

# Optimizing multiple spaced seeds for homology search

Jinbo Xu<sup>1</sup>, Daniel G. Brown<sup>1</sup>, Ming Li<sup>1</sup>, and Bin Ma<sup>2</sup>

<sup>1</sup> School of Computer Science, University of Waterloo,  
Waterloo, Ontario N2L 3G1, Canada

{j3xu, browndg, mli}@uwaterloo.ca

<sup>2</sup> Department of Computer Science, University of Western Ontario,  
London, Ontario N6A 5B8, Canada  
bma@csd.uwo.ca

**Abstract.** Optimized spaced seeds improve sensitivity and specificity in local homology search [1]. Several authors [1–4] have shown that multiple seeds can have better sensitivity and specificity than single seeds. We describe a linear programming-based algorithm to optimize a set of seeds. Theoretically, our algorithm offers a performance guarantee: the sensitivity of a chosen seed set is at least 70% of what can be achieved, in most reasonable models of homologous sequences. In practice, our algorithm generates a solution which is at least 90% of the optimal. Our method not only achieves performance better than or comparable to that of a greedy algorithm, but also gives this area a mathematical foundation.

## 1 Introduction

Heuristic homology search programs, such as BLASTN [5], are used extensively in genome analysis. To achieve a fast runtime, BLASTN only builds alignments between regions sharing a common region of  $k$  letters, called a seed. It then verifies which seeds come from good alignments using a local method. The various BLAST programs have contributed much to scientific research since its birth.

BLASTN does not identify its seeds optimally. PatternHunter [1] introduced the concept of optimized spaced seeds and their use in local alignment. This significantly increases sensitivity and computational efficiency. Since this introduction, spaced seeds have been studied by many researchers. Keich *et al.* [6] described a dynamic programming algorithm originally used in [1] to compute the exact hit probability of a seed on a random region under simple distribution. Buhler *et al.*[2] expanded upon this and built a heuristic algorithm to find optimal seeds for alignments when the model of homologous regions is a Markov chain. Choi *et al.* [7] also studied how to calculate the exact hitting probability of a spaced seed and proposed a new algorithm for identifying the optimal spaced seed. Brejova *et al.*[8] considered conserved regions determined by several models such as hidden Markov models (HMMs), particularly for coding

regions. These regions have three-periodic structure, and variation in conservation level throughout the region. Later, they extended the idea of spaced seeds to vector seeds [4], which encapsulate more freedom in describing a single seed.

A single spaced seed cannot achieve the sensitivity of exhaustive, quadratic runtime programs such as the Smith-Waterman algorithm [9]. It is a trivial fact, already pointed out by the original PatternHunter paper [1], that more seeds increase sensitivity. There is a tradeoff between sensitivity and the false positive rate that was described in the PatternHunter II paper [3] and the vector seeds work of Brejova *et al.* [4]. These papers show that the use of multiple seeds or one vector seed can have greater sensitivity than a single seed, even at the same false positive rate. Recently, Sun and Buhler have also confirmed this finding and developed a local search algorithm to find the best set of multiple seeds [10].

Here, we show how to optimize the choice of multiple seeds for homology search. Optimizing a single seed is *NP*-hard [3], and the exponential-runtime algorithms used for choosing the optimized single seed are not appropriate to the multiple-seed case [4, 7]. Instead, we formulate the optimal multiple spaced seeds problem as a *maximum coverage* problem and round the solution to a linear program for that problem. Our algorithm works regardless of the probabilistic model of homologous regions, giving a set of seeds whose sensitivity is guaranteed to be within a factor of  $1 - 1/e \approx .63$  of the best possible. In some cases, we can achieve a much better factor.

In practice, our algorithm returns seed sets whose sensitivity is comparable to the greedily chosen set [3, 4]. The primary advantage of our algorithm is that not only are our seed sets good, we also have a bound on the sensitivity of the best possible set.

A preliminary version of this work was described in our CPM paper [11]. Relative to that paper, the current work includes some new experimental results on two more region models and discusses how to find a seed set for 100% sensitivity. In this paper, we have developed an algorithm to optimize a seed set to achieve 100% sensitivity for a given set of regions. Compared to the greedy algorithm in Li *et al.*'s paper [3], our algorithm can find a smaller set of seeds, which can speed up the seed-based homology search.

The rest of the paper is organized as follows. Section 2 defines seeds, alignment regions and the optimization problem to be solved. In Section 3, we formulate the optimization problem as a maximum coverage problem and use a linear programming based randomized algorithm to solve it. In this section, we also prove the approximation ratio of our randomized algorithm. Section 4 describes the performance of our algorithm on both the probabilistic region models and the real DNA sequence alignments. In this section, we also compare the per-

formance of our algorithm with that of the greedy algorithm. In Section 5, we present an integer program formulation to find a seed set for 100% sensitivity. We conclude this paper with a short discussion in Section 6.

## 2 Problem Formulation

### 2.1 Alignments and seeds

We model ungapped pairwise nucleotide alignments, which we call homologous regions, by binary strings, where 1 represents a match, and 0 a mismatch. The length of a region  $R$  is  $|R|$ , and  $R[i]$  is the  $i$ th bit of  $R$ . We model these regions probabilistically, using a random distribution on binary strings. A random region model specifies the distributions of its length and its position-by-position distributions as well.

