Proc. ISMB '00, pp. 260--268, AAAI Press, Menlo Park, CA (2000)

Spectrum Alignment: Efficient Resequencing by Hybridization

I. Pe'er and **R. Shamir**
{izik,shamir}@math.tau.ac.il
Department of Computer Science,
Tel Aviv University, Tel Aviv, 69978 ISRAEL

Abstract

Recent high-density microarray technologies allow, in principle, the determination of all k-mers that appear along a DNA sequence, for $k = 8 - 10$ in a single experiment on a standard chip. The k-mer contents, also called the *spectrum* of the sequence, is not sufficient to uniquely reconstruct a sequence longer than a few hundred bases. We have devised a polynomial algorithm that reconstructs the sequence, given the spectrum and a homologous sequence. This situation occurs, for example, in the identification of single nucleotide polymorphisms (SNPs), and whenever a homologue of the target sequence is known. The algorithm is robust, can handle errors in the spectrum and assumes no knowledge of the k-mer multiplicities. Our simulations show that with realistic levels of SNPs, the algorithm correctly reconstructs a target sequence of length up to 2000 nucleotides when a polymorphic sequence is known. The technique is generalized to handle profiles and HMMs as input instead of a single homologous sequence.

Keywords: Sequencing by Hybridization, Mutation Detection, SNP genotyping, Hidden Markov Models, DNA Microarrays.

Introduction

In this paper we propose a new method to reconstruct a DNA target sequence. The method combines spectrum data (obtainable from a universal DNA chip), a known homologous reference sequence, and novel computational techniques, similar to those used for analysis of sequence homology. This introduction gives brief background on each of the three ingredients, and explains how they are used together to achieve the new method.

Sequencing by Hybridization

Sequencing by Hybridization (SBH) was proposed and patented in the late eighties as an alternative to gel-based sequencing (Bains & Smith 1988; Lysov *et al.* 1988; Southern 1988; Drmanac & Crkvenjakov 1987; Macevics 1989). This method makes use of a *DNA chip*, or *microarray* (cf. (Southern 1996)) which includes all oligonucleotides of length k (called k-*mers*). These oligonucleotides are hybridized to an unknown DNA fragment, whose sequence we would like to read. Under ideal conditions, this target molecule would hybridize to all k-mers whose Watson-Crick complementary sequences occur somewhere along its sequence. Thus, in principle, one could experimentally determine the set of all k-long substrings of the target, and try to infer the sequence from the hybridization data. Practical values of k are 8 to 10.

The fundamental computational problem in SBH is the reconstruction of a sequence from its *spectrum* - the set (or, in some models, multi-set) of all k-mers occurring along the sequence. Pevzner 1989 reduced the reconstruction problem to the polynomial task of finding an Eulerian path in a particular directed graph.

The main weakness of SBH is ambiguous solutions. Alternative solutions are manifested as bifurcations in the graph, and unless the number of such forks is very small, there is no good way to determine the true sequence. Theoretical analysis and simulations (Southern, Maskos, & Elder 1992; Pevzner & Lipshutz 1994) have shown that even when the spectrum is errorless and contains the correct multiplicity of each k-mer, the average length of a uniquely reconstructible sequence using an 8-mer chip is only about two hundred base pairs, far below a single read length on a commercial gel-lane machine.

While an effective and competitive sequencing solution using SBH has yet to be demonstrated, this mathematically elegant strategy continues to attract attention. It is still believed that SBH holds promise to considerably economize on the task of sequencing, one of the major efforts in modern biotechnology. Alternative chip designs (Bains & Smith 1988; Khrapko *et al.* 1989; Pevzner *et al.* 1991; Preparata, Frieze, & Upfal 1999; Ben-Dor *et al.* 1999; Preparata & Upfal 2000) as well as interactive protocols (Skiena & Sundaram 1995) were suggested, often assuming additional information sources, in order to resolve ambiguity of the hybridization-based reconstruction.

Proc. ISMB '00, pp. 260--268, AAAI Press, Menlo Park, CA (2000)

Similar Sequences are Ubiquitous

The understanding that many DNA sequences resemble each other is fundamental to biology. This similarity is due to the process of evolution: numerous contemporary sequences have evolved by mutations from a single ancestral molecule. Such related (*homologous*) sequences exhibit similarity to their unique ancestor, and thus to each other. This general scenario is routinely encountered in genome analysis:

- The genomic sequences of all individual organisms of the same species are almost identical. Current estimates of the variability rate among all humans, for instance, amount to one *Single Nucleotide Polymorphism (SNP)* per $100-300$ base pairs (National Center for Biotechnology Information 2000). Of course, when comparing DNA of a specific individual to the genomic reference, these two sequences would share the same allele in many of the polymorphic sites. Hence, the fraction of their base pairs on which they differ would be considerably smaller.
- Much of the eukaryotic genome is composed of *repetitive elements* - sequences which recur in thousands or millions of copies. Different repeats are usually $90-95\%$ identical.
- Various duplications of large genomic segments occurred during the course of evolution. In some cases, the whole genome was duplicated. This process created many homology relationships. In particular, duplications which include coding regions gave rise to several gene copies. At a later stage, these genes diverged and mutated, forming a gene family.
- The genomes of different, phylogenetically related species have homologous segments. The rate of sequence identity may approach 100% identity when comparing highly conserved genomic regions of closely related species, but may also drop to the twilight zone of near-random expected resemblance, for more divergent segments, and more distant taxa.

As sequence data accumulates in an accelerated rate, an increasing number of sequencing targets have a known homologous sequence. This motivates the development of new sequencing strategies which utilize homology information. Hybridization has been successfully used for sequencing SNPs given the reference genomic sequence, using custom made microarrays (Cargill *et al.* 1999; Hacia *et al.* ; Hacia 1999). To the best of our knowledge, this study is the first proposal to use standard, universal chips and homology information for sequencing.

Exploiting Homology Computationally

Sequence similarity is perhaps the most studied issue in bio-informatics (cf. (Durbin *et al.* 1998)). The evolution of homologous sequences from a common ancestral origin is mainly due to nucleotide substitution: a stochastic process which can be described by a nucleotide substitution matrix (Jukes & Cantor 1969; Kimura 1980). This description facilitates calculating the likelihood that one sequences is homologous to another.

