Journal of Bioinformatics and Computational Biology

OPTIMAL SPACED SEEDS FOR HOMOLOGOUS CODING REGIONS

BROŇA BREJOVÁ and DANIEL G. BROWN and TOMÁŠ VINAŘ
School of Computer Science, University of Waterloo, 200 University Ave West Waterloo, ON N2L3G1, Canada
{*bbrejova,browndg,tvinar*}*@math.uwaterloo.ca*

Received (1 February 2003)
Revised (22 June 2003)
Accepted (Day Month Year)

Optimal spaced seeds were developed as a method to increase sensitivity of local alignment programs similar to BLASTN. Such seeds have been used before in the program PatternHunter, and have given improved sensitivity and running time relative to BLASTN in genome-genome comparison. We study the problem of computing optimal spaced seeds for detecting homologous coding regions in unannotated genomic sequences. By using well-chosen seeds, we are able to improve the sensitivity of coding sequence alignment over that of TBLASTX, while keeping runtime comparable to BLASTN. We identify good seeds by first giving effective hidden Markov models of conservation in alignments of homologous coding regions. We give an efficient algorithm to compute the optimal spaced seed when conservation patterns are generated by these models. Our results offer the hope of improved gene finding due to fewer missed exons in DNA/DNA comparison, and more effective homology search in general, and may have applications outside of bioinformatics.

Keywords: Spaced seeds, local alignment, gene prediction, homology search, hidden Markov models, BLAST, probabilistic models

1. Introduction

In their program PatternHunter, Ma et al.[1] introduced spaced seeds as a way of improving the sensitivity of homology search programs at minimal cost. In this framework, as in the traditional one typified by BLASTN[2], DNA alignments are based on short matches, called seeds, between the two sequences. With spaced seeds, however, there are specific positions in the seed match that need not be identical. This contrasts with programs such as BLASTN, which is based on eleven consecutive characters found in both sequences.

Here, we extend this idea of spaced seeds to the case of detecting homologous coding regions in unannotated DNA sequences. For distantly related species, these regions form the majority of pairwise alignments. Also, DNA alignments between homologous coding regions can aid in gene finding[3]. Therefore, programs for genome-genome comparison must be able to find such alignments.

We develop a sequence alignment method that substantially improves upon existing algorithms for identifying homologous DNA sequences. In particular, our experiments on homologous human and mouse sequences show that our method has comparable sensitivity to TBLASTX, while requiring forty times less CPU time; alternatively, with different parameter settings, our methods give fewer than half as many false negatives while still running 24 times faster than TBLASTX. Compared to BLASTN, our method gives far fewer false negatives: four times fewer for seeds with comparable runtimes, in the case of human/fruit fly comparisons.

Our method is successful because it is based on spaced seeds that are optimal for probabilistic models that accurately represent alignments of coding regions. By contrast, PatternHunter's seed is the seed most likely to have a match in a homologous region of length 64 where each position is the same with probability 0.7. PatternHunter's seed is optimal in the sense that it has highest probability of all seeds in a given family of matching an alignment drawn from this distribution. Keich et al.[4] have given an algorithm which computes these probabilities for the simple alignment model in PatternHunter.

Since coding region alignments have complicated structure, including three-periodicity and internal dependencies, this simple model of alignments is not appropriate for identifying homologous coding regions. We develop models that take these factors into account, and thus model conservation in alignments much more accurately. Our models are represented by hidden Markov models, where the probability that a position is conserved depends on the state of the model. In Section 4, we extend the algorithm of Keich et al. to the case of hidden Markov models, and show that one can still compute the optimal spaced seed efficiently. It is these seeds which underly our successful method for alignment.

We note that the use of spaced seeds for homology search in coding sequences has implicitly been studied before, in the software package WABA[5]. This program uses the trivial three-periodic seed 110110110... Our method is more sensitive, while having comparable runtime. We also note that our methods were developed contemporaneously with other work by Buhler et al.[6], who extend Keich et al.'s algorithm[4] to the case of multiple seeds used for the same model. They do not, however, consider the case of either HMMs, or of probabilistic models specifically developed for coding regions.

2. Spaced seeds and simple probabilistic models of local alignments

Following ideas introduced in Ma et al.[1], we consider the use of spaced seeds, which generalize the algorithm used in BLASTN to identify homologous regions. BLASTN first finds short exact matches, called hits. A BLASTN hit consists of several consecutive positions (the default is 11). For each, an alignment is built that extends the hit on both sides. If the alignment score exceeds a threshold, the alignment is reported. Some significant alignments do not contain 11 consecutive

matches; thus, they are not discovered by BLASTN.

In a generalization introduced by Ma et al.[1], a hit consists of several non-consecutive matches in a prescribed configuration, called a spaced seed. A seed can be represented as a binary sequence, in which a `1` denotes a position that is required to match and a `0` denotes a position that is not required to match. For example, the seed `110111` requires two consecutive matches followed by one position which may or may not match, and another 3 consecutive matches. The seed corresponding to a BLASTN hit is simply `11111111111`.