Here, we consider the following four region models, introduced in Ma *et al.* [1] and Brejova *et al.* [8]:

1. *PH* model [1]. In this model, each bit of the region is set to one independently with probability 0.7, and the length of the region is a constant 64 bits.
2. *M3* model [8]. This is a codon-aware model for the coding region. The conservation level of each position depends on its relative position in its codon. It is discovered that the first two positions of a codon are more restricted and the third one is more varied. In this model, the conservation probabilities of the first two positions and the third position are 0.8 and 0.5, respectively.
3. *M8* model [8]. This is also a codon-aware model for coding regions. Any two codons are independent, but the three positions in a codon are not independent. There are in total eight possible settings for a codon. The probability of each setting is shown in Table 1.
4. *HMM* model [8]. This model represents coding regions. Here, each three bits of the region are produced as triplets, and the conservation level in one triplet depends on the previous triplet’s conservation level. This models regions of high and low conservation. The mean region length is approximately 90.

**Table 1.** Probability of eight codon settings in *M8* model

000	001	010	011	100	101	110	111
0.11	0.04	0.12	0.06	0.06	0.03	0.32	0.27

With models for alignments, we turn to seeds. A spaced seed  $S$  is a binary string. Its length is  $|S|$ , while its weight is its number of ones. A 1 in the seed  $S$  corresponds to a position with a required match and a 0 means “don’t care.”

**Definition 1.** A hit to a seed  $S$  in a region  $R$  is an offset  $i$  such that  $R[i+k] \geq S[k]$  for all  $1 \leq k \leq |S|$ .  $R$  is hit by the seed  $S$  if there is a hit to  $S$  in  $R$ . Given a set of seeds,  $A = \{S_1, S_2, \dots, S_k\}$ ,  $R$  is hit by  $A$  when  $R$  is hit by at least one seed in  $A$ .

## 2.2 Optimization problems

Given a probabilistic model  $P$  for regions and two integers  $W$  and  $M$ , the *optimal single spaced seed problem* is to find the spaced seed  $S^*$  with weight  $W$  and length no more than  $M$  that maximizes the probability of a hit to a region chosen from  $P$ . Similarly, for  $k > 1$ , the *optimal  $k$ -seeds problem* is to find a set  $A^*$  of  $k$  seeds, each with weight  $W$  and length no more than  $M$ , that maximizes the hit probability. These problems have been proven to be *NP*-hard by Li *et al.* [3].

We can also choose a seed from a collection of seeds. Let  $A = \{S_1, S_2, \dots, S_m\}$  be a collection of spaced seeds, and suppose we have a probabilistic model  $P$  for homologous regions. The *seed specific optimal  $k$ -seeds problem* is to find the  $k$ -element subset  $Q^*$  of  $A$  that maximizes the probability of a hit to  $P$ .

We will analyze this problem by generating many regions from the distribution of  $P$ , and then choosing a collection of seeds with high sensitivity to these instances of  $P$ . We will guarantee that our set of regions is representative by choosing enough regions. Given a collection of  $N$  such regions,  $R = \{R_1, \dots, R_N\}$ , and a set  $A$  of possible seeds, the *seed-region specific optimal  $k$ -seeds problem* is to find a  $k$ -element subset  $Q^*$  of  $A$  that hits the most regions in  $R$ .

A reduction from the maximum coverage problem shows that the seed-region specific optimal  $k$ -seeds problem is itself *NP*-hard [3]. The following sections describe how to formulate the seed-region specific optimal seeds problem as a maximum  $k$ -coverage problem and solve it by an LP based randomized algorithm.

## 2.3 Bounds on number of instances

Following the PatternHunter II paper [3], we solve the seed-specific optimal  $k$ -seeds problem by first generating a large set of random regions, and then choosing a good seed set for them. Chernoff bounds tell us how large a set we need so the hit probability for the regions is close to the model hit probability. In particular, the following is due to Chernoff bounds:

**Theorem 1.** *Let  $A$  be a set of seeds,  $P$  a model for random regions, and  $R_1, \dots, R_N$  independent regions from distribution  $P$ . Assume that  $A$  has probability  $p$  of hitting regions from  $P$ , and that the fraction of the  $N$  seeds that are hit by  $A$  is  $p'$ . For any  $0 < \delta < 1$ , the following bounds on  $p'$  and  $p$  hold:*

1.  $Pr[p' - p > \delta] \leq e^{-2N\delta^2}$ ,
2.  $Pr[p' - p < -\delta] \leq e^{-2N\delta^2}$ .

Hence, if we choose large enough  $N$ , we can approximate the desired probability,  $p$ , with the sample probability,  $p'$ . Once we choose the random regions, we cast the problem as a maximum  $k$ -coverage problem. Suppose we have a collection  $R = \{R_1, \dots, R_N\}$  of regions, and a set  $S = \{S_1, \dots, S_m\}$  of seeds. Let  $H_i$  be all regions that seed  $S_i$  hits. Our selection problem is to find the set of  $k$  sets  $H_i$  whose union is as large as possible; this is the maximum coverage problem. The obvious greedy algorithm approximates this to a  $1 - 1/e$  fraction [12, 3], and Feige has shown that this bound cannot be improved for the general maximum coverage problem [13, 3].

### 3 Choosing seed sets by linear programming

Our approach to optimizing the maximum coverage is through linear programming and randomized rounding. This approach improves upon the performance of the greedy algorithm in our special case of maximum coverage. The integer program is straightforward: binary decision variables  $x_i$  are 1 when seed  $S_i$  is selected, and  $y_j$  are 1 if region  $R_j$  is hit. Our problem is then modeled this way:

$$\max \frac{1}{N} \sum_{j=1}^N y_j \tag{1}$$

subject to:

$$y_j \leq \sum_{i:j \in H_i} x_i, \text{ for } j = 1, \dots, N, \tag{2}$$

$$\sum_{i=1}^m x_i = k \tag{3}$$

$$x_i \in \{0, 1\}, \text{ for } i = 1, \dots, m, \tag{4}$$

$$y_j \in \{0, 1\}, \text{ for } j = 1, \dots, N. \tag{5}$$

Constraint 2 guarantees that a region  $R_j$  is hit only if at least one seed that hits it is chosen, while constraint 3 permits only  $k$  seeds to be chosen.