Insertions and deletions of nucleotides also occur during evolution of homologous sequences, though at lower rates. This calls for *aligning* the two sequences, i.e., matching pairs of their loci according to the common origin of the paired nucleotides. The well known Smith-Waterman dynamic programming algorithm (Smith & Waterman 1981) computes the alignment score with affine gap penalties. Such a score can be formulated as the log-likelihood of the data using Hidden Markov Models (HMMs) (Durbin *et al.* 1998, chapter 4). The latter are often explicitly used to generalize the homology concept, and to model alignment against a family of sequences (Krogh *et al.* 1994; Eddy 1996).

Our Contribution

We describe here a new method for reconstructing the target sequence, by combining information on a reference sequence with experimental spectrum data obtainable from a standard chip. We call the technique resequencing by hybridization, or *spectrum alignment* since the algorithm attempts to find the best "alignment" of the reference sequence with the spectrum. Technically, this is done by developing a dynamic programming algorithm which runs on the de-Bruijn graph and the reference sequence simultaneously. The algorithm is polynomial, and can handle inexact, probabilistic information on the spectrum, which is common in hybridization results. Unlike other algorithms proposed for SBH and its extensions, it does not assume knowledge of the multiplicities of the k-mers in the sequence. We also show how to extend our results to handle profiles and HMMs as homology information, instead of a particular reference sequence,

The paper is organized as follows: We first give background on SBH and on homology, and necessary terminology is presented. We then provide an algorithm for resequencing by hybridization, allowing mismatches but no insertions or deletions in the target with respect the reference sequence. We then extend the methods to deal with the general case allowing insertions, deletions and substitutions . Finally, we present preliminary results on simulated data.

Preliminaries

Scoring by the Hybridization Data

Let $\Sigma = \{A, C, G, T\}$ be our alphabet, with $M = 4$ being the alphabet size. We denote sequences by a string over Σ between angle brackets ($\langle\rangle$). A *k-spectrum* of a sequence $\mathcal{T} = \langle t_1 t_2 \cdots t_L \rangle$ is the set of all k-long substrings (k-mers) of $\mathcal{T}$. For each k-mer $\vec{x} = \langle x_1 x_2 \cdots x_k \rangle \in \Sigma^k$, we define $\mathcal{T}(\vec{x})$ to be 1 if $\vec{x}$ is a substring of $\mathcal{T}$, and 0 otherwise. We denote $K = M^k$.

A hybridization experiment measures, for each k-mer $\vec{x} \in \Sigma^k$, the intensity of ots hybridization signal. For

Proc. ISMB '00, pp. 260–268, AAAI Press, Menlo Park, CA (2000).

our puspose, the relevant information in such a signal is the probabilities $P_0(\vec{x}), P_1(\vec{x})$ of this observed intensity assuming $\mathcal{T}(\vec{x}) = 0$, and $\mathcal{T}(\vec{x}) = 1$, respectively. We therefore define a *probabilistic spectrum* (PS) to be a pair (P_0, P_1) of functions $P_i : \Sigma^k \mapsto [0,1]$. PS is assumed to be the result of the hybridization experiment, which we analyse. If the experiment were *perfect*, i.e., if the probabilities were all zero or one (with $P_0(\vec{x}) + P_1(\vec{x}) \equiv 1$), then the hybridization data would be translated into a k-spectrum. In such a case we could perfectly determine the occurrence of a k-mer in the sequence by examining the hybridization signal. In practice, though, both $P_0(\vec{x}), P_1(\vec{x})$ are positive, and any deterministic binarization of the hybridization signal will contain errors. Our algorithms will therefore use the probabilistic data. We assume knowledge of these error parameters even prior to sequence reconstruction.

Many suggested combinatorial models for SBH assume, for theoretical convenience, that the multiplicities of k-mer occurrences are known (Pevzner 1989; Preparata, Frieze, & Upfal 1999). This assumption is impractical using current technology and our algorithm does not rely on it (In fact, when all multiplicities are 1 our performance improves.)

The *de-Bruijn graph* is a directed graph $G_k(V, E)$ whose vertices are labeled by all the $(k-1)$-mers $V = \Sigma^{k-1}$, and its edges are labeled by k-mers, $E = \Sigma^k$. The edge labeled $\langle x_1 x_2 \cdots x_k \rangle$ connects the vertex $\langle x_1 x_2 \cdots x_{k-1} \rangle$ to the vertex $\langle x_2 \cdots x_k \rangle$. Whenever k is clear from context we omit it, referring to the De-Bruijn graph as G. There is a 1-1 correspondence between candidate L-long target sequences and $(L - k + 1)$-long paths in G, whose edge labels comprise the target spectrum. In case the spectrum data-set is perfect and the multiplicities are known, omitting all zero probability edges from G one gets one gets Pevzner's formulation, i.e., every solution is an Eulerian path (Pevzner 1989). For our more general formulation we devise a scoring scheme for paths, and search for the best scoring path in G. Hereafter, we interchangeably refer to edges and their labels, and also to sequences and their corresponding paths. Observe that since k-mers may re-occur, paths do not have to be simple.

We assume that hybridization results of different oligos are mutually independent (see discussion). Hence, the *experimental likelihood* $L^e(\hat{\mathcal{T}})$ of a candidate target sequence $\hat{\mathcal{T}}$ is

$$L^e(\hat{\mathcal{T}}) = Prob(\mathrm{PS}|\hat{\mathcal{T}}) = \prod_{\vec{x} \in \Sigma^k} P_{\hat{\mathcal{T}}(\vec{x})}(\vec{x}) \qquad (1)$$

Taking (base 2) logarithm, define $w(\vec{x}) = \log \frac{P_1(\vec{x})}{P_0(\vec{x})}$. Throughout, when handling probabilities, some of which are perfect, problems of division by zero might occur. We get around those by implicitly perturbing perfect probabilities to δ and $1 - \delta$. We can thus write:

$$\log L^e(\hat{\mathcal{T}}) = \sum_{\vec{x} \in \Sigma^k} \log P_0(\vec{x}) + \sum_{\hat{\mathcal{T}}(\vec{x})=1} w(\vec{x}) \qquad (2)$$

The first term is a constant, independent of $\hat{\mathcal{T}}$, and is omitted hereafter.