Local alignments can be represented as binary sequences, where `1` represents a match and `0` a mismatch. We model only ungapped alignments, as seed hits are found only in gapless regions. We can characterize a particular type of local alignment by giving a probabilistic model defining a probability distribution over all binary strings of a given length.

Then, given a particular probabilistic model, we can find a seed s^* from a class of seeds C maximizing the probability of a hit in an alignment sampled from this model. For us, the class C is all seeds with W ones and length at most M. To avoid redundancy, we require that the first and last positions of a seed are `1`. The parameter W is the *weight* of a seed.

Ma et al.[1] introduced the concept of optimal seeds in PatternHunter. PatternHunter's probabilistic model of alignment $PH(N, p)$ represents a similarity region of length N, where each position is a match independently with probability p. Formally, it is a sequence of N independent Bernoulli random variables $X_0, X_1, \ldots, X_{N-1}$, with $\Pr(X_i = 1) = p$ for each i. PatternHunter's seed `111001001001010111` optimizes this model with parameters $N = 64$ and $p = 0.7$.

Keich et al.[4] give a dynamic programming algorithm to compute the probability that a given seed Q of length M has at least one hit in an alignment sampled from $PH(N, p)$. The running time is $O(2^{M-W}M(M^2 + N))$ for each seed. One computes the probability of a hit for each seed in C and chooses the best.

As we will show in the next section, the simple PH model does not well represent coding sequences, and thus is poor at picking optimal seeds for finding alignments between them. We present richer probabilistic models for coding sequences that take into account their dependencies and complexity.

3. Modeling Local Alignments in Protein Coding Regions

Alignments in coding regions have specific properties not modeled well by the simple PH model. First, the PH model assumes equal mutation rate on all positions in the alignment. However, in protein coding regions, silent mutations accumulate more quickly than mutations that change amino acids. Therefore, third positions in codons of even closely related proteins can undergo substantial mutation, since this is the position in codons most likely to be redundant. There also can be substantial within-codon dependencies in alignment positions. Finally, coding sequence alignments (and, actually, non-coding alignments) often are quite inhomogeneous: some

Table 1. Parameters of model $M^{(3)}$ estimated from our training set.

Data set	p_0	p_1	p_2
human/mouse	0.82	0.87	0.61
human/fruit fly	0.67	0.77	0.40

regions are highly conserved between the two species and others are not.

We address these issues by developing three models, increasing in complexity and accuracy, for alignments of coding regions. Our models can be represented as hidden Markov models (HMMs). HMMs model dependencies between adjacent positions in the sequence and have sucessfully modeled biological sequences in a number of applications[7]. Our work adds to this list by demonstrating that HMMs can capture several important properties of alignments of protein coding regions in DNA sequences.

We also extend the probabilistic model so that the length of the alignment is a random variable, chosen from some fixed distribution of alignment lengths. The probability of a seed match in such an extended model is the weighted average of the probabilities for models with fixed lengths. In our experiments, in Section 5, this distribution is the empirical one from the data in our test set.

3.1. *Three-periodicity*

The most obvious property of alignments in coding regions is their three-periodicity and the fact that some of the positions of a codon are less conserved than others. A simple extension of the *PH* model, which we call $M^{(3)}$, encapsulates this idea. Model $M^{(3)}(N, p_0, p_1, p_2)$ represents a region of length N, where the probability of a site being a match depends on its relative codon position, but positions within the alignment are still independent. Formally, it is a sequence of N independent Bernoulli random variables $X_0, X_1, \ldots, X_{N-1}$ where $\Pr(X_i = 1) = p_{i \bmod 3}$. This model can be expressed as a simple HMM with three states, as depicted in Figure 1, and we can use the algorithm presented in Section 4 to compute the probability that a seed matches an alignment sampled from this model.

Table 1 shows the parameters of the model $M^{(3)}$ estimated from our training set of alignments between human and fruit fly protein coding regions and between human and mouse protein coding regions (more details about the data sets can be found in Section 5). In both cases, the third position is much less conserved than the others, and the first position is somewhat less conserved than the second.

3.2. *Dependencies within codon*

Model $M^{(3)}$ models the different conservation levels within codons arising from the redundancy of the genetic code. However, there are also dependencies among positions within codons. These arise in part from the differing prevalence of some