### 3.1 A simple LP-rounding bound

No polynomial-time algorithm is known to exist for integer programs. Instead, we relax the integrality constraints on  $x_i$  and  $y_j$  and solve the resulting linear program. Then, we use a random rounding technique to pick a set of seeds. Assume the optimal linear solution to the LP is  $(\mathbf{x}^*, \mathbf{y}^*)$ . We create a probability distribution on the  $m$  seeds that consists of the scaled  $x^*$  values,  $x_i^*/k$ , and choose  $k$  indices,  $s_1, \dots, s_k$ , from this probability distribution. We may choose the same index multiple times. Our rounded solution to the LP,  $(x^+, y^+)$  has  $x_i^+ = 1$  if we chose index  $i$ , and  $y_j^+ = 1$  if we chose one of the seeds that hits region  $R_j$ . The set  $\mathcal{S} = \{S_i\}$  of seeds whose indices we have chosen form our set of seeds. (This set may be smaller than  $k$ .)

The performance of this algorithm is as good as the greedy algorithm.

**Theorem 2.** *The LP-rounding algorithm generates a solution with expected approximation ratio at least  $1 - 1/e$  for the seed-region specific optimal multiple seeds problem, and can be derandomized to ensure this performance deterministically.*

To prove this theorem, we first note that the probability that a particular seed  $S_i$  is *not* chosen is simple to estimate:

$$\Pr[x_i^+ = 0] = \left(1 - \frac{x_i^*}{k}\right)^k \leq e^{-x_i^*}.$$

Now, let us consider a single region and estimate its probability of being missed.

$$\Pr[y_j^+ = 0] = \Pr\left[\bigcap_{k:j \in H_k} x_k^+ = 0\right],$$

since a region is missed if no seed that hits it was chosen. If one seed is not chosen, other seeds are more likely to be chosen (since one seed is chosen in each round), so

$$\Pr\left[\bigcap_{k:j \in H_k} x_k^+ = 0\right] \leq \prod_{k:j \in H_k} \Pr[x_k^+ = 0] \leq \prod_{k:j \in H_k} e^{-x_k^*}.$$

Moving sums to products, we see that  $\Pr[y_j^+ = 0] \leq \exp(-\sum_{k:j \in H_k} x_k^*)$ . Since  $y_j^* \leq \sum_{i:j \in H_i} x_i^*$ , this probability is at most  $\exp(-y_j^*)$ , which for  $y$  between 0 and 1 is bounded above by  $1 - (1 - 1/e)y_j^*$ . Thus, the probability that  $y_j^+ = 1$  is at least  $(1 - 1/e)y_j^*$ . The expected fraction of the regions that are hit by the chosen set of seeds is at least a  $(1 - 1/e)$  fraction of the optimal objective function value for the linear program. Since the LP bound is an upper bound on the optimal integer programming solution, the expected approximation ratio of the algorithm is at least  $1 - 1/e$ . This algorithm can also be derandomized by standard techniques [14, 15].

### 3.2 A better bound for the rounding algorithm

In some cases, we can prove a better performance bound than  $1 - 1/e$ . If we have lower and upper bounds on the number of regions hit by every possible seed, the seed set chosen by the randomized LP algorithm may have provable expected performance better than what can be proved in general. For example, in the *PH* model, any seed with weight 11 and length no more than 18 hits at least 30% of the sample regions if the number of samples is big enough.

In this direction, we can prove the following theorem.

**Theorem 3.** *If each seed in  $A$  hits at least  $\theta N$  regions and the optimal linear solution of the LP has objective function value  $l^*$  ( $0 \leq l^* \leq 1$ ), then the rounding algorithm has expected approximation ratio at least  $\frac{\theta k - l^*}{(k-1)l^*}(1 - e^{-k}) + (1 - \frac{\theta k - l^*}{(k-1)l^*})(1 - e^{-1})$  for the seed-region specific optimal multiple seeds problem.*

*Proof.* Summing up the right hand side of (2), for any feasible solution of the linear program, we see that  $\sum_{j=1}^N \sum_{i:j \in H_i} x_i \geq \theta(\sum_{i=1}^m x_i) = \theta k N$ .

Let  $N_1$  be the number of  $y$  variables with  $y^*$  less than one, and  $N_2$  the number of  $y$  variables with  $y^*$  equal to one. Let  $l^*$  denote the objective function value of the optimal linear solution. Let  $Y_1^*$  be the sum of the  $y_j^*$  that are less than one, and  $X_1^*$  be the sum of the corresponding right hand sides of (2). These are equal, since we raise  $Y_1^*$  as much as possible. Similarly, let  $Y_2^*$  be the sum of the  $y_j^*$  which equal one (Obviously,  $Y_2^* = N_2$ ), and let  $X_2^*$  the sum of the corresponding right hand sides of (2).