Let $p = e_0, \ldots, e_{L-k}$ be the path in G corresponding to $\hat{\mathcal{T}}$. Then

$$\log \widetilde{L^e}(\hat{\mathcal{T}}) = \sum_{i=0}^{L-k} w(e_i) \qquad (3)$$

is an approximate likelihood score, deviating from the true likelihood whenever an edge is revisited along p. $\widetilde{L^e}(\hat{\mathcal{T}})$ has the advantage of being easily computable in a recursive manner:

$$\log \widetilde{L^e}(e_0, \ldots, e_l) = \log \widetilde{L^e}(e_0, \ldots, e_{l-1}) + w(e_l) \qquad (4)$$

Pevzner assumes perfect hybridization data (Pevzner 1989). In this case, every path in G whose likelihood score equals one is a possible solution to the SBH problem, while all other paths have probability zero. Indeed, Pevzner simply discards improbable k-mers from G. One can handle imperfect data in an analogous manner, by discarding edges with probability smaller than ϵ. After this procedure, isolated vertices correspond to highly improbable $(k-1)$-mers, and can be discarded as well. We denote the resulting graph $[G]_\epsilon = ([V]_\epsilon, [E]_\epsilon)$, and call its size, $[K]_\epsilon$, the *effective size* of the dataset. Observe that $|[V]_\epsilon| = O([K]_\epsilon), [E]_\epsilon = O([K]_\epsilon)$. Of course, $[K]_0 = K$, but for $\epsilon > 0$, usually $[K]_\epsilon \ll K$, so working with $[G]_\epsilon$ considerably reduces complexity. We omit the ϵ subscript in the sequel.

Scoring by the Homology Information

In this section we show how to use homology information in order to obtain a prior distribution on the space of candidate target sequences. Assume that the unknown target sequence $\mathcal{T} = \langle t_1 \cdots t_l \rangle$ has a known, homologous reference $\mathcal{H} = \langle h_1 \cdots h_l \rangle$, without insertions or deletions (*indels*). This is the case, for instance, when the target $\mathcal{T}$ is a specimen from a population whose reference wild type $\mathcal{H}$ has already been sequenced, and one expects that SNPs will be the only cause of difference between $\mathcal{H}$ and $\mathcal{T}$ (statistically, SNPs are much more prevalent than indels (Wang *et al.* 1998)). We assume a set of $M \times M$ position specific substitution matrices $\mathcal{M}^{(1)}, \ldots, \mathcal{M}^{(l)}$ are known, where for each position j along the sequence:

$$\mathcal{M}^{(j)}[i, i'] = Prob(t_j = i | h_j = i') \qquad (5)$$

This is a very general setting. Standard literature discussing nucleotide substitution matrices (Jukes & Cantor 1969; Kimura 1980) assumes all substitution matrices to be the same, i.e., $\mathcal{M}^{(j)} = \mathcal{M}$ for all j. More recent studies support difference between sites for DNA (Yang 1993) and protein (Eddy 1996) sequences.

The setting just presented implies a distribution on the space of possible target sequences. This *prior distribution for ungapped homology*, D^u, is explicitly given

Proc. ISMB '00, pp. 260--268, AAAI Press, Menlo Park, CA (2000).

for each candidate target sequence $\hat{\mathcal{T}}$ by:

$$D^u(\hat{\mathcal{T}}) = Prob(\hat{\mathcal{T}}|\mathcal{H}) = \prod_{j=1}^{l} \mathcal{M}^{(j)}[t_j, h_j] \qquad (6)$$

One may recursively compute:

$$D^u(\langle t_1 \cdots t_j \rangle) = D^u(\langle t_1 \cdots t_{j-1} \rangle) \cdot \mathcal{M}^{(j)}[t_j, h_j] \qquad (7)$$

We denote $\mathcal{L}^{(j)}[x, y] \equiv \log \mathcal{M}^{(j)}[x, y]$.

Spectrum Alignment

In this section we show how to combine our two sources of information on the target sequence, i.e., the result, PS , of the hybridization experiment, and the reference sequence $\mathcal{H}$. We formalize a Bayesian score, which is a composition of the scores discussed in the previous sections, and present a fast dynamic programming algorithm to compute this score.

The probability of a candidate solution sequence $\hat{\mathcal{T}}$, given the information we have is:

$$Prob(\hat{\mathcal{T}}|\mathcal{H}, \mathrm{PS}) = \frac{Prob(\mathcal{H}) \cdot Prob(\hat{\mathcal{T}}|\mathcal{H}) \cdot Prob(\mathrm{PS}|\mathcal{H}, \hat{\mathcal{T}})}{Prob(\mathcal{H}, \mathrm{PS})} \qquad (8)$$

Given $\hat{\mathcal{T}}$, the hybridization signal is independent of $\mathcal{H}$:

$$Prob(\mathrm{PS}|\mathcal{H}, \hat{\mathcal{T}}) = Prob(\mathrm{PS}|\hat{\mathcal{T}})$$

Thus, omitting the constant $\frac{Prob(\mathcal{H})}{Prob(\mathcal{H}, \mathrm{PS})}$ we can write:

$$Prob(\hat{\mathcal{T}}|\mathcal{H}, \mathrm{PS}) \cong D^u(\hat{\mathcal{T}}) \cdot L^e(\hat{\mathcal{T}}) \qquad (9)$$

We shall use the approximated likelihood, $\widetilde{L^e}(\hat{\mathcal{T}})$, and after taking logarithms we obtain the following *ungapped score* of a candidate target:

$$Score^u(\hat{\mathcal{T}}) = \log \widetilde{L^e}(\hat{\mathcal{T}}) + \log D^u(\hat{\mathcal{T}}) \qquad (10)$$

