BIOINFORMATICS

Vol. 23 ISMB/ECCB 2007, pages i97–i103
doi:10.1093/bioinformatics/btm225

Homology search for genes

Xuefeng Cui[1], Tomáš Vinař[2], Broňa Brejová[2], Dennis Shasha[3] and Ming Li[1,*]

[1]Cheriton School of Computer Science, University of Waterloo, Ontario, Canada N2L 3G1, [2]Department of Biological Statistics and Computational Biology, Cornell University, Ithaca, NY 14853, USA and [3]Department of Computer Science, New York University, NY 10012, USA

ABSTRACT

Motivation: Life science researchers often require an exhaustive list of protein coding genes similar to a given query gene. To find such genes, homology search tools, such as BLAST or PatternHunter, return a set of high-scoring pairs (HSPs). These HSPs then need to be correlated with existing sequence annotations, or assembled manually into putative gene structures. This process is error-prone and labor-intensive, especially in genomes without reliable gene annotation.

Results: We have developed a homology search solution that automates this process, and instead of HSPs returns complete gene structures. We achieve better sensitivity and specificity by adapting a hidden Markov model for gene finding to reflect features of the query gene. Compared to traditional homology search, our novel approach identifies splice sites much more reliably and can even locate exons that were lost in the query gene.

On a testing set of 400 mouse query genes, we report 79% exon sensitivity and 80% exon specificity in the human genome based on orthologous genes annotated in NCBI HomoloGene. In the same set, we also found 50 (12%) gene structures with better protein alignment scores than the ones identified in HomoloGene.

Availability: The Java implementation is available for download from http://www.bioinformatics.uwaterloo.ca/software

Contact: mli@uwaterloo.ca

1 INTRODUCTION

Sequence homology search has been a core topic in the bioinformatics literature since the seminal paper introducing BLAST (Altschul *et al.*, 1990). The focus of the field is on designing faster and more sensitive methods to search for sequences similar to a query DNA or protein sequence in one or more huge databases [see, e.g. Kisman *et al.* (2005); Ma *et al.* (2002)]. The similarity measure reflects the likelihood of two sequences to be evolutionarily related. Thus, a typical homology search returns a set of high-scoring pairs (HSPs), consisting of the subsequences of the database and the query sequence that can be aligned to one another with a high similarity score.

Researchers studying the function of a particular gene or a gene family often need to locate genes that are similar to a query gene, either within the same species or in related organisms. Such queries are complicated by the fact that the protein-coding segments of genes (*exons*) are interrupted by non-coding segments (*introns*). Intronic sequences diverge much faster than coding sequences because they are under much weaker evolutionary constraints.

A typical homology search for a query gene in a target genome will return HSPs that roughly correspond to exons of that gene. In a well-annotated target genome this is often sufficient: we can examine annotated genes that overlap these HSPs. However, only few genomes have reliable annotations to date; indeed, even in the human genome, the annotation is still not complete (Guigo *et al.*, 2006). In the absence of a reliable annotation, only a crude approximation of the correct intron/exon structure of the gene can be recovered from these HSPs. However, a correct intron/exon structure is essential for subsequent analysis, e.g. for protein folding, or for scans for positive selection.

In this article, we explore the problem of fast search for homologous genes: for a given query gene (together with its intron/exon structure in the query genome), we want to locate similar genes in the target genome, including their intron/exon structure.[1]

Our problem is on the boundary of homology search, gene finding and gene mapping. Indeed, for each of these tasks, there are established tools. However, none of these tools is directly applicable to our problem. The homology search programs such as BLAST (Altschul *et al.*, 1990) or PatternHunter (Ma *et al.*, 2002) will only return approximate locations of exon boundaries. Moreover, they are not likely to locate exons that are less than 25 nt long (Volfovsky *et al.*, 2003) and exons that were inserted or lost in one of the sequences. Gene prediction programs, such as GENSCAN (Burge and Karlin, 1997), Augustus (Stanke and Waack, 2003) or ExonHunter (Brejová *et al.*, 2005), can do a reasonable job on the genomic scale, however, the predictions for a particular gene can be of low quality. Additionaly, parametric files for gene finders are often available only for a few species, and retraining gene finders for new species is non-trivial. Finally, gene mapping tools for aligning ESTs and mRNAs to a genome, such as sim4 (Florea *et al.*, 1998) or BLAT (Kent, 2002), are designed to work only within the same species, and they also fail to locate short and missing exons.

We introduce a novel method for finding homologous genes that starts from HSPs reported by a homology search program. Based on these HSPs, we identify possible locations of a gene, and use an HMM-based extension step to identify candidates for homologous genes. The key insight in our method is that

*To whom correspondence should be addressed.