Then we have the following equations:

$$\begin{aligned} X_1^* + X_2^* &\geq \theta k N \\ X_2^* &\leq k N_2 \\ Y_1^* + Y_2^* &= N l^* \\ N_1 &\geq X_1^* \end{aligned} \tag{6}$$

Based on the above equations, simple calculations show:

$$\begin{aligned} \frac{X_2^* - N_2}{N l^*} &= \frac{X_2^* - Y_2^*}{N l^*} \\ &\geq \frac{\theta k N - Y_1^* - Y_2^*}{N l^*} \\ &\geq \frac{\theta k - l^*}{l^*} \end{aligned} \tag{7}$$

According to the discussion in Subsection 3.1, we have the following two equations:

$$\Pr[y_j^+ = 1] \geq 1 - \exp\left(-\sum_{k:j_k} x_k^*\right) \quad (8)$$

$$\Pr[y_j^+ = 1] \geq (1 - 1/e)y_j^* \quad (9)$$

Now we finish the proof of Theorem 3.

Based on Equation 9, we have

$$\frac{\sum_{j:y_j^* < 1} E[y_j^+]}{\sum_{j:y_j^* < 1} y_j^*} \geq 1 - e^{-1} \quad (10)$$

We need to deal with the case that  $y_j^*$  equals to 1. Based on Equation 8, we have

$$\begin{aligned} \frac{\sum_{j:y_j^*=1} E[y_j^+]}{\sum_{j:y_j^*=1} y_j^*} &\geq \frac{1}{N_2} \sum_{j:y_j^*=1} (1 - e^{-\sum_{i:j \in H_i} x_i}) \\ &\geq \frac{1}{N_2} \left( \frac{X_2^* - N_2}{k-1} (1 - e^{-k}) + (N_2 - \frac{X_2^* - N_2}{k-1}) (1 - e^{-1}) \right) \\ &\geq \frac{X_2^*/N_2 - 1}{k-1} (1 - e^{-k}) + \left(1 - \frac{X_2^*/N_2 - 1}{k-1}\right) (1 - e^{-1}) \end{aligned} \quad (12)$$

where (11) comes from Lemma 2 in the appendix.

Based on (10), (12), and Lemma 1 in the appendix, we have

$$\begin{aligned} \frac{\sum_{j=1}^N E[y_j^+]}{\sum_{j=1}^N y_j^*} &= \frac{\sum_{j:y_j^*=1} E[y_j^+] + \sum_{j:y_j^* < 1} E[y_j^+]}{\sum_{j:y_j^*=1} y_j^* + \sum_{j:y_j^* < 1} y_j^*} \\ &\geq \left( \frac{\frac{X_2^*}{N_2} - 1}{k-1} (1 - e^{-k}) + \left(1 - \frac{\frac{X_2^*}{N_2} - 1}{k-1}\right) (1 - e^{-1}) \right) \frac{Y_2^*}{Nl^*} + (1 - e^{-1}) \left(1 - \frac{Y_2^*}{Nl^*}\right) \\ &\geq \frac{X_2^* - N_2}{(k-1)Nl^*} (1 - e^{-k}) + \left(1 - \frac{X_2^* - N_2}{(k-1)Nl^*}\right) (1 - e^{-1}) \\ &\geq \frac{\theta k - l^*}{(k-1)l^*} (1 - e^{-k}) + \left(1 - \frac{\theta k - l^*}{(k-1)l^*}\right) (1 - e^{-1}) \end{aligned} \quad (13)$$

where (13) comes from (7).

This completes the proof of this theorem.

Table 2 give the provable approximation ratio for the region models, for all seeds with length at most 18 and weight 11. In the *PH* model,  $\theta \approx 0.30$ , while

for the *HMM* model,  $\theta \approx 0.73$ . In addition, in computing these two tables, we replace  $l^*$  with 1.

In both cases, the approximation ratio is better than  $1 - 1/e$  and improves as  $k$  grows. However, if the seed weight is larger than 11, then Theorem 3 cannot prove an approximation ratio larger than  $1 - 1/e$ . But, if the seed weight is 10, or each region of the *PH* model is longer than 64, the theoretical approximation ratio will be better.

**Table 2.** Approximation ratios for region models and seeds of length at most 18 and weight 11.

number of seeds	2	4	6	8	10	12	14	16
ratio for <i>PH</i> model	0.632	0.655	0.691	0.706	0.714	0.719	0.723	0.725
ratio for <i>M3</i> model	0.632	0.646	0.678	0.693	0.702	0.707	0.711	0.714
ratio for <i>M8</i> model	0.632	0.666	0.700	0.715	0.723	0.728	0.731	0.734
ratio for <i>HMM</i> model	0.739	0.856	0.879	0.886	0.890	0.892	0.893	0.894

## 4 Experimental Results

Here, we present experiments with our seed search algorithm. We show that our algorithm finds high quality seed sets, and we compare our algorithm and the simple greedy algorithm, documenting that both algorithms choose good seeds.

### 4.1 Probabilistic model results

We used our algorithms to find good seed sets for our four random region models, *PH*, *M3*, *M8* and *HMM*. We generated 5000 instances of each model,  $R_M = \{R_1 \dots R_N\}$ . For any seed set  $A$ , let  $p_R(A)$  be the fraction of these 5000 instances that were hit by the set.

If our algorithm finds a seed set  $\hat{A}$ , we can use the algorithms described by Keich *et al.* [6] or Brejova *et al.*[8] to compute the fraction  $p_M(\hat{A})$  of regions from the model  $M$  hit by the seed. As an upper bound on  $p_M$ , we use either 1 or a bound based on the LP solution. Let  $A^*$  be the optimal set of seeds for the model, and let  $I^*$  and  $l^*$  be the IP and LP objective function optima, respectively. We know that  $p_R(A^*) \leq I^* \leq l^*$ , since the LP bound is an upper bound on the IP bound, and  $A^*$  may not perform better than the IP solution on  $R$ . Theorem 1 shows that with probability at least 0.99,  $p_M(A^*) \leq p_R(A^*) + 0.0215$ . Thus, an upper bound on the best possible seed performance, with probability 0.99, is  $l^* + 0.0215$ . To estimate the optimization ratio of our algorithm, we compare  $p_M(\hat{A})$  to  $l^* + 0.0215$ .