We can compute the highest scoring target sequence by dynamic programming. For each vertex $\vec{y} = \langle y_1 \cdots y_{k-1} \rangle \in \Sigma^{k-1}$, and integer $j = k-1, k, k+1, \ldots, l$, let $S^u[\vec{y}, j]$ be the maximum score of a j-long sequence ending with $\vec{y}$ aligned to $\langle h_1 \cdots h_j \rangle$. Initialize, for each $\vec{y}$:

$$S^u[\vec{y}, k-1] = \sum_{j=1}^{k-1} \mathcal{L}^{(j)}[y_j, h_j] \qquad (11)$$

Loop over $j = k, \ldots, l$, and for each vertex $\vec{y} = \langle y_1 \cdots y_{k-1} \rangle$ recursively update:

$$S^u[\vec{y}, j] = L^{(j)}[y_{k-1}, h_j] + \max_{e=(\vec{z},\vec{y}) \in E} \{S^u[\vec{z}, j-1] + w(e)\} \qquad (12)$$

Finally, return:

$$MAXScore^u = \max_{\vec{y} \in V} S^u[\vec{y}, l] \qquad (13)$$

As in the Smith-Waterman algorithm (Smith & Waterman 1981), a sequence $\mathcal{T}^*$ attaining the optimal score can be reconstructed by standard means from the matrix S^u, saving trace-back pointers to follow the optimally scoring path. The time complexity is $O(lK)$, since the maximization in (12) is over a set of constant size 4. Note that although the complexity is exponential in k, it is constant for a given microarray (currently feasible values are $k = 8, 9$). Working with $[G]$, with high probability the correct solution will not be missed, but the time complexity will drop sharply to $O(l[K])$.

A crucial issue for the practicality of this algorithm is its space requirement. Computing the optimal score alone requires space which is linear in the (effective) size of the hybridization experimental data, that is $O([K])$ space. However, in order to reconstruct the optimal path, we need to record trace back pointers for the full $l \times [K]$ matrix. By following the paradigm of Hirschberg 1975 for linear-space pairwise alignment, we provide an algorithm which requires only linear space. The reduced space complexity is traded for time complexity, which increases by an $O(\log l)$ factor.

For each position $j = l, l-1, \ldots, k, k-1$, we can decompose the score of the entire sequence. We present the total score as a sum of two expressions: the contribution of its $(j-k+1)$-prefix, which equals the score of this prefix computed by S^u, plus the contribution of the corresponding suffix. Formally, for each vertex $\vec{y} = \langle y_1 \cdots y_{k-1} \rangle \in [V]$ let $R^u[\vec{y}, j]$ be the maximum contribution to the score of a $(l-j+k-1)$-long sequence beginning with $\vec{y}$ aligned to $\langle h_{j-k+2} \cdots h_l \rangle$. Initialize, for each $\vec{y}$:

$$R^u[\vec{y}, l] = 0 \qquad (14)$$

Loop over $j = l-1, l-2, \ldots, k-1$, and for each vertex $\vec{y} = \langle y_1 \cdots y_{k-1} \rangle$ recursively update:

$$R^u[\vec{y}, j] = \max_{e=(\vec{y},\vec{z}) \in E} \{R^u[\vec{z}, j+1] + w(e) + \mathcal{L}^{(j+1)}[z_{k-1}, h_{j+1}]\} \qquad (15)$$

Observe that, for all $k-1 \le j \le l$

$$MAXScore^u = \max_{\vec{y} \in V} \{S^u[\vec{y}, j] + R^u[\vec{y}, j]\} \qquad (16)$$

One can use Equation 16 to decompose the problem into two similar problems, of half its size. Recursively solving these sub-problems gives a a divide-and-conquer approach for finding the optimal sequence. The linear space algorithm is therefore as follows:

Proc. ISMB '00, pp. 260--268, AAAI Press, Menlo Park, CA (2000)

1. If l is smaller than some constant C: solve the problem directly, according to the dynamic program of Equation 7. Otherwise:
2. Set $m = \frac{l+k-1}{2}$.
3. For each $j = k-1, k, \ldots, m$: Compute $S^u[\vec{y}, j]$ for all $\vec{y}$, re-using space.
4. For each $j = l, l-1, \ldots, m$: Compute $R^u[\vec{y}, j]$ for all $\vec{y}$, re-using space.
5. Find $\vec{y}_m = arg\max_{\vec{y} \in V} \{S^u[\vec{y}, m] + R^u[\vec{y}, m]\}$, thereby computing: $MAXScore^u$, by (16).
6. Recursively compute:
   (a) The optimal sequence aligned to $\langle h_1 \cdots h_m \rangle$ ending with $\vec{y}_m$.
   (b) The optimal sequence aligned to $\langle h_m \cdots h_l \rangle$ beginning with $\vec{y}_m$.

Observe, that for each $\vec{y}, j$, the values of $S^u[\vec{y}, j]$ and $R^u[\vec{y}, j]$ are computed a total of $\log l$ times. Thus the algorithm takes $O([K] l \log l)$ time and $O([K])$ space using the effective spectrum.

Handling Gaps

Deletions

In this section we assume that the unknown target sequence $\mathcal{T} = \langle t_1 \cdots t_{l'} \rangle$ is obtained from its reference $\mathcal{H} = \langle h_1 \cdots h_l \rangle$, by substitutions and deletions only, and base insertions do not occur. Insertions in the target are, of course, equivalent to deletions in the reference and vice versa, but since the reference is known we consider all sequence editing operations (mutations) to have occurred in the target sequence.

Although a model of homology without insertions is unrealistic, we include discussion of this case due to the simplicity and efficiency in which deletions fit into our model. The general case, allowing insertions, will be described in the next subsection.

We begin with a few notations. Denote the probability of initiating a gap right before h_j (aligning h_j to *space*) is 2^{α_j}. Similarly, β_j is the logarithm of the probability for gap extension at h_j. Also define $\widehat{\beta_j} = \log(1 - 2^{\beta_j})$, $\widehat{\alpha_j} = \log(1 - 2^{\alpha_j})$. To overcome boundary problems at the ends of the sequence, we extend the alphabet by including left and right space characters: $\overline{\Sigma} = \Sigma \cup \{\triangleright, \triangleleft\}$. We augment the reference sequence by the string $\triangleright^k$ on its left and $\triangleleft^k$ on the right. We extend the substitution matrix by using probabilities that force alignment of each of $\triangleright$, $\triangleleft$ to itself. Formally, we define:

$$\overline{\Sigma^{k-1}} = \Sigma^{k-1} \cup \{\vec{x}\vec{z} | \vec{x} = \triangleright^j, \vec{z} \in \Sigma^{k-1-j}\} \cup \{\vec{z}\vec{x} | \vec{z} \in \Sigma^j, \vec{x} = \triangleleft^{k-1-j}\} \quad (17)$$

We arbitrarily set $w(\vec{y})$ to 0 for each $\vec{y} \in \overline{\Sigma^{k-1}} \setminus \Sigma^{k-1}$. Thus, the weighted de-Bruijn graph is naturally extended over $\overline{\Sigma^{k-1}}$, and so is $[G] = ([V], [E])$, its effective subgraph. Hereafter, we use the notation $[G]$ for the extended graph.