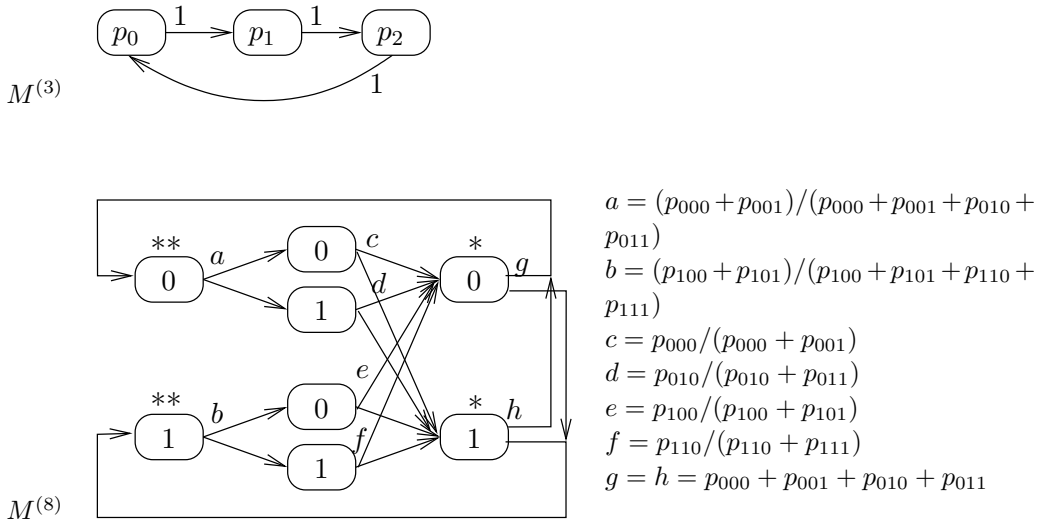

Fig. 1. HMM representation of models $M^{(3)}(N, p_0, p_1, p_2)$ and $M^{(8)}(n, p_{000}, \ldots, p_{111})$. Each state is labeled with emission probability of '1'. In the HMM for $M^{(3)}$, each transition has probability 1. In the HMM for $M^{(8)}$, each state has two outgoing transitions. One of them has the transition probability shown in the picture; the other has probability such that they add to 1.

amino acids and their corresponding codons. To model these dependencies, we use another model, $M^{(8)}$.

The model $M^{(8)}(n, p_{000}, p_{001}, \ldots, p_{111})$ represents a region of n codons ($N = 3n$ nucleotides). Each codon has conservation pattern $x \in \{0,1\}^3$ with probability p_x; the sum of $p_{000}, p_{001}, \ldots, p_{111}$ is 1. In this model, the positions within one codon have arbitrary dependencies specified by the parameters $p_{000}, \ldots, p_{111}$, yet individual codons are independent of each other. Formally, this model is a sequence of n independent triples of Bernoulli random variables $(X_0, X_1, X_2), \ldots, (X_{3n-3}, X_{3n-2}, X_{3n-1})$, such that $\Pr(X_{3i} = a, X_{3i+1} = b, X_{3i+2} = c) = p_{abc}$. Model $M^{(8)}$ can be represented by the HMM shown in Figure 1.

Table 2 demonstrates the existence of dependencies within codon. The first row of the table shows the probabilities of all triplets as estimated by model $M^{(3)}$, under the assumption that codon positions are independent. The second row shows the probabilities of all triplets in model $M^{(8)}$, which exactly match the data set probabilities. In particular, triplets 000 and 111 occur in the data more often than expected by the $M^{(3)}$ model. Hence, we expect that the $M^{(3)}$ model will be inferior to the $M^{(8)}$ model, since it underestimates the probability of a triplet being fully conserved and of the triplet being entirely unconserved.

Table 2. Comparison of probabilities of conservation patterns within a codon in models $M^{(3)}$ and $M^{(8)}$. Parameters of models estimated from human/fruit fly alignments.

Model	p_{000}	p_{001}	p_{010}	p_{011}	p_{100}	p_{101}	p_{110}	p_{111}
$M^{(3)}$	0.05	0.03	0.15	0.10	0.09	0.06	0.31	0.20
$M^{(8)}$	0.11	0.04	0.12	0.06	0.06	0.03	0.32	0.27

3.3. *Inhomogeneity of alignments*

The previous models assume that alignments have roughly the same conservation rate throughout their length. In fact, the pattern of conservation of a typical alignment is highly non-uniform. Many alignments include short, highly conserved regions surrounded by less well conserved regions. (This is unsurprising, as highly conserved regions are more likely to be functional parts of the proteins[8].)

We address this problem with a new HMM. In this model, regions with high and low conservation can alternate. This model is an HMM consisting of four copies of the HMM for $M^{(8)}$ shown in Figure 1, each copy corresponding to a different level of conservation. To allow transitions between different conservation levels, we add transitions from all copies of both states labeled by a star in Figure 1 to all copies of both states labeled by two stars, thus allowing changes in conservation rate after each codon. In our experiments this model will be called *HMM*.

In the next section, we present an algorithm for computing the probability that a seed has a hit in a string sampled from an HMM. Our algorithm is an extension of the algorithm presented by Keich et al.[4] In particular, the *PH* probabilistic model is equivalent to a single state HMM, and in this special case, our algorithm is identical to the one presented by Keich et al.

4. Finding Optimal Seeds for HMMs

Assume that we are given a seed Q with length M and weight W and an HMM that characterizes alignments. We want to compute the probability of seed Q having at least one hit in an alignment of length N. This probability is the *sensitivity* of the seed Q.

Let S be the set of states of the HMM. Each step of the generative process consists of emission of one character in the current state, followed by transition to the next state. Let $\mathit{String}(s, x)$ be the event that the HMM generates string x, assuming that it starts in state s. Let $\mathit{State}(s, t, i)$ be the probability that after i steps the HMM will be in state t, assuming that it starts in state s. Notice that $\Pr(\mathit{State}(s, 0, s)) = 1$ and $\sum_{x\in\{0,1\}^i} \Pr(\mathit{String}(s, x)) = 1$, for all i.