**Table 3.** Estimated approximation ratio of the seed sets generated by our algorithm with PH model and HMM model.

# of seeds	PH model		HMM model		M3 model		M8 model	
	$W = 10,$ $M \leq 18$	$W = 11,$ $M \leq 18$	$W = 10,$ $M \leq 18$	$W = 11,$ $M \leq 18$	$W = 10,$ $M \leq 18$	$W = 11,$ $M \leq 18$	$W = 10,$ $M \leq 18$	$W = 11,$ $M \leq 18$
2	0.916	0.899	0.970	0.962	0.954	0.935	0.911	0.887
4	0.934	0.912	0.961	0.960	0.948	0.935	0.905	0.887
6	0.939	0.919	0.959	0.960	0.952	0.941	0.906	0.897
8	0.943	0.931	0.953	0.956	0.954	0.939	0.910	0.898
10	0.948	0.931	0.954	0.953	0.955	0.942	0.907	0.891
12	0.950	0.936	0.953	0.950	0.957	0.945	0.905	0.891
14	0.951	0.937	0.953	0.950	0.958	0.946	0.913	0.887
16	0.952	0.938	0.951	0.948	0.957	0.945	0.919	0.887

Table 3 shows the estimated approximation ratio of the seed sets generated by our algorithm for seeds of weight 10 or 11 and length at most 18. Our seed sets are at least 90% as good as the best possible (with probability 0.99). Solving one linear programming instance takes seven hours to terminate on a single 500MHZ CPU, which we note is affordable, given that one only computes these seed sets once.

## 4.2 Seed sets for ESTs

Here, we tested the sensitivity of our seed sets against real alignments generated by comparing human and mouse EST sequences. Out of a collection of 29715 mouse EST sequences and 4407 human EST sequences, we randomly chose three million pairs to test for sensitivity. We computed the optimal alignments between these sequences using the Smith-Waterman algorithm (with matches scoring +1, mismatches and gap extensions -1, and gap openings -5), and tested to see how many of them were hit by a seed set. We identified true positives as being hits to alignments scoring at least 16, while false positives were hits to alignments with lower score. Please refer to Li *et al.* [3] for the detailed description of this procedure. In measuring false positives, we consider two indices. One is the number of sequence pairs that are hit by our seed sets and the other is the number of hit positions. At each hit position, we need to check if this hit can be extended to a valid alignment. Therefore, the number of hit positions is more related to the running time of homology search. In examining whether or not one sequence pair (say A and B) is hit by a seed, we consider not only the alignment between A and B, but also that between A and the reverse complement of B.

We used the seed sets chosen in the previous experiment to test how well they did in this practical test, and Table 4 and 5 show the results. Most notably,

**Table 4.** sensitivity and false positive values of chosen seed sets for PH model and HMM model. TP: hit rate of the pairs with alignment score at least 16. FP1: the ratio between the number of hit pairs and the number of all the pairs. FP2: the ratio between the number of hits and the number of all the pairs.

#seeds	PH model						HMM model					
	$W = 11, M \leq 18$			$W = 10, M \leq 18$			$W = 11, M \leq 18$			$W = 10, M \leq 18$		
	TP	FP1	FP2	TP	FP1	FP2	TP	FP1	FP2	TP	FP1	FP2
1	0.341	0.167	0.177	0.661	0.485	0.727	0.363	0.171	0.188	0.647	0.490	0.753
2	0.528	0.310	0.367	0.835	0.677	1.439	0.539	0.317	0.386	0.869	0.703	1.652
3	0.648	0.411	0.550	0.907	0.771	2.143	0.675	0.419	0.596	0.932	0.793	2.496
4	0.715	0.483	0.717	0.948	0.825	2.877	0.751	0.506	0.821	0.958	0.836	3.265
5	0.763	0.535	0.888	0.964	0.854	3.477	0.801	0.568	1.034	0.973	0.865	3.959
6	0.810	0.595	1.094	0.973	0.874	4.057	0.840	0.619	1.237	0.983	0.888	4.788
7	0.838	0.639	1.251	0.986	0.895	5.103	0.859	0.651	1.408	0.989	0.900	5.631
8	0.883	0.678	1.514	0.988	0.903	5.435	0.882	0.687	1.654	0.991	0.909	6.241
9	0.888	0.700	1.638	0.993	0.916	6.544	0.902	0.712	1.804	0.993	0.917	6.798
10	0.898	0.718	1.761	0.994	0.922	6.986	0.914	0.734	1.949	0.995	0.924	7.744
11	0.919	0.747	2.014	0.995	0.926	7.481	0.930	0.758	2.239	0.996	0.930	8.447
12	0.931	0.763	2.185	0.996	0.931	8.186	0.938	0.771	2.363	0.997	0.934	8.923
13	0.937	0.777	2.369	0.996	0.935	8.679	0.941	0.781	2.480	0.997	0.937	9.450
14	0.947	0.792	2.540	0.997	0.940	9.574	0.952	0.799	2.751	0.998	0.942	10.397
15	0.952	0.804	2.709	0.998	0.943	10.447	0.955	0.808	2.951	0.998	0.943	10.941
16	0.954	0.810	2.794	0.998	0.944	10.765	0.959	0.818	3.016	0.998	0.946	11.862

the seed sets all quickly converge to high sensitivity. It is worth noting that four well-chosen seeds of weight 11 will often have better sensitivity and specificity than one seed of weight 10. Figure 1 further compares the performance of four seeds with weight 11 chosen to optimize the *HMM* model and a single seed with weight 10, which have similar false positive rates. The set of four weight 11 seeds is much more sensitive on the pairs with an alignment score between 16 and 40 than the single *HMM* seed with weight 10. For a pair with score at least 40, any seed can easily detect it. Our findings show that using seeds with weight 11 can detect more subtly similar pairs than using seeds with weight 10 at the same false positive level.