[1]In our work, we measure similarity of two genes at the potein level using traditional amino acid scoring matrices; however, other similarity measures are possible.

our HMM and algorithm for its decoding are tuned to identify genes similar to the query gene, thus achieving better performance than an ordinary gene finder on these genes. Finally, we align the predicted gene structures to the original query and rank them based on the protein similarity score.

We validate our method using a set of 400 one-to-one orthologous genes identified in NCBI HomoloGene (Wheeler *et al*., 2006). Our tests indicate that for alignments between close species (human-mouse), our approach works very well, identifying 79% of exons correctly on both boundaries. In the same set, we also identified 50 putative human gene transcipts that have better protein similarity to mouse genes than their orthologs identified in HomoloGene. We also observe that the performance of our method deteriorates with evolutionary distance.

Two closely related methods, GeneWise (Birney *et al*., 2004) and Projector (Meyer and Durbin, 2004), place emphasis on the second stage of the search for homologous genes. Both of these methods use pair HMMs to align the protein sequence (GeneWise) or DNA sequence with annotation (Projector) to a novel DNA sequence. In our approach, we also locate the target region for this alignment, and identify the extent of the homologous gene alignment within the target genome. Since locating the correct extent of a gene is one of the hardest problems in gene finding, applying those tools may be hard in the case of a genome-wide search for homologous genes. Moreover, pair HMMs require computationally more intensive algorithms and larger training sets.

2 RESULTS

Our method uses four steps to search for homologous genes. First, we find HSPs between the query gene and the target genomic sequence using a seed-based homology search method. Since finding the HSPs with high sensitivity is critical for the later steps, we use TBLASTN (Altschul *et al*., 1990). TBLASTN searches the query protein sequence against the nucleotide sequence translated in all six frames. To avoid problems with frame shifts within the HSPs, we forbid gaps in the TBLASTN settings. This also has a side effect of speeding-up the homology search.

Second, based on the HSPs, we locate several *target regions*: the sections of the target sequence that likely contain genes homologous to the query. Each target region is characterized by a triple $(I, \lambda_5, \lambda_3)$, where I is the *initiator* position that has been identified as likely to occur within the coding sequence of a target gene, λ_5 is the expected length of the coding region on the $5'$ side of the initiator and λ_3 is the expected length of the coding region on the $3'$ side of the initiator. We describe and evaluate this step in more detail in Section 3.1.

Third, we use a modified bi-directional Viterbi algorithm to extend initiator I in both directions to form a viable gene structure. As a basis for this step, we use a hidden Markov model (HMM) that is *biased* towards the query gene. In particular, parameters of this HMM are computed as a linear combination of the parameters estimated from the target genome (or from a closely related genome), and the observations from the query gene sequences. The bias is weighted by a mixing coefficient w. Our bi-directional Viterbi algorithm does not require a fixed window in the sequence. Instead, we use parameters λ_3 and λ_5 estimated in the previous step to dynamically locate likely start and stop sites. Our bi-directional Viterbi algorithm also makes use of the HSP information from the first step (though this is not necessary to achieve good performance). We describe our models and algorithms in Sections 3.2–3.4.

In the final step, we realign all predictions to the query gene at the protein level, and rank them by the alignment score.

We validate our approach on a testing set of 400 one-to-one orthologous gene pairs from human and mouse derived from NCBI HomoloGene database (see Section 3.5). We use the mouse genes from the pairs as the query genes with mixing coefficient $w = 0.5$, and compare the best-scoring result with the target gene in the human genome using traditional measures from gene finding: sensitivity and specificity on exon and nucleotide levels (Keibler and Brent, 2003). The results of this experiment are presented in Table 1.

On the exon level, our approach achieves 79% sensitivity and 80% specificity. This is much better than a typical *ab initio* gene finder (e.g. GENSCAN has 71% sensitivity and 50% specificity). Note that in our experiment GENSCAN had an easier task, since we considered only its predictions that overlapped with the true target genes, while in evaluating our approach, we considered all highest-scoring predictions, including those that did not overlap with the target region. We also compared our results to HSPs located by TBLASTN. On the nucleotide

Table 1. Sensitivity and specificity of our homology search, TBLASTN alone, our HMM without the bias and length penalty and GENSCAN *ab initio* predictions

	Mouse query genes		Chicken query genes	
	Exon SN/SP	Nucleotide SN/SP	Exon SN/SP	Nucleotide SN/SP
Our homology search	**79%/80%**	85%/**87%**	38%/41%	56%/52%
TBLASTN	7%/5%	**91%** /83%	3%/2%	71%/61%
Generic HMM	41%/61%	57%/84%	27%/**44%**	46%/**72%**
GENSCAN	71%/50%	88%/57%	**65%**/43%	**88%**/58%

Sensitivity (SN) is measured as the percentage of correctly identified exons or coding nucleotides out of all true exons or coding nucleotides. Similarly, specificity (SP) is the percentage of correctly predicted exons or coding nucleotides out of all predicted exons or coding nucleotides.

level, TBLASTN performs quite well, but it cannot predict the exact location of exon boundaries. Finally, we also tested the accuracy of the HMM underlying our method, without the benefits of biasing or length distribution (but including penalties for disagreement with HSPs). We see that this simple HMM does not compare very well to GENSCAN, but biasing and length penalties lead to a great improvement.