For each $\vec{y} = \langle y_1 \cdots y_{k-1} \rangle \in [V]$, $j = k-1, k, k+1, \ldots, l$, let $S^d[\vec{x}, j]$ be the maximum score of aligning a sequence ending with $\vec{y}$ to $\langle h_1 \cdots h_j \rangle$ where h_j is aligned to a gap (and y_{k-1} is aligned to some $h_i, i < j$). Further let $T^d[\vec{x}, j]$ be the maximum score of aligning a sequence ending with $\langle y_1 \cdots y_{k-1} \rangle$, to $\langle h_1 \cdots h_j \rangle$ where h_j is aligned to y_{k-1}. Initialize, for each $\vec{y}$:

$$S^d[\vec{y}, k-1] = -\infty; \quad (18)$$

$$T^d[\vec{y}, k-1] = \begin{cases} 0 & \vec{y} = \triangleright^{k-1} \\ -\infty & \text{otherwise} \end{cases} \quad (19)$$

Loop over $j = k, \ldots, l$, and for each $\vec{y} = \langle y_1 \cdots y_{k-1} \rangle \in [V]$, recursively update:

$$S^d[\vec{y}, j] = \max\{T^d[\vec{y}, j-1] + \alpha_j, S^d[\vec{y}, j-1] + \beta_j\} \quad (20)$$

$$T^d[\vec{y}, j] = \mathcal{L}^{(j)}[y_{k-1}, h_j] + \max_{e=(\vec{z},\vec{y}) \in E} \Big\{ w(e) + \max \begin{Bmatrix} T^d[\vec{z}, j-1] + \widehat{\alpha_j} \\ S^d[\vec{z}, j-1] + \widehat{\beta_j} \end{Bmatrix} \Big\} \quad (21)$$

Finally, return:

$$MAXScore^d = T^d[\triangleleft^{k-1}, l] \quad (22)$$

The complexity of this algorithm is still $O(l[K])$ and a linear space variant can be obtained, similarly to the one presented previously.

Insertions and Deletions

In this subsection, and in the following one, we present two distinct algorithms for sequence reconstruction assuming both insertions and deletions with respect to the reference sequence. The algorithm we present in this section is a relatively simple extension of the dynamic programs we have presented thus far. The only change is that a different weighted graph is used, forcing higher complexity. In the next section we will also present a faster algorithm.

The algorithm presented in the previous section computes a maximum-likelihood target sequence, when the log-likelihood is a sum of edge weights along the weighted de-Bruijn graph (G, w), and log-probabilities derived from homology. In this section, we compute a different weighted graph, (G', w'). Substituting (G, w) for (G', w'), the algorithm described in the previous section solves the problem variant with both deletions and insertions.

We introduce some more notation. Denote by $\mathcal{T}_j$ the target prefix whose last nucleotide is aligned to h_j in the reference sequence. Further denote by a_j (respectively, b_j) the log-probability of initiating (extending) an insertion in the target after $\mathcal{T}_j$, and define $\widehat{a_j} = 1 - a_j$, $\widehat{b_j} = 1 - b_j$.

Proc. ISMB '00, pp. 260–268, AAAI Press, Menlo Park, CA (2000)

Consider the weighted graph (G, w). Define the $K \times K$ matrix W as follows:

$$W[\vec{x}, \vec{y}] = \begin{cases} 2^{w(\vec{y})} & \text{The } (k-1)\text{-suffix of } \vec{x} \\ & \text{is the } (k-1)\text{-prefix of } \vec{y}. \\ 0 & \text{Otherwise.} \end{cases} \tag{23}$$

$W^i[\vec{x}, \vec{y}]$ is thus the probability of moving from $\vec{x}$ to $\vec{y}$ along i edges. The probability of an insertion of length i after $\mathcal{T}_j$ is $a_j b_j^i \widehat{b_j}$. Suppose that the prefix T_j ends with $\vec{x}$. Then $a_j b_j^{i-1} \widehat{b_j} W^i[\vec{x}, \vec{y}]$ is the probability of $\mathcal{T}_{j+1}$ ending with $\vec{y}$ and being i nucleotides longer than $\mathcal{T}_j$. We are now ready to compute the matrix W', that governs the stochastic progression from $\mathcal{T}_j$ to $\mathcal{T}_{j+1}$:

$$\begin{aligned} W' &= \widehat{a_j} W + a_j b_j W^2 \widehat{b_j} + a_j b_j^2 W^3 \widehat{b_j} \cdots & (24) \\ &= \widehat{a_j} W + a_j b_j \widehat{b_j} W^2 \sum_{i \geq 2} b_j^{i-2} W^{i-2} & (25) \\ &= \widehat{a_j} W + a_j b_j \widehat{b_j} W^2 (I - b_j W)^{-1} & (26) \end{aligned}$$

We define a new weighted graph (G', w'). The vertex set of G is also the vertex set of G'. The edge set E' of G' is the set of all pairs $(\vec{x}, \vec{y})$ with $W'[\vec{x}, \vec{y}] > 0$. Each such edge $e = (\vec{x}, \vec{y})$ is associated with a weight $w'(e) = \log W'[\vec{x}, \vec{y}]$. One can apply the algorithm for deletions only, but use (G', w') instead of (G, w). This solves the problem for insertions and deletions.

Note that in contrast to G, degrees in G' are not bounded by 4. Therefore, computing each dynamic program cell has complexity $O(K)$ in the worst case, with the total complexity of the algorithm being $O(l|E'|)$. Again, considering only the effective size of the graph allows more efficient computation, taking $O(l|[E']|)$. Unfortunately, this is may be $\Omega(l[K]^2)$ in the worst case.