We will proceed by dynamic programming, where the subproblem is defined as follows: for any $i \leq N$, binary string x of length at most $\min\{i, M\}$, and $s \in S$, the subproblem $A_{i,x,s}$ is the probability that the string generated by the HMM in i steps starting in state s contains a match of the seed Q, provided that x is a prefix of the generated string. The probability that Q has at least one hit in the entire

alignment then equals the sum of $A_{N,\lambda,s}$ over all states s, weighted by the initial probabilities of the states (λ denotes the empty string).

Let $matches(x, Q)$ be the event that string $x\mathtt{1}^{M-|x|}$ is a hit for seed Q. That is, when we align the starts of x and Q, x has `1` at all positions where Q has `1`. If $matches(x, Q)$, we say that x is a matching string. Let $\mathit{suffix}(x, Q)$ be the longest suffix z of x such that z is a matching string.

The probability $A_{i,x,s}$ can be computed by the following recurrent formula:

$$A_{i,x,s} = \begin{cases} 0 & \text{if } i < M \quad \text{(A)} \\ 1 & \text{if } |x| = M \text{ and } matches(x, Q) \quad \text{(B)} \\ \sum_{t \in S} p_t \cdot A_{i-|y|,z,t} & \text{if not } matches(x, Q), \text{ where} \\ & \quad x = yz,\ z = \mathit{suffix}(x, Q), \\ & \quad p_t = \Pr[State(s, t, |y|) \mid String(s, x)] \quad \text{(C)} \\ p \cdot A_{i,x\mathtt{1},s} & \\ \quad + (1-p) \cdot A_{i,x\mathtt{0},s} & \text{if } |x| < M \text{ and } matches(x, Q), \text{ where} \\ & \quad p = \Pr[String(s, x1) \mid String(s, x)] \quad \text{(D)} \end{cases}$$

Case (A) recognizes that seeds of length M cannot have hits in shorter regions. In case (B), string x guarantees a hit. In case (C), since the string x does not match Q, a hit will not start at the first position of the alignment, so only following positions need to be considered. In particular, the longest matching suffix z of string x corresponds to the first possible position where a hit can occur. We need to consider all possible states t in which the HMM can start to generate suffix z. Finally, case (D) provides a formula for combining subproblems where the last $|x| + 1$ characters are fixed into a subproblem where only $|x|$ characters are fixed.

This recurrent formula can be used to compute probabilities $A_{i,x,s}$ in order of increasing i and shrinking x. To complete the dynamic programming algorithm we need to demonstrate how to compute the probabilities needed in cases (C) and (D). First, the standard Forward algorithm for HMMs[9] can be used to compute values $B_{s,t,x}$ and $C_{s,x}$ defined as follows:

$$B_{s,t,x} = \Pr(String(s, x) \text{ and } State(s, t, |x|))$$

$$C_{s,x} = \Pr(String(s, x)) = \sum_{t \in S} B_{s,t,x}$$

The conditional probabilities required can now be computed using these two formulas:

$$\Pr(State(s, t, |y|) \mid String(s, \overbrace{yz}^{x})) = \frac{B_{s,t,y} \cdot C_{t,z}}{C_{s,x}} \quad (1)$$

$$\Pr(String(s, x1) \mid String(s, x)) = \frac{\sum_t B_{s,t,x} \cdot C_{t,1}}{C_{s,x}} \quad (2)$$

Keich et al.[4] show how to efficiently organize the computation of a similar recurrent formula so that the running time is $O(M2^{M-W}(M^2+N))$ for each seed (under the assumption that numbers of size between 0 and 2^{M-W} can be manipulated in constant time). Their methods can be applied to our modified algorithm in the following way.

Lemma 1. *The probability that a given seed of length M and weight W has a hit in an alignment of length N generated by a given HMM with σ states can be computed in $O(M2^{M-W}(M^2+\sigma^3+\sigma^2 N))$ time.*

Proof. Let $\Gamma = \{x, x0 \mid x \text{ is a matching string}, |x| \leq M\}$. The number of strings in Γ is $O(M2^{M-W})$.

First observe that to compute values $A_{N,\Lambda,s}$, we do not need to compute values of $A_{i,x,s}$ for all x, but it is sufficient to compute the values for $x \in \Gamma$.

A value of $\mathit{suffix}(x0, Q)$ can be easily computed in $O(M^2)$ time. Then we compute values of $B_{s,t,x}$ for each pair of states s and t and for each $x \in \Gamma$. Using the fact that Γ is closed under prefix operation this can be done in $O(\sigma^3 M 2^{M-W})$ time. It is possible to prove that only values of $B_{s,t,x}$ for $x \in \Gamma$ are needed to compute probabilities from formulas (1) and (2) for all $x \in \Gamma$. In particular, notice that if $x \in \Gamma$ and $x = yz$ where $z = \mathit{suffix}(x, Q)$, then both y and z are also in Γ.