We did a similar evaluation on a testing set of 400 chicken-human orthologs. Here we used a lower mixing coefficient $w = 0.25$, because chicken is evolutionary more distant from human and thus there will be less conservation between the chicken query gene and the human target gene. The lower accuracy in this case is a result of rather simple method we used for biasing model parameters. We believe that these results can be substantially improved by taking into account the increased likelihood of silent mutations, effects of which become noticeable at this evolutionary distance.

Interestingly, we were able to locate 50 human genes (12% of the human-mouse testing set) that had better protein alignment scores to their mouse query genes than the orthologous human genes identified by HomoloGene. On the chicken-human testing set, this number was even higher (107 genes or 27%). We hypothesize that at least some of these genes represent better predictions of homologous genes than the ones found by HomoloGene.

Figure 1 shows one such case, and compares our prediction to the HomoloGene target, BLAT alignment and GENSCAN prediction. A BLAT alignment of the mouse gene to the human sequence does not cover all exons and contains a false match. Our prediction has one more exon than the HomoloGene target. However, our prediction is identical to an experimentally confirmed splicing isoform present in the Mammalian Gene Collection of full-length clones (Gerhard *et al.*, 2004). Therefore, we are confident that our prediction is correct. GENSCAN predicted several spurious exons and extended the gene on the 3′ end. This example demonstrates that our method can identify exons gained and lost over the course of evolution, and that its predictions can be (at least in some cases) of better quality than those of other methods.

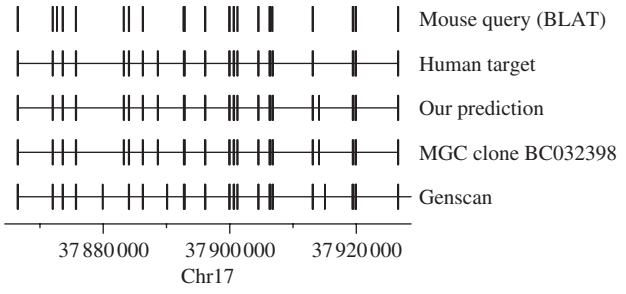

Fig. 1. Example of a gene structure predicted by our system.

3 METHODS

3.1 Using HSPs to locate target regions

First, we remove the HSPs with scores lower than 55. We have chosen this threshold to balance sensitivity and specificity of the search using the validation data set of human-mouse homologous genes. This validation set does not overlap with the testing set used in Section 2. To locate the target regions, we cluster the remaining HSPs based on their distance. Two clusters are merged if the corresponding HSPs are on the same strand and their distance is smaller than a given threshold.

It is difficult to estimate where the target region starts and ends, especially since the 5′ and 3′ ends of the query gene and the target gene may not be conserved. We describe each target region by a triple $(I, \lambda_5, \lambda_3)$. Initiator I is the position of the middle nucleotide of the highest scoring HSP within the target region; this is very likely to be located within a protein coding exon. On the validation data, we have observed that lengths of coding sequences change only slowly during the course of evolution. Thus, λ_5 will characterize the expected coding length of the 5′ end based on the position of I within the query sequence and λ_3 will characterize the expected coding length of the 3′ end (see Fig. 2).

We use several best-scoring target regions. Table 2 shows the effect of using multiple target regions on sensitivity; to be able to locate the target gene, one must choose a target region that overlaps that gene. We also use multiple initiators in each region (the k-th additional initiator is based on the k-th highest-scoring HSP). Table 3 shows the effect of using multiple initiators. In these two tables, the sensitivity is measured as the fraction of the target genes that have at least one initiator within their coding region. In general, increasing the number of target regions and initiators helps to increase overall sensitivity, however, it also slows down the search, because we run a separate instance of the bi-directional Viterbi algorithm for each target triple.

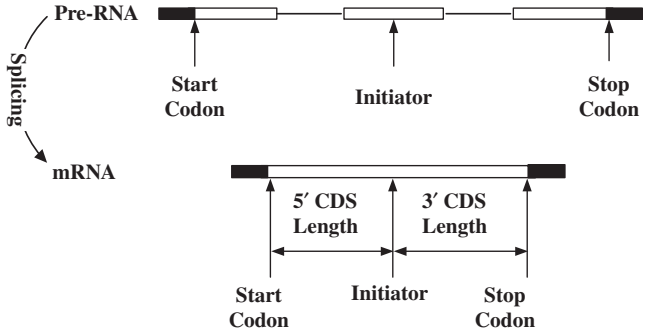

Fig. 2. Initiator and its associated 5′ and 3′ CDS lengths. White boxes represent exons, black boxes represent untranslated regions and lines represent introns.