### 4.3 Curated coding alignments

We further tested our algorithm on two protein coding DNA alignment data sets, one of human/mouse alignments, and one of human/fruit fly alignments. Each data set is treated separately. Please refer to Brejova *et al.* [8] for a detailed description of these data. In the fly/human set, 972 alignments are used for training and 810 for testing, while in the mouse/human set, 2171 alignments are used to train and 1660 to test. These are available at <http://monod.uwaterloo.ca/supplements/03seeds>. We used the

**Table 5.** sensitivity and false positive values of chosen seed sets for M3 model and M8 model. TP: hit rate of the pairs with alignment score at least 16. FP1: the ratio between the number of hit pairs and the number of all the pairs. FP2: the ratio between the number of hits and the number of all the pairs.

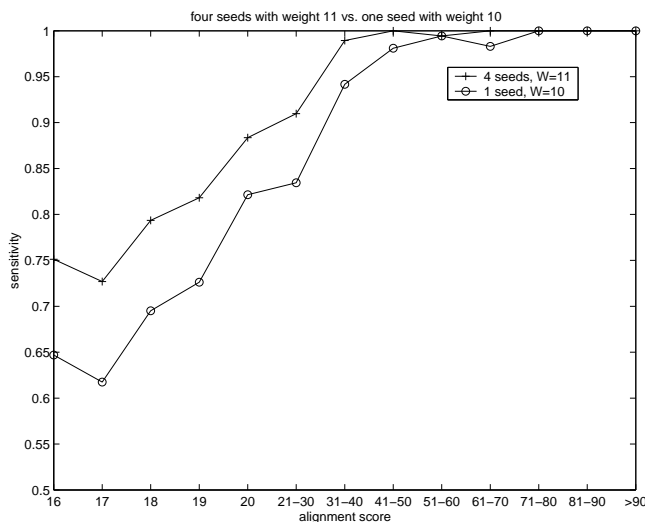
#seeds	M3 model						M8 model					
	$W = 11, M \leq 18$			$W = 10, M \leq 18$			$W = 11, M \leq 18$			$W = 10, M \leq 18$		
	TP	FP1	FP2	TP	FP1	FP2	TP	FP1	FP2	TP	FP1	FP2
1	0.363	0.171	0.188	0.630	0.477	0.695	0.388	0.193	0.214	0.647	0.489	0.752
2	0.506	0.291	0.356	0.801	0.655	1.334	0.556	0.321	0.404	0.825	0.676	1.455
3	0.616	0.389	0.532	0.883	0.756	1.989	0.662	0.424	0.610	0.928	0.792	2.470
4	0.695	0.476	0.711	0.936	0.817	2.750	0.751	0.505	0.818	0.956	0.836	3.182
5	0.744	0.534	0.870	0.967	0.857	3.620	0.794	0.559	0.997	0.969	0.861	3.810
6	0.800	0.595	1.082	0.974	0.874	4.125	0.831	0.614	1.211	0.981	0.884	4.619
7	0.834	0.628	1.249	0.981	0.890	4.861	0.857	0.649	1.365	0.987	0.899	5.377
8	0.847	0.655	1.383	0.985	0.902	5.413	0.877	0.681	1.551	0.990	0.908	6.097
9	0.867	0.686	1.524	0.989	0.911	6.002	0.896	0.710	1.747	0.993	0.919	6.977
10	0.898	0.717	1.799	0.992	0.919	6.712	0.912	0.734	1.970	0.994	0.923	7.542
11	0.903	0.731	1.871	0.994	0.927	7.585	0.925	0.753	2.162	0.995	0.929	8.095
12	0.907	0.743	1.986	0.995	0.931	8.217	0.936	0.771	2.319	0.996	0.935	8.895
13	0.922	0.762	2.182	0.996	0.935	8.902	0.945	0.787	2.545	0.997	0.938	9.792
14	0.939	0.780	2.462	0.996	0.937	8.936	0.946	0.792	2.564	0.997	0.941	9.821
15	0.945	0.796	2.593	0.997	0.941	9.917	0.950	0.801	2.787	0.998	0.943	10.619
16	0.943	0.797	2.632	0.997	0.943	10.304	0.958	0.814	2.982	0.998	0.946	11.341

training samples to search for the optimal seed sets with our LP algorithm (note that no probabilistic model is used here), and then tested our seed sets on the test samples.

Table 6 shows the sensitivity of the seed sets with weight 10 and weight 11. As shown in these two figures, four seeds with weight 11 are 2% more sensitive than a single seed with weight 10. However, there is no difference between the sensitivity of  $k$  ( $k = 2, 3, 4$ ) seeds with weight 10 and  $4k$  seeds with weight 11, which is because the alignments themselves are fairly high-scoring. A single optimal seed with weight 10 hits 86% of the human/fruit alignments and 91% of human/mouse alignments.

#### 4.4 Comparison with Greedy Algorithm

We also implemented the greedy algorithms described in Li *et al.* [3] and Brejova *et al.* [4] and compared it with our LP-based randomized algorithm. Table 7 compares the sensitivity of the seed sets found by two algorithms on the two curated sets of coding alignments; the results are comparable. The advantage is that the LP based algorithm enables us to estimate a usually much better approximation ratio of the solutions, while the greedy algorithm only gives a very poor  $(1 - 1/e)$  approximation ratio.