Once all auxiliary tables are computed, each value $A_{i,x,s}$ can be computed in $O(\sigma)$ time. This gives total time $O(M2^{M-W}(M^2+\sigma^3+\sigma^2 N))$. □

In practice, the algorithm proved quite efficient in our experiments. We were able to evaluate the theoretical sensitivity to alignments of length 195 of all seeds with weight 10 and length at most 18 in 3 hours and 58 minutes, on a 2.4 GHz Pentium IV workstation.

5. Experimental Data

5.1. *Data and methods*

We have conducted our experiments on two data sets of protein coding region alignments: human vs. fruit fly and human vs. mouse. The data sets were treated separately. First, we found statistically significant interspecies protein alignments. Then, we filtered these so that each protein occurs in at most one alignment. Next, we split the resulting sets into testing and training halves.

We then mapped the protein sequences to their coding sequences in the genomes. In this way, we transformed alignments of pairs of proteins into alignments of pairs of DNA sequences. Note that one protein alignment can yield several DNA alignments because the coding regions for each protein can be interrupted by non-coding introns (see Figure 2). We call the DNA alignments in this set *fragments*. We discarded weakly conserved and very short fragments because they cannot be detected by alignment programs using the spaced seed method.

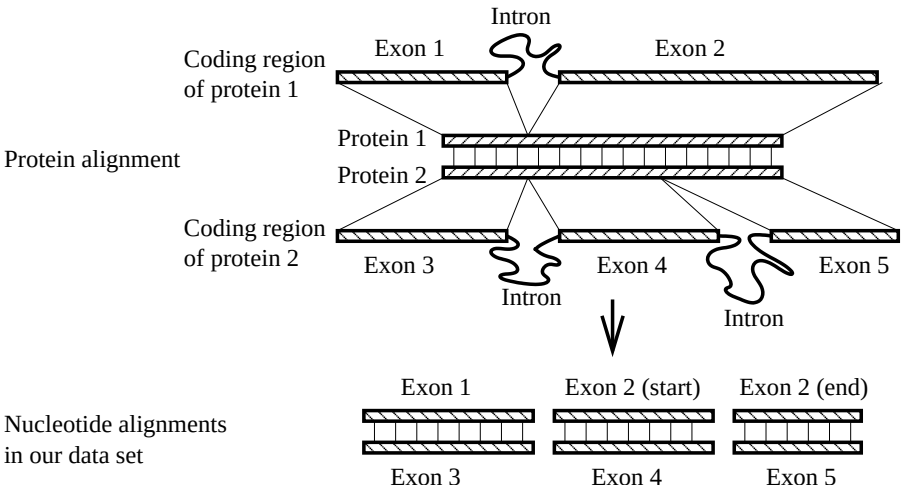

Fig. 2. Several alignments of coding regions corresponding to one protein alignment.

Finally, note that the fragments are gapped alignments. However, to find a fragment by an alignment program using the spaced seed method, the seed hit must be within an ungapped region. Thus, we broke gapped fragments in the training set into *ungapped* fragments, and again discarded weak and short fragments. In the testing set, we have kept gapped fragments, because a single hit is sufficient to discover the entire fragment by the spaced seed method.

By this process, we have ensured that our data sets contain biologically meaningful nucleotide alignments of protein coding regions. Only coding regions of related proteins are aligned, codon boundaries are always correctly aligned, and alignments do not extend to non-coding parts of the genome.

The initial data set consisted of all human, fruit fly (*Drosophila*) and mouse proteins from SWISSPROT database[10], release 40.38, for which a correctly annotated coding regions could be found in the GenBank database. The initial alignments were created by BLASTP 2.0.8[2] using E-value threshold 10^{-30}. The resulting set contained 339 human vs. fruit fly protein alignments and 675 human vs. mouse protein alignments. In the final data sets we filtered out gapped fragments with alignment score less than 16 (scoring scheme: match $+1$, mismatch -1, gap opening -5, gap extension -1) and ungapped fragments with less than 10 matches. We show the sizes and mean lengths of alignments in both data sets in Table 3. The complete data set can be obtained at `http://www.bioinformatics.uwaterloo.ca/supplements/03seeds`.

We used the training set of ungapped fragments to estimate the parameters of PH, $M^{(3)}$, $M^{(8)}$, and HMM. We estimated the parameters of PH, $M^{(3)}$, and $M^{(8)}$ by counting frequencies of corresponding conservation patterns, while we estimated the parameters of the HMM model by the Baum-Welch algorithm[7]. We estimated

Table 3. Parameters of the data sets.

	Training set (ungapped)		Testing set (gapped)	
Data set	$n =$	mean length	$n =$	mean length
Human/fruit fly	972	104	810	138
Human/mouse	2171	120	1660	152

the length distribution of alignments from the set of ungapped fragments as well.

In our experiments we studied all 24310 seeds of weight 10 and length at most 18. We used the dynamic programming algorithm to compute the probability of a hit in a random alignment for each seed under each of the models.

To evaluate the seed performance, we have computed for each seed how many gapped alignments from the testing set contain a hit of the seed and can be thus potentially found by an alignment program using this seed. The results of these experiments are discussed in the following sections.