A Faster Algorithm

A general probabilistic model of homology can facilitate a more efficient algorithm that allows both insertions and deletions. Hidden Markov Models (HMMs) were proved useful for profiling protein families (Krogh *et al.* 1994). We use a similar formulation to describe homology between nucleotide sequences. The reference, along with the statistical assumptions, actually creates a profile.

Below, we briefly sketch the model. The reader is referred to (Durbin *et al.* 1998, chapter 5) for details. The model assumes a set Q of Markov chain states with a predefined set of allowed transitions between them. For each level (position along the profile) $j = 1, \ldots, l_Q$, Q includes three states: M_j (match), I_j (insert), and D_j (delete). M_j and D_j can be reached from the three $(j-1)$-th level states. I_j can be reached from the three j-th level states (including a self-loop). Transition probabilities are as described in previous sections, e.g., $a_j = Prob(M_j \mapsto I_j)$. Additionally, each insert or match state, q, induces a vector of emission probabilities $\mathcal{M}^q$, where $\mathcal{M}^q[i]$ is the probability that the target nucleotide is i. We denote $\mathcal{L}^q[i] \equiv 0$ for $q = D_j$, $\mathcal{L}^q[i] \equiv \log \mathcal{M}^q[i]$ otherwise. We write $lpb(\mathcal{X}) \equiv \log Prob(\mathcal{X})$ for short.

The dynamic programming scheme we use for ungapped homology cannot be directly modified to handle the HMM because of the insertion loops. To wit, we generalize this scheme by an additional dimension, which denotes the position along the target sequence.

Define a three dimensional array S, where for each $q \in Q, \vec{y} = \langle y_1 \cdots y_{k-1} \rangle \in [V], r = k, \ldots, L$ let $S[q, \vec{y}, r]$ be the maximum score of an r-long sequence ending with $\langle y_1 \cdots y_{k-1} \rangle$, whose alignment to the profile ends in q. Initialize:

$$\begin{aligned} S[q_{start}, \triangleright^{k-1}, k-1] &= 0 & (27) \\ S[q, \vec{y}, k-1] &= -\infty \quad \text{for other values of } \vec{y}, q & (28) \end{aligned}$$

Loop over $r = k, \ldots l$, and for each $\vec{y} = \langle y_1 \cdots y_{k-1} \rangle \in [V]$, $r \leq l_Q$, recursively update:

$$\begin{aligned} S[q, \vec{y}, r] = \; & \mathcal{L}^q[y_{k-1}] \\ & + \max_{\substack{e = (\vec{z}, \vec{y}) \in E \\ q' | q' \mapsto q}} \{ S[q', \vec{z}, r-1] \\ & \qquad + lpb(q' \mapsto q) \\ & \qquad + w(e) \} \end{aligned} \tag{29}$$

Finally, return:

$$MAXScore = \max_l \{ S[q_{end}, \triangleleft^{k-1}, l] \} \tag{30}$$

Let L be some known bound on the size of the target sequence. Naive implementation of this algorithm requires $O(l_Q \cdot [K] \cdot L)$ time and space. By the means presented earlier, the complexity of this algorithm can be reduced to $O(l_Q \cdot [K] \cdot L \log L)$ time and $O(l_Q \cdot [K])$ memory. Furthermore, one can consider the dynamic program as filling a $l_Q \times L$ matrix, with a $[K]$-long vector in each matrix cell. Since all values far from the main diagonal of this matrix should be negligible, we can settle for computing only values with distance smaller than R to the main diagonal, reducing the complexity to $O(R(l_Q + L) \cdot [K] \cdot \log L)$ time and $O(R(l_Q + L) \cdot [K])$ space.

Computational Results

The algorithm was implemented and tested on simulated data. As a reference, we used prefixes of real genomic DNA sequences: the gene-rich human mitochondrial sequence, accession number `J01415` (Table 1 and Table 2), and the longest single-exon gene on the X-chromosome of C. elegans, `F20B4.7` (Tables 3-5), which is located at positions 17620436..17623111 along this chromosome. Each simulated run randomly generates:

1. A target sequence according to a prescribed probability q of substitution.
2. A probabilistic 8-spectrum according to a prescribed hybridization error.

For simplicity, insertions and deletions were not allowed, substitutions were equiprobable, and we used

Proc. ISMB '00, pp. 260–268, AAAI Press, Menlo Park, CA (2000)

a single parameter p to fix the hybridization error, setting $P_i(x) = 1 - p$ if $I(x) = i$. Furthermore, all probabilistic parameters were constant, i.e., position/k-mer independent. For each parameter set, several simulated data sets were generated and the algorithm was applied to each.

We quantified the performance by the following figures of merit:

1. Full success rate - The fraction of runs for which the target sequence was perfectly reconstructed.
2. Δ-success rate - The fraction of runs for which the target sequence of length l was reconstructed with fewer than $\Delta \cdot l$ base-calling errors.
3. Average sequencing error - The fraction of base-calling errors made by our algorithm.

Table 1 and Table 3 present results for a scenario of distinct, but closely related sequences, e.g., orthologous genes in a pair of primates. We assume perfect hybridization data, with 97% sequence identity. The results show that sequences of length up to 2000 can be reconstructed almost perfectly. The non-monotonicity of the figures of merit with respect to the target length is probably due to sequence content.

Tables 2,4, and 5 present results for a scenario of SNP-genotyping. We assume the rate of SNPs to be $1:700$ (Wang *et al.* 1998), and simulate an error of $p = 2\%$ (Table 2 and Table 4) or $p = 5\%$ (Table 5) in the hybridization data. The results show that high success rate is achievable even in the presence of spectrum errors. Obviously, further simulations are needed to explore the limits of the method.

length	#runs	% full success	% Δ-success $\Delta = 10^{-3}$	% Δ-success $\Delta = 2 \cdot 10^{-3}$	%avg. error
500	10	100	100	100	0.000
1000	10	100	100	100	0.000
1500	10	100	100	100	0.000
2000	17	94	94	94	0.003
2500	13	46	53	69	0.295
3000	14	71	78	78	0.488
3500	5	0	20	20	4.091
4000	13	76	84	84	2.173
4500	11	9	18	45	0.091
5000	15	0	13	53	4.149
5500	7	14	28	71	0.119

Table 1: Results on simulations with Human mitochondrial sequence as reference. The target was generated with $q = 3\%$ difference from the reference, and the spectrum was then generated from the target with no hybridization errors (perfect data).