Table 2. Sensitivity increases using multiple target regions

Number of target regions	1	3	5	7	9
Mouse	93.0%	98.5%	99.0%	99.0%	99.5%
Chicken	88.5%	95.5%	97.5%	97.5%	98.5%

Table 3. Sensitivity increases using multiple initiators

Number of initiators	1	2	3	4	5
Mouse	97.5%	98.5%	98.5%	98.5%	98.5%
Chicken	92.5%	94.0%	94.5%	95.0%	95.0%

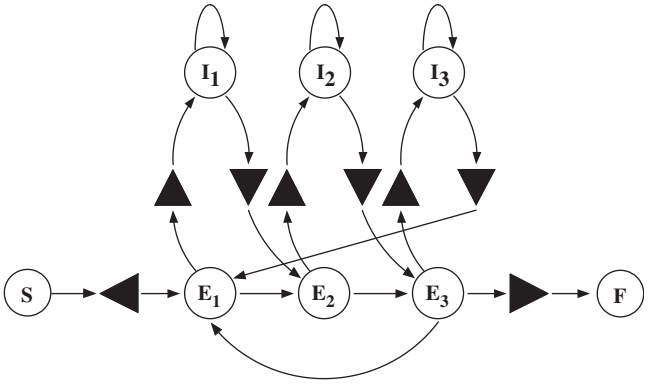

Fig. 3. HMM for gene homology search. State S is the silent initial state, and state F is the silent final state. States E_i represent coding exons, and states I_i represent introns. The triangles represent submodels for signals (start, stop, donor and acceptor site signals).

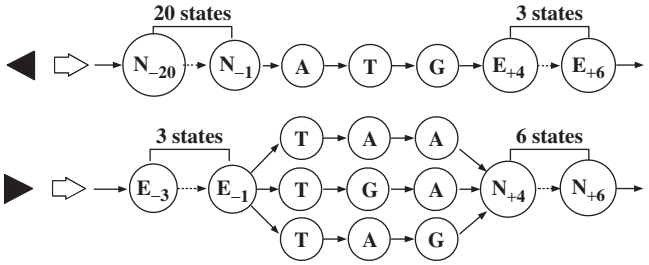

Fig. 4. Start/stop signal submodels. States E_i represent coding exons, and states N_i represent non-coding (untranslated or intergenic) sequences. The states showing a single symbol emit only that symbol.

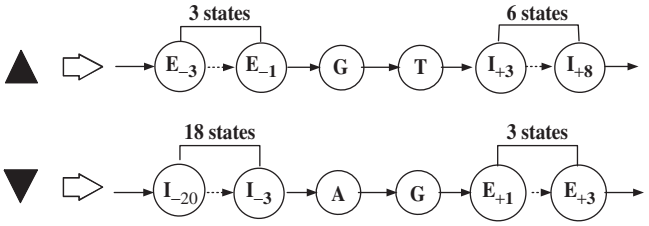

Fig. 5. Donor/acceptor signal submodels. States E_i represent coding exons, and states I_i represent introns. The states showing a single symbol emit only that symbol.

In our experiments, we use three target regions with three initiators each. In general, the number of target regions can be chosen by a user depending on whether they expect to find a single homolog or multiple homologs for the query gene.

3.2 Biased hidden Markov models

To find candidate gene structures, we use a simple HMM shown in Figure 3. This HMM models a single gene on the forward strand without untranslated regions or intergenic regions. The submodels for translation start and stop signals are shown in Figure 4, and the submodels for donor and acceptor signals are shown in Figure 5. States representing exons and introns are Markov chains of order 4, states representing signals are Markov chains of order 2. To simplify notation, we give formulas only for the zeroth-order Markov chain states; the formulas can be easily extended to higher-order states.

This HMM represents the joint probability $\Pr(G, H)$ of generating a genomic sequence G by a state path H. For a given sequence G, we can compute a state path H maximizing $\Pr(H \mid G)$ using the efficient Viterbi algorithm in $O(nm^2)$ time, where n is the length of the sequence, and m is the number of states of the HMM (Rabiner, 1989). Such a state path segments the sequence into introns and exons, thus giving the most likely gene prediction according to the model.