5.2. *Optimal seeds and their performance*

Here, we present the optimal seeds under different probabilistic models of alignments, and we compare their sensitivity on the testing data set as well as sensitivity when using the seeds with alignment program PatternHunter. The theoretical sensitivity of a seed is the probability that the seed has at least one match to an alignment. In our experiments with real data, the sensitivity of a seed is the fraction of alignments that have a match to the seed.

Table 4 lists the seeds we selected for further examination, and presents the sensitivity of each seed on both testing sets. We also evaluated the seeds with the program PatternHunter and we list the fraction of testing fragments overlapping an alignment reported by PatternHunter.

On the human vs. fruit fly data set, the seed DATA-OPT has the highest sensitivity on the testing set. It was not discovered by any model. Models *HMM* and $M^{(8)}$ both discovered the seed HMM-OPT, which has almost the same performance as DATA-OPT. The seed M3-OPT is optimal under model $M^{(3)}$, but its performance is lower than that of the other seeds. We also include the seeds used by WABA, PatternHunter and BLASTN for comparison. Out of these, only WABA performs comparably with M3-OPT. We also include the seed globally worst on the testing set.

The results are similar on the human vs. mouse data set. Here, models $M^{(8)}$ and $M^{(3)}$ found the seed DATA-OPT, which is again optimal for the testing data. Model *HMM* reported the seed HMM-OPT again. Note that in this case, there is much less difference between the best and the worst seeds, since alignments between human and mouse are much more conserved than alignments between fruit fly and mouse, and are thus easier to detect by any seed. Still, there is a clear separation between performance of the seeds tailored to detecting alignments in coding regions

Table 4. Performace of selected seeds on both testing sets. Columns labeled Hit indicate how many fragments in the testing set have a hit of the seed. Columns labeled PH indicate how many of these fragments overlap an alignment discovered by the PatternHunter program using the seed.

Seed		Fruit Fly		Mouse	
		Hit	**PH**	**Hit**	**PH**
11011011000011011	DATA-OPT	86%	85%	94%	92%
11011000011011011	HMM-OPT	85%	85%	94%	92%
11001011001011011	M3-OPT	82%	76%	93%	90%
11011011011011	WABA	81%	79%	92%	90%
111001001001010111	PH-OPT	59%	57%	87%	86%
1111111111	BLAST	45%	43%	82%	81%
101010101010101011	WORST	38%	39%	80%	79%

and other seeds.

In both data sets, the seeds that take into account the 3-periodic structure of the coding regions (WABA, M3-OPT, HMM-OPT and DATA-OPT), have higher sensitivity than other seeds. Further, the optimal seeds under models *HMM* and $M^{(8)}$ have significantly better performance than the WABA seed, currently used for local alignments in coding regions[5]. By using these seeds we were able to reduce the number false negatives by 29% in the fruit fly set and by 20% in the mouse set.

This increase in sensitivity does not come at a cost of greatly increased running time. The running time of PatternHunter increased by at most 3% compared to the original PH-OPT seed.

5.3. *Our models as predictors of seed performance*

To validate our approach further, we studied how well the sensitivity predicted under each of the models corresponds to the sensitivity measured on the testing data set. Good models should assign higher probability to seeds which perform better in practice, and the probability predicted by the model should correspond to sensitivity on real data.

As illustrated in Figure 3, the predicted probability increases with the sensitivity of the seed on testing data in model $M^{(3)}$, $M^{(8)}$, and *HMM*. This is in contrast to the model *PH* where there is no clear correspondence between predicted and real sensitivity. Models *HMM* and $M^{(8)}$ exhibit better quality of ordering among the top seeds as well as among the worst seeds. In addition to that, *HMM* is clearly the best predictor of the real sensitivity; in fact, the correlation between the estimated sensitivity by *HMM* and the actual sensitivity is the highest, at $r^2 = 0.9687$. Sensitivity is consistently underpredicted in all models, since the training set consisted of ungapped fragments, which are slightly shorter than the gapped fragments in the testing set.

Table 5 further demonstrates the ordering capabilities of each of the models.

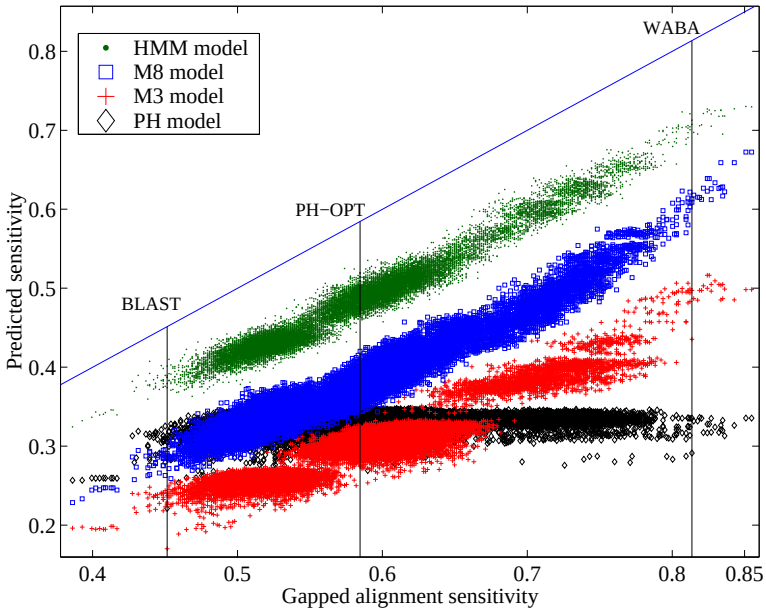

Fig. 3. Sensitivity of all seeds on the testing set versus their predicted sensitivity under different probabilistic models. Each dot represents one seed under one model. Vertical lines indicate sensitivity of the WABA, PH-OPT and BLAST seeds.