The algorithm was implemented in $C{+}{+}$ and executed on Linux and SGI machines. Running times, on a Pentium 3, 600MHz machine, were roughly $0.12 l \log l$ seconds for an l-long reference sequence (ranging from roughly 7 minutes for a 500bp-long sequence to 2.5 hours for 6Kb). Only the main memory was used, with

length	#runs	% full success	% Δ-success $\Delta = 10^{-3}$	% Δ-success $\Delta = 2 \cdot 10^{-3}$	%avg. error
250	10	100	100	100	0.000
500	10	100	100	100	0.000
750	10	90	90	100	0.013
1000	10	90	90	90	0.010
1250	10	90	100	100	0.032
1500	12	91	100	100	0.033
1750	10	60	80	80	0.109
2000	10	60	90	90	4.965
2500	10	0	80	100	10.312
3000	10	30	70	90	0.230

Table 2: Results on simulations with Human mitochondrial sequence as reference. The target was generated with $q = 1:700$ difference from the reference, and the spectrum was then generated from the target with $p = 2\%$ hybridization errors.

length	#runs	% full success	% Δ-success $\Delta = 10^{-3}$	% Δ-success $\Delta = 2 \cdot 10^{-3}$	%avg. error
1000	10	100	100	100	0.000
1250	10	100	100	100	0.000
1500	10	100	100	100	0.000
1750	10	100	100	100	0.000
2000	10	100	100	100	0.000
2250	10	70	90	90	0.049
2500	10	50	70	100	0.056
2676	10	100	100	100	0.000

Table 3: Results on simulations with C. elegans genomic sequence as reference. Simulation parameters were as in Table 1: $q = 3\%$, $p = 0$.

length	#runs	% full success	% Δ-success $\Delta = 10^{-3}$	% Δ-success $\Delta = 2 \cdot 10^{-3}$	%avg. error
1000	10	100	100	100	0.000
1250	10	50	50	50	0.304
1500	10	100	100	100	0.000
1750	10	90	90	100	0.011
2000	10	90	90	100	0.020
2250	10	40	60	60	7.333
2500	10	20	30	60	0.232
2676	10	20	30	60	0.175

Table 4: Results on simulations with C. elegans genomic sequence as reference. Simulation parameters were as in Table 2: $q = 1:700$, $p = 2\%$.

Proc. ISMB '00, pp. 260--268, AAAI Press, Menlo Park, CA (2000)

length	#runs	% full	% Δ-success		%avg.
		success	$\Delta = 10^{-3}$	$\Delta = 2 \cdot 10^{-3}$	error
1000	10	100	100	100	0.000
1250	10	20	20	20	0.720
1500	10	70	70	70	0.120
1750	10	100	100	100	0.000
2000	10	80	90	100	0.025
2250	10	0	0	0	0.573
2500	10	30	30	30	0.340
2676	10	30	30	50	5.724

Table 5: Results on simulations with C. elegans genomic sequence as reference. Simulation parameters were $q = 1 : 700$, $p = 5\%$.

the application consuming at most 40Mb. As a first implementation, we did not reduce the graph to its effective size. This would of course reduce both space and time dramatically, at the expense of possibly missing the truely maximal scoring sequence.

Discussion

We have developed a new method that combines spectrum data and homology information in order to algorithmically reconstruct a target sequence. The method is general enough to allow for insertions and deletions, hybridization errors, and a profile or a HMM instead of a single reference sequence. As the spectrum data needed originates from standard chips that can easily be mass produced, the cost of generating the hybridization data can potentially be reduced to a very small fraction in comparison to current special-purpose chips.

The algorithm was implemented and tested, and very preliminary simulation results are reported here. Resequencing a target of up to 2000 nucleotides was demonstrated with good success rate, when a reference sequence with realistic SNP level is known, even in the presence of 2% or 5% spectrum errors. For homology detection, with 3% mismatches, up to 2000 nucleotides were shown to be resequenced almost perfectly assuming no spectrum errors. As expected, the results demonstrate the tradeoff between the target sequence length, the quality of the hybridization data, and the homology distance required.

This paper constitutes a proof of concept, and there is much room for further work. Modifications and improvements are possible both in the theoretical analysis, and in the implementation details, especially in view of some of the practical considerations.

On the algorithmic side, there are several promising possible refinements and future developments:

- It is clear that one can do better using the exact likelihood score, instead of the approximate score that we give. Refined examination of the errors made by our application, suggests that post-processing the output sequence, locally modifying it in search for the optimum point of the exact score, may correct many of these errors.
- For simplicity, we imposed on the problem the independence assumption leading to Equation 1. This obviously oversimplifies the problem. For instance, oligonucleotides corresponding to k-substrings which occur proximally along the target sequence would have correlated hybridization results. Thus, replacing the independence assumption by a more realistic one will render the results more practical.
- Another promising direction is sequence profiles, which have been extensively used for protein sequences (Krogh *et al.* 1994). Our method can be applied, as is, also for sequencing a target whose reference is not a single related sequence, but rather an HMM profile (for nucleotides, instead of amino acids).

Practical aspects suggest several extensions to the basic procedure, in order to render it applicative to real-life data:

- Though the algorithm is polynomial for fixed k, its dependence on k is exponential (4^k), which makes it rather slow in practice. Using the effective size of the graph instead, as suggested, would economize on time and space considerably. This may be necessary for incrementing k, as microarray technology progresses.
- Since the target is usually a PCR product, its primer sequences are known. This information can and should be used when initializing and terminating our dynamic program.
- The target is usually diploid DNA, and might be a heterozygote. The algorithm should thus be extended to handle a hybridization signal from two sequences.
- When searching for mutations in many candidate genes, or exons, one could use pooling of hybridization targets. The algorithm can be easily modified to re-sequence several short targets in one experiment, instead of a single long one.
- Despite their mathematical elegance, all-k-mer chips have practical problems. Oligonucleotides with different GC-content impose conflicting constraints on the experiment temperature, and some are unusable due to self loops. Our algorithm can be modified to work with a collection of equi-temperature oligos, instead of equal-length ones, and to compensate for defunct k-mers.