**Fig. 1.** Sensitivity comparison of four seeds with weight 11 chosen for the *HMM* model and a single seed with weight 10.

### 5 Achieving 100% Sensitivity: An Application

We have discussed how to optimize a fixed number of spaced seeds to achieve the best sensitivity. In one application, we were required to run PatternHunter at 100% sensitivity by using multiple spaced seeds, with the following dataset and assumptions:

- The object is a large genome database.
- The queries are millions of relatively short nucleotide sequences of length 21.
- 75% homology level is assumed. That is, there are 16 matches out of the 21 positions.
- No gap in the alignments.
- The seeds have weight 9 and length no more than 15.

Smith-Waterman (SSearch) requires about 10 CPU-years for this task. Blast obviously does not work here unless it uses a ridiculously short seed and runs extremely slowly. Then the problem becomes finding the smallest number of seeds so that the sensitivity of these seeds on the region model is 100%. Using the Greedy algorithm discussed in Li *et al.*'s paper [3], this problem was solved with 40 seeds [16]. PatternHunter needed 8 days to complete the task with these 40 seeds. Using the techniques developed in this paper, we can use an integer program to formulate this problem as follows.

**Table 6.** Sensitivity of the seeds on coding region alignment data

# seeds	fruit fly vs. human		mouse vs. human	
	$\bar{W} = 10, M \leq 18$	$\bar{W} = 11, M \leq 18$	$\bar{W} = 10, M \leq 18$	$\bar{W} = 11, M \leq 18$
1	0.863	0.781	0.919	0.872
2	0.920	0.831	0.952	0.910
3	0.944	0.868	0.961	0.930
4	0.956	0.885	0.975	0.938
5	0.956	0.910	0.980	0.948
6	0.964	0.912	0.981	0.948
7	0.974	0.917	0.984	0.961
8	0.973	0.920	0.989	0.968
9	0.981	0.928	0.989	0.972
10	0.981	0.933	0.988	0.970
11	0.983	0.937	0.989	0.973
12	0.980	0.940	0.990	0.970
13	0.985	0.947	0.990	0.981
14	0.985	0.948	0.993	0.977
15	0.989	0.949	0.993	0.980
16	0.986	0.957	0.997	0.978

$$\min \sum_{i=1}^m x_i \quad (14)$$

subject to:

$$1 \leq \sum_{i:j \in H_i} x_i, \text{ for } j = 1, \dots, N, \quad (15)$$

$$x_i \in \{0, 1\}, \text{ for } i = 1, \dots, m \quad (16)$$

where, constraint 15 guarantees that a region  $R_j$  is hit by at least one seed. The above algorithm generated only 35 seeds, which improves computational efficiency over the greedy solution by approximately 13%.

## 6 Discussion

The experimental result with real data demonstrates that using more seeds with a bigger weight is better than using fewer seeds with a smaller weight in detecting the subtly similar sequence pairs. However, using more seeds with a bigger weight requires much more memory to index the sequence database.

As we know, for a general maximum  $k$ -coverage problem, the greedy algorithm and the LP-based algorithm have the same approximation ratio,  $(1 - e^{-1})$ . An interesting question is if the greedy algorithm can also guarantee a better

**Table 7.** Performance comparison of seeds generated by the greedy algorithm and the LP based randomized algorithm

#seeds	fruit fly vs. human				mouse vs. human			
	$W = 10, M \leq 18$		$W = 11, M \leq 18$		$W = 10, M \leq 18$		$W = 11, M \leq 18$	
	Greedy	LP	Greedy	LP	Greedy	LP	Greedy	LP
1	0.863	0.863	0.781	0.781	0.919	0.919	0.872	0.872
2	0.920	0.920	0.831	0.831	0.952	0.952	0.902	0.910
3	0.944	0.944	0.860	0.868	0.970	0.961	0.922	0.930
4	0.952	0.956	0.886	0.885	0.977	0.975	0.941	0.938
6	0.963	0.964	0.911	0.912	0.984	0.981	0.957	0.948
8	0.970	0.973	0.925	0.920	0.991	0.989	0.964	0.968
10	0.978	0.981	0.938	0.933	0.993	0.988	0.970	0.970
12	0.985	0.980	0.944	0.940	0.994	0.990	0.973	0.970
14	0.989	0.985	0.948	0.948	0.995	0.993	0.977	0.977
16	0.989	0.986	0.951	0.957	0.996	0.997	0.981	0.978

theoretical bound for the optimal multiple seeds problem, given that the experimental results described in this paper indicate that the greedy algorithm has comparable performance to our LP-based algorithm.

## 7 Acknowledgements

The authors would like to thank Brona Brejova for her help and insightful comments. Our research is supported by the National Science and Engineering Research Council of Canada, CITO's Champion of Innovation Program, the Killam Fellowship, Canada Research Chair Program, and the Human Frontier Science Program.

