was supported by NSF Grant IIS-9820497. S.K. was supported by NSF Grant KDI-9980088. O.W. was supported by the Department of Energy Grant No. DE-FC02-95ER61962.A003. S.L.S. was supported by NSF Grant IIS-9902923 and NIH Grants R01 LM06845-01 and K01-HG00022-1. S.L.S. and A.L.D. were supported by NSF Grant IRI-9530462.

REFERENCES

1. Salzberg,S., Delcher,A., Kasif,S. and White,O. (1998) *Nucleic Acids Res.*, **26**, 544–548.
2. Fraser,C.M., Casjens,S., Huang,W., Sutton,G., Clayton,R., Lathigra,R., White,O., Ketchum,K., Dodson,R., Hickey,E. *et al.* (1997) *Nature*, **390**, 580–586.
3. Fraser,C.M., Norris,S.J., Weinstock,G.M., White,O., Sutton,G., Clayton,R., Dodson,R., Gwinn,M., Hickey,E., Ketchum,K.A. *et al.* (1998) *Science*, **281**, 375–388.
4. Stephens,R., Kalman,S., Lammel,C., Fan,J., Marathe,R., Aravind,L., Mitchell,W., Olinger,L., Tatusov,R., Zhao,Q. *et al.* (1998) *Science*, **282**, 754–759.
5. Nelson,K.E., Clayton,R.A., Gill,S.R., Gwinn,M.L., Dodson,R.J., Haft,D.H., Hickey,E.K., Peterson,J.D., Nelson,W.C., Ketchum,K.A. *et al.* (1999) *Nature*, **399**, 323–329.
6. Borodovsky,M. and Mcininch,J.D. (1993) *Comput. Chem.*, **17**, 123–133.
7. Borodovsky,M., McIninch,J., Koonin,E., Rudd,K., Medigue,C. and Danchin,A. (1995) *Nucleic Acids Res.*, **23**, 3554–3562.
8. Quinlan,J.R. (1993) *Programs for Machine Learning*. Kaufmann Publishers, San Mateo, CA.
9. Buntine,W. (1992) *Stat. Comput.*, **2**, 63–73.
10. Helmbold,D.P. and Schapire,R.E. (1997) *Machine Learning*, **27**, 51–68.
11. Willems,F.M.J., Shtarskov,Y.M. and Tjalkens,T.J. (1995) *IEEE Trans. Inf. Theory*, **4**, 663–664.
12. Burge,C. (1998) In Salzberg,S., Searls,D. and Kasif,S. (eds), *Computational Methods in Molecular Biology, New Comprehensive Biochemistry*. Elsevier Science B.V., Amsterdam, pp. 129–164.
13. Salzberg,S., Delcher,A., Fasman,K. and Henderson,J. (1998) *J. Comput. Biol.*, **5**, 667–680.
14. Fleischmann,R.D., Adams,M., White,O., Clayton,R., Kirkness,E., Kerlavage,A., Bult,C., Tomb,J.-F., Dougherty,B., Merrick,J. *et al.* (1995) *Science*, **269**, 496–512.
15. Fraser,C.M., Gocayne,J., White,O., Adams,M., Clayton,R., Fleischmann,R., Bult,C., Kerlavage,A., Sutton,G., Kelley,J. *et al.* (1995) *Science*, **270**, 397–403.
16. Bult,C.J., White,O., Olsen,G.J., Zhou,L., Fleischmann,R.D., Sutton,G.G., Blake,J.A., FitzGerald,L.M., Clayton,R.A., Gocayne,J.D. *et al.* (1996) *Science*, **273**, 1058–1073.
17. Tomb,J.-F., White,O., Kerlavage,A.R., Clayton,R., Sutton,G., Fleischmann,R., Ketchum,K., Klenk,H., Gill,S., Dougherty,B. *et al.* (1997) *Nature*, **388**, 539–547.
18. Blattner,F.R., Plunkett,G., Bloch,C.A., Perna,N.T., Burland,V., Riley,M., Collado-Vides,J., Glasner,J.D., Rode,C.K., Mayhew,G.F. *et al.* (1997) *Science*, **277**, 1453–1462.
19. Kunst,F., Ogasawara,N., Moszer,I., Albertini,A.M., Alloni,G., Azevedo,V., Bertero,M.G., Bessieres,P., Bolotin,A., Borchert,S. *et al.* (1997) *Nature*, **390**, 249–256.
20. Klenk,H.P., Clayton,R.A., Tomb,J.-F., White,O., Nelson,K.E., Ketchum,K.A., Dodson,R.J., Gwinn,M., Hickey,E.K., Peterson,J.D. *et al.* (1997) *Nature*, **390**, 364–370.
21. Cole,S.T., Brosch,R., Parkhill,J., Garnier,T., Churcher,C., Harris,D., Gordon,S.V., Eiglmeier,K., Gas,S., Barry,C.E. *et al.* (1998) *Nature*, **393**, 537–544.
22. Salzberg,S.L., Pertea,M., Delcher,A.L., Gardner,M.J. and Tettelin,H. (1999) *Genomics*, **59**, 24–31.
23. Altschul,S., Gish,W., Miller,W., Myers,E. and Lipman,D. (1990) *J. Mol. Biol.*, **215**, 403–410.