Acknowledgments

The first author was supported by the Clore foundation scholarship. The second author was supported in part by a grant from the Ministry of Science, Israel.

References

Bains, W., and Smith, G. C. 1988. A novel method for nucleic acid sequence determination. *J. Theor. Biology* 135:303–307.

Proc. ISMB '00, pp. 260–268; AAAI Press, Menlo Park, CA (2000)

Ben-Dor, A.; Pe'er, I.; Shamir, R.; and Sharan, R. 1999. On the complexity of positional sequencing by hybridization. In *Proceedings of the Tenth International Conference on Combinatorial Pattern Matching (CPM' 99)*, 88–100. New York: ACM Press.

Cargill, M.; Altshuler, D.; Ireland, J.; Sklar, P.; Ardlie, K.; Patil, N.; Lane, C. R.; Lim, E. P.; Kalyanaraman, N.; Nemesh, J.; Ziaugra, L.; Friedland, L.; Rolfe, A.; Warrington, J.; Lipshutz, R.; Daley, G. Q.; and Lander, E. S. 1999. Characterization of single-nucleotide polymorphisms in coding regions of human genes. *Nature Genetics* 22(3):231–238.

Drmanac, R., and Crkvenjakov, R. 1987. Yugoslav Patent Application 570.

Durbin, R.; Eddy, S.; Krough, A.; and Mitchison, G. 1998. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids.* Cambridge University Press.

Eddy, S. R. 1996. Hidden markov models. *Current Opinions in Structural Biology* 6(3):361–365.

Hacia, J. G.; Fan, J.; Ryder, O.; Jin, L.; Edgemon, K.; Ghandour, G.; Aeryn Mayer, R.; Sun, B.; Hsie, L.; Robbins, C. M.; Brody, L. C.; Wang, D.; Lander, E. S.; Lipshutz, R.; Fodor, S. P. A.; and Collins, F. S. Determination of ancestral alleles for human single-nucleotide polymorphisms using high-density oligonucleotide arrays.

Hacia, J. G. 1999. Resequencing and mutational analysis using oligonucleotide microarrays. *Nature Genetics* 21:42–47.

Hirschberg, D. S. 1975. A linear space algorithm for computing maximal common subsequences. *Communications of the ACM* 18,6:341–343.

Jukes, T. H., and Cantor, C. R. 1969. Evolution of protein molecules. In Munro, H., ed., *Mammalian protein metabolism.* New York: Academic Press. 21–123.

Khrapko, K. R.; Lysov, Y. P.; Khorlyn, A. A.; Shick, V. V.; Florentiev, V. L.; and Mirzabekov, A. D. 1989. An oligonucleotide hybridization approach to DNA sequencing. *FEBS letters* 256:118–122.

Kimura, M. 1980. A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences. *Journal of Molecular Evolution* 16:111–120.

Krogh, A.; Brown, M.; Mian, S.; Sjölander, M.; and Haussler, D. 1994. Hidden markov models in computational biology. applications to protein modeling. *Journal of Molecular Biology* 235(5):1501–1531.

Lysov, Y.; Floretiev, V.; Khorlyn, A.; Khrapko, K.; Shick, V.; and Mirzabekov, A. 1988. DNA sequencing by hybridization with oligonucleotides. *Dokl. Acad. Sci. USSR* 303:1508–1511.

Macevics, S. C. 1989. International Patent Application PS US89 04741.

National Center for Biotechnology Information. 2000. dbsnp: A database of single nucleotide polymorphisms. `http://www.ncbi.nlm.nih.gov/SNP/`.

Pevzner, P. A., and Lipshutz, R. J. 1994. Towards DNA sequencing chips. In *Symposium on Mathematical Foundations of Computer Science*, 143–158. Springer. LNCS vol. 841.

Pevzner, P. A.; Lysov, Y. P.; Khrapko, K. R.; Belyavsky, A. V.; Florentiev, V. L.; and Mirzabekov, A. D. 1991. Improved chips for sequencing by hybridization. *J. Biomol. Struct. Dyn.* 9:399–410.

Pevzner, P. A. 1989. l-tuple DNA sequencing: computer analysis. *J. Biomol. Struct. Dyn.* 7:63–73.

Preparata, F. P., and Upfal, E. 2000. Sequencing by hybridization at the information theory bound: an optimal algorithm. In *Proceedings of the Fourth Annual International Conference on Computational Molecular Biology (RECOMB '00)*, 88–100. New York: ACM Press.

Preparata, F.; Frieze, A.; and Upfal, E. 1999. Optimal reconstruction of a sequence from its probes. *Journal of Computational Biology* 6(3-4):361–368.

Skiena, S. S., and Sundaram, G. 1995. Reconstructing strings from substrings. *J. Comput. Biol.* 2:333–353.

Smith, T. F., and Waterman, M. S. 1981. Identification of common molecular subsequences. *Journal of Molecular Biology* 147(1):195–197.

Southern, E. M.; Maskos, U.; and Elder, J. K. 1992. Analyzing and comparing nucleic acid sequences by hybridization to arrays of oligonucleotides: evaluation using experimental models. *Genomics* 13:1008–1017.

Southern, E. 1988. UK Patent Application GB8810400.

Southern, E. M. 1996. DNA chips: analysing sequence by hybridization to oligonucleotides on a large scale. *Trends in Genetics* 12:110–115.

Wang, D. G.; Fan, J.; Siao, C.; A.Berno; Young, P.; Sapolsky, R.; Ghandour, G.; Perkins, N.; Winchester, E.; Spencer, J.; Kruglyak, L.; Stein, L.; Hsie, L.; Topaloglou, T.; Hubbell, E.; Robinson, E.; Mittmann, M.; Morris, M. S.; Shen, N.; Kilburn, D.; Rioux, J.; Nusbaum, C.; Lipshutz, R.; Chee, M.; and Lander, E. S. 1998. Large scale identification, mapping, and genotyping of single-nucleotide polymorphisms in the human genome. *Science* 280:1077–1082.

Yang, Z. 1993. Maximum-likelihood estimation of phylogeny from DNA sequences when substitution rates differ over sites. *Molecular Biology and Evoluion* 10:1396–1401.