## References

1. B. Ma, J. Tromp, and M. Li. Patternhunter: Faster and more sensitive homology search. *Bioinformatics*, 18:440–445, 2002.
2. J. Buhler, U. Keich, and Y. Sun. Designing seeds for similarity search in genomic DNA. In *Proceedings of the Seventh Annual International Conference on Computational Molecular Biology (RECOMB03)*, pages 67–75, Berlin, Germany, April 2003.
3. M. Li, B. Ma, D. Kisman, and J. Tromp. Patternhunter II: Highly sensitive and fast homology search. *Journal of Bioinformatics and Computational Biology*, 2004. In Press.
4. B. Brejova, D. Brown, and T. Vinar. Vector seeds: an extension to spaced seeds allows substantial improvements in sensitivity and specificity. In G. Benson and R. Page, editors, *Algorithms and Bioinformatics: 3rd International Workshop (WABI)*, volume 2812 of *Lecture Notes in Bioinformatics*, pages 39–54, Budapest, Hungary, September 2003. Springer.
5. S.F. Altschul, W. Gish, W. Miller, E.W. Myers, and D.J. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215:403–410, 1990.
6. U. Keich, M. Li, B. Ma, and J. Tromp. On spaced seeds for similarity search. *Discrete Applied Mathematics*, 2004. In Press.

7. K.P. Choi and L. Zhang. Sensitivity analysis and efficient method for identifying optimal spaced seeds. *Journal of Computer and System Sciences*, 68:22–40, 2004.
8. B. Brejova, D. Brown, and T. Vinar. Optimal Spaced Seeds for Hidden Markov Models, with Application to Homologous Coding Regions. In R. Baeza-Yates, E. Chavez, and M. Crochemore, editors, *Combinatorial Pattern Matching, 14th Annual Symposium (CPM)*, volume 2676 of *Lecture Notes in Computer Science*, pages 42–54, Morelia, Mexico, June 25–27 2003. Springer.
9. T.F. Smith and M.S. Waterman. Identification common molecular subsequences. *Journal of Molecular Biology*, 147:195–197, 1981.
10. Y. Sun and J. Buhler. Designing multiple simultaneous seeds for DNA similarity search. In *Proceedings of the Eighth Annual International Conference on Computational Molecular Biology (RECOMB04)*, pages 76–84, California, USA, March 2004.
11. J. Xu, D. Brown, M. Li, and B. Ma. Optimizing multiple spaced seeds for homology search. In *Combinatorial Pattern Matching, 14th Annual Symposium (CPM)*, Lecture Notes in Computer Science, Istanbul, Turkey, July 2004.
12. D.S. Hochbaum. Approximating covering and packing problems: set cover, vertex cover, independent set, and related problems. In Dorit S. Hochbaum, editor, *Approximation Algorithms for NP-hard Problems*, chapter 3, pages 135–137. PWS, 1997.
13. U. Feige. A threshold of  $\ln n$  for approximating set cover. *Journal of the ACM*, 45(4):634–652, 1998.
14. R. Motwani and P. Raghavan. *Randomized Algorithm*. Cambridge University Press, New York, 1995.
15. P. Raghavan. Probabilistic construction of deterministic algorithms: Approximating packing integer programs. *Journal of Computer and System Sciences*, 37:130–143, October 1988.
16. <http://www.bioinformaticssolutions.com/products/phsuccess.php>.

## 8 Appendix

**Lemma 1.** Given  $a_1, a_2, b_1, b_2 > 0$ . If  $\frac{a_1}{a_2} \geq \beta_1$ ,  $\frac{b_1}{b_2} \geq \beta_2$ ,  $\frac{a_2}{a_2+b_2} \geq \alpha > 0$ , and  $\beta_1 \geq \beta_2 > 0$ , then  $\frac{a_1+b_1}{a_2+b_2} \geq \alpha\beta_1 + (1-\alpha)\beta_2$ .

This lemma is a simple number fact, and trivial to verify.

**Lemma 2.** Given  $z_1, z_2, \dots, z_n, b \geq z_i \geq 1, b > 1, \sum_{i=1}^n z_i = Z$ . Then we have  $\sum_{i=1}^n (1 - e^{-z_i}) \geq \frac{Z-n}{b-1}(1 - e^{-b}) + (n - \frac{Z-n}{b-1})(1 - e^{-1})$ .

*Proof.* If there exist  $z_i, z_j, (1 \leq i < j \leq n)$  such that  $1 < z_i, z_j < b$ . Without loss of generality, assume  $z_i \leq z_j$ . For any  $0 < \epsilon \leq \min(z_i - 1, b - z_j)$ , we have

$$1 - e^{-(z_i-\epsilon)} + 1 - e^{-(z_j+\epsilon)} \leq 1 - e^{-z_i} + 1 - e^{-z_j}$$

since

$$(e^\epsilon - 1)(e^{-z_i} - e^{-z_j-\epsilon}) \geq 0$$

This indicates that if there are two  $z_i, z_j$  which are not equal to 1 or  $b$ , then we can always adjust their values to decrease the value of  $\sum_{i=1}^n (1 - e^{-z_i})$  while keep  $\sum_{i=1}^n z_i$  unchanged.

Therefore,  $\sum_{i=1}^n (1 - e^{-z_i})$  becomes minimum if and only if at most one  $z_i$  is not equal to 1 or  $b$ . If all  $z_i$  equal to 1 or  $b$ , then the number of  $z_i$  equal to  $b$  is at least  $\frac{Z-n}{b-1}$ . The lemma holds. If there is one  $k$  ( $1 \leq k \leq n$ ) such that  $1 < z_k < b$ , then by simple calculation we have

$$\begin{aligned} \sum_{i=1}^n (1 - e^{-z_i}) &\geq \frac{Z-n}{b-1}(1 - e^{-b}) + (n - \frac{Z-n}{b-1})(1 - e^{-1}) + \frac{z_k-1}{b-1}(e^{-b} - e^{-1}) + e^{-1} - e^{-z_k} \\ &> \frac{Z-n}{b-1}(1 - e^{-b}) + (n - \frac{Z-n}{b-1})(1 - e^{-1}). \end{aligned}$$