To complete the model, we need to train the probabilities of transitions between the states of the HMM and the probabilities of emissions of sequence symbols by the states of the HMM. We use the training set of genes in the target genome to count the observed number of transitions between states x and y, $T(x, y)$, and the number of emissions of a nucleotide n in state x, $E(x, n)$.

Each gene has its own structural properties, such as a characteristic distribution of nucleotides in the coding sequences. Many of these properties are conserved for homologous genes throughout the evolution. Thus, in addition to the genome-specific training data set, we also use the query gene to train the parameters of the HMM; in other words, we *bias the HMM* towards the query gene. Let $T_Q(x, y)$ be the number of transitions from state x to state y observed in the query gene, and $E_Q(x, m)$ be the number of nucleotides n emitted by state x observed in the query gene. We use a mixing coefficient w to regulate the impact of observations from the query gene on the HMM.

Formally, we set the transition probabilities $P_t(x, y)$ that the HMM transitions from state x to state y as

$$P_t(x, y) = \frac{T(x, y) + wMT'(x, y) + C}{\sum_i T(x, i) + wMT'(x, i) + C}, \tag{1}$$

and the emission probabilities $P_e(x, n)$ of emitting nucleotide n in state x as

$$P_e(x, n) = \frac{E(x, n) + wME'(x, n) + C}{\sum_{j \in \{A,C,G,T\}} E(x, j) + wME'(x, j) + C}. \tag{2}$$

Here, M is the number of genes in the training set, and C is a pseudocount. Intuitively, the biased training is equivalent to training on a virtual training data set enriched by wM copies of the query gene. We do not use the biased training for emission probabilities of intron states (which is equivalent to setting $w = 0$ for these emission probabilities). Our biased training is related to the Gibbs sampling approach of Chatterji and Pachter (2004) that modifies parameters of a gene finder based on a set of homologous genes in multiple species. However, while they are finding the genes in all the species simultaneously, the gene structure in the query sequence is fixed in our case.

Table 4 shows the effect of biased training on the validation set of mouse-human and chicken-human orthologs. We used the mouse and chicken genes as queries, and, to avoid the effects of an incorrect initiator, we used an 'ideal' initiator located in the middle of the coding sequence of the target gene. We used several mixing coefficients w.

For the mouse query genes, we achieve the best exon sensitivity and specificity with $w = 0.5$. Biased training improves exon sensitivity by 38% and exon specificity by 31%. For the chicken query genes, we achieve the best results when $w = 0.25$. However, the best exon sensitivity and specificity is still low (49%/54%). To achieve practical results for chicken genes, we need to explore more complicated ways of biasing HMM parameters, taking into account increased likelihood in synonymous changes on protein level.

Table 4. Effect of mixing coefficient w on sensitivity and specificity of gene homology search on the validation set

	w	Exon SN/SP	Nucleotide SN/SP
Mouse	0	41%/50%	54%/55%
	0.25	76%/79%	84%/84%
	0.5	79%/81%	87%/87%
	0.75	79%/80%	87%/87%
	1.0	77%/78%	85%/86%
Chicken	0	34%/41%	49%/51%
	0.25	49%/54%	64%/66%
	0.5	45%/51%	62%/64%
	0.75	34%/45%	52%/55%
	1.0	29%/37%	44%/47%

Table 5. Sensitivity and specificity of gene homology search with and without the CDS length penalty on the validation set

		Exon SN/SP	Nucleotide SN/SP
Mouse	with	79.30%/80.77%	86.70%/86.53%
	without	1.48%/7.38%	8.72%/69.72%
Chicken	with	49.37%/54.33%	63.77%/65.79%
	without	1.97%/8.07%	10.56%/75.45%

3.3 Bi-directional Viterbi algorithm and coding length penalties

Because we cannot easily identify the extent of the homologous target gene, we cannot use the Viterbi algorithm directly to recover the most probable gene structure. Here, we modify the Viterbi algorithm to start from the initiator I, and proceed in both directions. From the alignment corresponding to the initiator, we can determine state S_I in which the HMM should generate the nucleotide at the initiator position.

Recall that we describe the target region not only by its initiator, but also by the expected coding length λ_5 on the 5′ side of the initiator, and λ_3 on the 3′ side of the initiator. Let $\lambda = \lambda_5 + \lambda_3$ be the coding length of the query gene. Below, we will define the probability distribution $\Pr(\hat{\lambda} \mid \lambda)$ over possible coding lengths of genes homologous to a gene with coding length λ.

In our algorithm, we aim to compute the state path H^* through the HMM, and values i and j defining the extent of the homologous gene, maximizing the following quantity:

$$\Pr(\lambda_{H^*} \mid \lambda) \cdot \Pr(H^* \mid G_{i\ldots j}) = \Pr(\lambda_{H^*} \mid \lambda) \cdot \frac{\Pr(G_{i\ldots j}, H^*)}{\sum_H \Pr(G_{i\ldots j}, H)}, \quad (3)$$

where λ_{H^*} is the coding length defined by the state path H^*. Note that $\Pr(H^* \mid G_{i\ldots j})$ is normalized over the length of the sequence $j-i$, therefore it is appropriate to assign an additional length penalty.