Table 5. Ranks of seeds under different probabilistic models. The table shows the rank of each of the chosen seeds in the testing data set as well as under each of the considered probabilistic models.

	Testing data		Rank under a prob. model				
Seed	Rank	Sens.	*HMM*	$M^{(8)}$	$M^{(3)}$	*PH*	Name
11011011000011011	1	0.855	2	2	17	9746	DATA-OPT
11011000011011011	2	0.851	1	1	16	9746	HMM-OPT
11000011011011011	3	0.843	3	3	42	19124	
11011011001011001	4	0.836	15	15	15	23212	
101101100001101101	5	0.835	42	18	37	13208	
11001011001011011	15	0.824	4	6	1	17945	M3-OPT
11011011011011	22	0.814	17	27	129	24187	WABA
111001001001010111	11258	0.585	10427	10350	3254	1	PH-OPT
1111111111	24270	0.451	24285	24233	24310	24310	BLAST
101010101010101011	24310	0.386	24310	24310	24298	24306	WORST

Models *HMM* and $M^{(8)}$ correctly identified the top 3 seeds, while the seed classified as the best in $M^{(3)}$ model ranks 15th on the testing data. Also, there is no clear correspondence between ranks in *PH* model and rank on the testing data. The BLASTN seed (which is currently the most widely used) is among the worst seeds in any of the considered models.

5.4. *Performance relative to TBLASTX, BLASTN and BLAT*

Finally, and perhaps most importantly, we have compared the performance of PatternHunter using our optimal seed HMM-OPT with the performance of TBLASTX, and with the performance of translated BLAT[11], which is a sequence alignment program for genome comparison focusing on coding regions and also allowing mismatches in original seeds. Both TBLASTX and translated BLAT find hits from the sequences translated into all frames, and then extend these hits using an amino acid scoring matrix. On the other hand, PatternHunter with our seed works directly on the genomic sequences.

The results of our experiments comparing the exonic sequences in our data sets from human and mouse and from human and fruit fly are summarized in Table 6.

Here, we see that the false negative rates of TBLASTX and our method are quite close, 6.4% for TBLASTX versus 8.4% for our method. However, the runtimes are not at all close: TBLASTX required fifteen minutes to align these exonic sequences, while our program required 22 seconds. To see if a more sensitive seed might still beat TBLASTX, we computed the optimal seed for model $M^{(8)}$ with eight ones in a seed length of at most eighteen. In Table 6, we call this seed M8-18-8. Using this seed, 11000011011011, we found 97.2% of the BLASTP alignments, with a runtime of 37 seconds. This is still 24 times faster than TBLASTX, yet has fewer than half as many false negatives. While this seed does generate more false positives, as detected in the slower runtime, the runtime is only two times slower than the runtime for the weight 10 seeds.

In experiments on the full genomic regions containing the exons, results were also not comparable: TBLASTX required 7.26 hours (26141 s) to align these regions, while PatternHunter using the HMM-OPT seed required 23 minutes (1394 s). Using the more sensitive weight 8 seed, the alignment still required only two hours (7219 s).

Compared to the BLASTN seed, our methods were also much more successful. With comparable runtimes, we found only 81% of alignments with the BLASTN seed. When we used the comparisons between human and fruit fly, things were worse: there, BLASTN's seed matched only 42.6% of alignments, while HMM-OPT matched 84.7%. This corresponds to a factor of four difference in false negatives.

BLAT also fared substantially less well than our alignment method. At its default seed length setting of five amino acids, BLAT found 80.7% of alignments, about 1.73 times faster than our method, which found 92.5% of them. BLAT did not have a false positive rate comparable to our method until we reduced the seed length to three amino acids; then, it took almost three hundred times as long to do its alignments, while still only finding 90.7% of alignments. However, we note that BLAT also stitches mRNA alignments together, and in general attempts to solve a different problem than just identifying homologous coding alignments as for our system.

Our experiments show that PatternHunter, using our optimal seeds, is substantially more effective than existing sequence alignment packages tailored to align

Table 6. Sensitivity and runtime of various methods for aligning homologous coding sequences. The DATA-OPT, HMM-OPT and M8-18-8 rows correspond to using various spaced seeds with PatternHunter; the other rows correspond to other programs. The M8-18-8 seed is a weight 8 seed that offers sensitivity comparable to or superior to TBLASTX with vastly reduced runtimes.