In the rest of this section, we discuss the details of this computation. We did not find an efficient algorithm that would optimize (3) globally. Instead, we consider potential state paths H^* that maximize $\Pr(H^* \mid G_{i\ldots j})$ for all values of i and j, and from these candidates we select the final state path by multiplying their probabilities by the length penalty and choosing the highest-scoring path.

3.3.1 Coding length penalties. To model the evolution of lengths of coding sequences, we adopt the approach of Burge (1997). Under this model, we assume that only a single codon insertion or deletion can occur in a single generation, and the probabilities of insertion and deletion do not change. In addition, we assume that the evolution of each individual exon is independent of those of other exons. Subject to these conditions, the exon lengths will be independent normally distributed variables (Burge, 1997). Since the length of the coding sequence is the sum of the exon lengths, we can approximate the probability of coding length λ becoming coding length $\hat{\lambda}$ in a homologous gene by a normal distribution with mean λ and variance 2λ (the variance being set somewhat arbitrarily):

$$\Pr(\hat{\lambda}|\lambda) = \frac{1}{\sqrt{4\pi\lambda}} \cdot e^{-\frac{(\hat{\lambda}-\lambda)^2}{4\lambda}}. \quad (4)$$

We have studied the effects of the CDS length penalty on our validation sets (see Table 5). Again, we have used the ideal initiator instead of alignments and $w = 0.5$ for mouse query genes, and $w = 0.25$ for chicken query genes. Without the CDS length penalty, our HMM favors short gene structures and achieves very poor sensitivity and specificity. The performance is significantly improved by using the CDS length penalty.

3.3.2 Bi-directional Viterbi algorithm. Using the initiator I as an anchor, we can decompose the computation probability on the 5′ side of the initiator, and probability on the 3′ side of the initiator. In particular, for all $i < I$, let $P_v(i, x)$ be the probability of the most probable HMM state path generating the sequence $G_{i\ldots I-1}$, starting in state x and ending in state S_I. Similarly, for $i > I$, let $P_v(i, x)$ be the probability of the most probable HMM state path generating the sequence $G_{I+1\ldots i}$, starting in state S_I, and ending in state x. We can compute these probabilities using the following recurrence:

$$P_v(i,x) = \begin{cases} 1, & \text{if } i = I \text{ and } x = S_I; \\ 0, & \text{if } i = I \text{ and } x \neq S_I; \\ P_e(x, G_i) \cdot \max_y\{P_v(i-1, y) \cdot P_t(y, x)\}, & \text{if } i > I_0; \\ P_e(x, G_i) \cdot \max_y\{P_v(i+1, y) \cdot P_t(x, y)\}, & \text{if i} < \text{I}_0. \end{cases} \quad (5)$$

Using the above recurrence, we can compute each column of $P_v(i, *)$ in time $O(m^2)$ from either the previous column, if $i > I$, or from the following column if $i < I$. Similarly, we can modify the forward and backward algorithm to compute the normalizing coefficient in (3).

3.3.3 Stopping condition. We have shown how to compute candidates for the sections of the most probable state path to the left of the initiator I with decreasing $i < I$. Each of these candidates needs to be multiplied by an appropriate length penalty. Note that if the current highest-scoring candidate for the left section has score Q, we do not need to consider any lengths $\lambda'_5 > \lambda_5$ for which $\Pr(\lambda'_5 \mid \lambda_5) < Q$. Thus, we can stop extending each particular state path to the left whenever this condition is reached. Analogous condition can be applied to the sections of the most probable state path to the right of the initiator I. Reaching this stopping condition is not guaranteed. Therefore we also limit the maximum extension length by a large upper bound of 650 Kb (approximately half of the length of the longest known gene NM_013988 on human chromosome 6).

3.4 Using HSPs in Viterbi algorithm

We also devised a simple method for incorporating information from HSPs beyond the creation of initiators. For a given state

Table 6. Sensitivity and specificity of gene homology search with and without the alignment evidence on the validation set

		Exon SN/SP	Nucleotide SN/SP
Mouse	using	81.82%/81.98%	89.27%/89.44%
	not using	79.30%/80.77%	86.70%/86.53%
Chicken	using	50.86%/56.75%	65.80%/67.83%
	not using	49.37%/54.33%	63.77%/65.79%

Table 7. Overview of the validation data set of homologous genes

	Human-mouse		Human-chicken	
Number of genes	200	200	200	200
Average gene length	33 284	24 781	46 470	22 277
Average CDS length	1337	1326	1406	1526
Average number of exons	11	10	8	9

path H, we use a bonus factor between $B_{\min}$ and $B_{\max}$ to multiply the probabilities in states which are consistent with HSPs. The bonus factor is largest in the middle of an HSP and decreases linearly with the distance from the middle, being lowest at the boundary of the HSP. We also use a constant multiplicative penalty factor P for the states that are inconsistent with HSPs. We set the values manually ($B_{\min} \sim 3$, $B_{\max} \sim 12$ and $P \sim 0.2$). Variants of the Viterbi algorithm can be easily modified to accommodate such bonuses and penalties (Brejová *et al.*, 2005). This addition improves the results only slightly (see Table 6).

3.5 Data sets

3.5.1 Training data set of human genes. We trained the target genome portion of the transition and emission probabilities of the HMM on a data set of 1070 human genes. We have started from the data set of Stanke and Waack (2003) of gene finder AUGUSTUS. The original data set contained 1284 human genes retrieved from NCBI GenBank in October 2002, and we removed 214 genes that did not satisfy various technical requirements of our HMM in Figure 3. Specifically, start, stop, donor and acceptor sites have to conform to canonical consensus sequences; there must be at least 7 nt in each exon, and at least 29 nt in each intron; each reference sequence must include at least 20 nt before the start site, and at least 6 nt after the stop codon. The genes in the resulting training set have an average gene length of 6477 nt, an average CDS length of 1196 nt, and an average five exons per gene.

3.5.2 Validation and testing data sets of homologous genes. The validation data set of homologous genes contains 200 pairs of human-mouse and 200 pairs of human-chicken homologous genes. We use this set to study the parameters of our method for gene homology search, such as the number of target regions per target gene, the number of initiators per target region, and the mixing coefficient w. The basic statistics of the validation data set are shown in Table 7.

Such a validation data set is not required to extend our method to additional species, and it was used mainly to demonstrate how to set the mixing coefficient w and number of initiators. We expect that the same settings can be applied to other pairs of genomes at similar evolutionary distances. The HMM was trained only with a single-species data set in Section 3.5.1. Hence the major difference of our approach from that of GeneWise and Projector is that we train only on genes from one (the target) genome as required by the nature of our problem, whereas pair HMM approaches (such as Projector) are trained with homologous genes from a pair of genomes.

Table 8. Overview of the testing data set of homologous genes

	Human-mouse		Human-chicken	
Number of Genes	400	400	400	400
Average Gene length	36 537	28 120	49 486	23 315
Average CDS length	1337	1323	1474	1633
Average number of exons	10	10	9	10

The testing data set of homologous genes contains the reference sequences and the CDS annotations of a seperate 400 pairs of human-mouse and 400 pairs of human-chicken homologous genes. We use the testing data set to evaluate the performance of our gene homology search method with all the parameter values fixed. The basic statistics of the testing data set are shown in Table 8.

We have built the validation and testing data sets of homologous genes from NCBI HomoloGene 49.1 released on 28 April 2006 (Wheeler *et al.*, 2006) which is a database of groups of homologous genes of several completely sequenced eukaryotic genomes. We have used the following protocol:

(1) We filtered all homologous gene groups in NCBI HomoloGene, and retained those that contain one gene from each of the two species (human-mouse, or human-chicken). We also removed the genes from other than the two species.

(2) We used GMAP (Wu and Watanable, 2005) to map each gene to its genome. Then we removed genes whose mapped CDS annotation was inconsistent with the CDS annotation in NCBI GenBank. Here, two annotations were consistent if and only if they had the same number of exons, and each pair of corresponding exons had the same length.

(3) We randomly selected 200 pairs of human-mouse and 200 pairs of human-chicken homologous genes as the validation data set. We randomly selected 400 pairs of human-mouse and 400 pairs of human-chicken homologous genes as the testing data set. The validation and testing data sets do not overlap.

3.5.3 Target genomic sequence. In all experiments requiring the target genomic sequence, we used the NCBI human genome 36.1 released in March 2006 as the target genomic sequence. We downloaded the complete reference sequences with a total size of ~3.14 GB from the UCSC genome browser (Kuhn *et al.*, 2007). Repeats in the genomic sequences were masked by RepeatMasker (Smit *et al.*, 2006) and Tandem Repeats Finder (Benson, 1999).

4 CONCLUSION

This article has introduced a novel method for finding homologous genes on a genome-wide scale. Our method uses HSPs returned by a classical sequence alignment program, such as TBLASTN, as a starting point, identifies target regions where homologous genes are likely to be located, and uses biased HMMs to identify homologous genes in these regions. We have demonstrated good performance in identifying

homologous genes in the human genome where query genes came from the mouse genome. We also demonstrated that our method can successfully identify exons that were lost or gained in the course of evolution. Thus, we conclude that our method is well suited for locating homologous genes between close species. Our method will be useful to life science researchers who study evolutionary histories of genes and gene families especially among closely related species. The main advantage of our method is that it returns complete gene structures, not only fragments of gene candidates.