	Human / mouse		Human / fruit fly	
Seed	Sensitivity	runtime (s)	Sensitivity	runtime (s)
DATA-OPT	0.925	21.8	0.851	13.8
HMM-OPT	0.916	21.9	0.847	14.0
TBLASTX	0.936	891.1	0.947	123.7
BLAT	0.807	12.6	0.581	9.0
M8-18-8	0.972	37.3	0.946	19.9

homologous coding regions, offering greater sensitivity and much lower runtime costs.

6. Conclusion

We have found the optimal spaced seeds for detecting homologous coding regions, whose conservation pattern is characterized by hidden Markov models. These seeds turn out to be substantially more sensitive to alignments than the seeds used by existing sequence alignment methods, while alignments based on them require essentially no additional time. In our experiments, we found that they substantially outperform the naive BLAST search, and the PatternHunter seed, developed for a different model. They also outperform the simple 3-periodic seed used in WABA, matching 20-29% of the alignments missed by that seed in our experiments.

In identifying these optimal seeds, we have given an algorithm to compute the optimal spaced seeds for detecting sequences generated by a hidden Markov model. By extending work of Keich et al.[4], we have given an efficient algorithm for computing the probability that a sequence generated by such a model matches a given seed; then, by examining all such seeds, one can find the best.

Our specific models for conservation patterns in homologous coding sequences incorporate many properties of these regions. We developed HMMs for local alignments that take into account their 3-periodic nature, dependence among positions within triplets, and the inhomogeneity of local coding alignments. The best of these HMMs is much better at predicting the sensitivity of a given seed than more naive models are. In particular, in one experiment, the 3 best seeds according to this model were, in fact, the best overall.

Other questions for future work remain open. First, is it possible to develop similar seed-finding algorithms for other probabilistic models more complicated than HMMs, while still keeping a reasonable runtime? Alternatively, can one approximate the probability of a seed match without the expensive dynamic programming we have described? Also, do other pattern-finding applications of this sort exist, where one searches for patterns in the output of HMMs, and if so, how readily can

these fit within our framework? Finally, can still more accurate models for coding alignments be developed, or can similar techniques be used to study patterns of conservations in sequence elements other than protein coding regions?

Acknowledgement

This work was supported by grants from the Natural Sciences and Engineering Research Council of Canada and by the Human Frontier Science Program. We would also like to thank Bioinformatics Solutions, Inc. (`http://www.BioinformaticsSolutions.com/`) for providing us with a version of PatternHunter customized to our needs and Ming Li for providing access to the manuscript of Keich et al.[4]

References

1. B. Ma, J. Tromp, and M. Li. "PatternHunter: faster and more sensitive homology search". *Bioinformatics* **18**(3), 440–445, March (2002).
2. S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. "Basic local alignment search tool". *J. Mol. Biol.* **215**(3), 403–410 (1990).
3. I. Korf, P. Flicek, D. Duan, and M. R. Brent. "Integrating genomic homology into gene structure prediction". *Bioinformatics* **17 Suppl 1**, S140–8 (2001).
4. U. Keich, M. Li, B. Ma, and J. Tromp. "On spaced seeds". Unpublished.
5. W. J. Kent and A. M. Zahler. "Conservation, regulation, synteny, and introns in a large-scale C. briggsae-C. elegans genomic alignment". *Genome Res.* **10**(8), 1115–1125 (2000).
6. J. Buhler, U. Keich, and Y. Sun. "Designing seeds for similarity search in genomic dna". In *Proceedings of the 7th Annual International Conference on Computational Biology (RECOMB)*, (2003). 67-75.
7. R. Durbin, S. Eddy, A. Krogh, and G. Mitchison. *Biological sequence analysis*. Cambridge University Press, (1998).
8. Z. Yang. "Maximum-likelihood estimation of phylogeny from DNA sequences when substitution rates differ over sites". *Mol. Biol. Evol.* **10**(6), 1396–1401 (1993).
9. L. R. Rabiner. "A tutorial on Hidden Markov models and selected applications in speech recognition". *Proc. IEEE* **77**(2), 257–285 (1989).
10. A. Bairoch and R. Apweiler. "The SWISS-PROT protein sequence database and its supplement TrEMBL in 2000". *Nucleic Acids Res.* **28**(1), 45–48 (2000).
11. W. J. Kent. "BLAT–the BLAST-like alignment tool". *Genome Res.* **12**(4), 656–664 (2002).

Broňa Brejová has completed her undergraduate studies in computer science at Comenius University, Bratislava, Slovakia in 1999. Since then she is a student in Ph.D. program at the School of Computer Science, University of Waterloo, Canada.

Daniel Brown received his undergraduate degree in mathematics with computer science from the Massachusetts Institute of Technology in 1995, and his Ph.D. in computer science from Cornell University in 2000. He then spent a year as a Research Scientist at the Whitehead Institute/MIT Center for Genome Research, in Cambridge, Massachusetts, working on the Human and Mouse Genome Projects. Since 2001, he has been an assistant professor in the School of Computer Science at the University of Waterloo.

Tomáš Vinař recieved his diploma degree in computer science from the Comenius University, Bratislava in 1999. Currently he is a Ph.D. candidate at the School of Computer Science, University of Waterloo.