Our experiments also show that our current implementation does not work well for more distant species, such as human and chicken. We believe that this can be remedied by incorporating a more complex HMM combined with a more careful mixing of the HMM parameters.

ACKNOWLEDGEMENTS

We would like to thank Daniel G. Brown and Brendan McConkey for insightful comments. Research of X.C. and M.L. is supported by NSERC, CITO, and Canada Research Chair program. Part of the work of T.V. and B.B. was done while at the University of Waterloo and was supported by NSERC, CITO and Canada Research Chair program. Research of T.V. at Cornell is supported by NSF grant DBI-0644111 and NSF/NIGMS grant DMS-0201037. Research of B.B. at Cornell is supported by NIH/NCI (subcontract 22XS013A). Research of D.S. is supported by NSF grants DBI-044566, N2010 IOB-0519985, N2010 DBI-0519984, DBI-0421604 and MCB-0209754. These supports are greatly appreciated.

Conflict of Interest: none declared.

REFERENCES

Altschul,S.F. *et al.* (1990) Basic local alignment search tool. *J. Mol. Biol.*, **215**, 403–410.

Benson,G. (1999) Tandem repeats finder: a program to analyze DNA sequences. *Nucleic Acids Res.*, **27**, 573–580.

Birney,E. *et al.* (2004) GeneWise and GenomeWise. *Genome Res.*, **14**, 988–995.

Brejová.B *et al.* (2005) ExonHunter: a comprehensive approach to gene finding. *Bioinformatics*, **21**, i57–i65. Intelligent Systems for Molecular Biology (ISMB 2005).

Burge,C. (1997) *Identification of Genes in Human Genomic DNA*. PhD thesis, Stanford University.

Burge,C. and Karlin,S. (1997) Prediction of complete gene structures in human genomic DNA. *J. Mol. Biol.*, **268**, 78–94.

Chatterji, S. and Pachter, L. (2004) Multiple organism gene finding by collapsed Gibbs sampling. In *Proceedings of the Eighth Annual International Conference on Research in Computational Molecular Biology (RECOMB)*, ACM Press, New York, NY, USA pp. 187–193.

Florea,L. *et al.* (1998) A computer program for aligning a cDNA sequence with a genomic DNA sequence. *Genome Res.*, **8**, 967–974.

Gerhard,D.S. *et al.* (2004) The status, quality, and expansion of the NIH full-length cDNA project: the Mammalian Gene Collection (MGC). *Genome Res.*, **14**, 2121–2127.

Guigo,R. *et al.* (2006) EGASP: the human ENCODE Genome Annotation Assessment Project. *Genome Biol.*, **7**, 1–31.

Keibler,E. and Brent,M.R. (2003) Eval: a software package for analysis of genome annotations. *BMC Bioinformatics*, **4**, 50.

Kent,W.J. (2002) BLAT: the BLAST-like alignment tool. *Genome Res.*, **12**, 656–664.

Kisman,D. *et al.* (2005) tPatternHunter: gapped, fast and sensitive translated homology search. *Bioinformatics*, **21**, 542–544.

Kuhn,R.M. *et al.* (2007) The UCSC genome browser database: update 2007. *Nucleic Acids Res.*, **35**, 668–673. http://genome.ucsc.edu/

Ma,B. *et al.* (2002) PatternHunter: faster and more sensitive homology search. *Bioinformatics*, **18**, 440–445.

Meyer,I.M. and Durbin,R. (2004) Gene structure conservation aids similarity based gene prediction. *Nucleic Acids Res.*, **32**, 776–783.

Rabiner,L.R. (1989) A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. of the IEEE*, **77**, 257–286.

Smit,A. *et al.* (2006) RepeatMasker *http://www.repeatmasker.org/*.

Stanke,M. and Waack,S. (2003) Gene prediction with a hidden Markov model and a new intron submodel. *Bioinformatics*, **19**, 215–225. European Conference on Computational Biology (ECCB 2003).

Volfovsky, N. *et al.* (2003) Computational discovery of internal micro-exons. *Genome Res.*, **13**, 1216–1221.

Wheeler,D.L. *et al.* (2006) Database resources of the National Center for Biotechnology Information. *Nucleic Acids Res.*, **34**, 173–180. http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=homologene

Wu,T.D. and Watanabe,C.K. (2005) GMAP: a genomic mapping and alignment program for mRNA and EST sequences. *Bioinformatics*, **21**, 1859–1875.