1996

Methods for Comparing a DNA Sequence with a Protein Sequence

Xiaoqiu Huang
Iowa State University, xqhuang@iastate.edu

Jinghui Zhang
National Institutes of Health

CABIOS

Vol. 12 no. 6 1996
Pages 497–506

Methods for comparing a DNA sequence with a protein sequence

Xiaoqiu Huang and Jinghui Zhang[1]

Abstract

We describe two methods for constructing an optimal global alignment of, and an optimal local alignment between, a DNA sequence and a protein sequence. The alignment model of the methods addresses the problems of frameshifts and introns in the DNA sequence. The methods require computer memory proportional to the sequence lengths, so they can rigorously process very huge sequences. The simplified versions of the methods were implemented as computer programs named NAP and LAP. The experimental results demonstrate that the programs are sensitive and powerful tools for finding genes by DNA–protein sequence homology.

Introduction

The discovery of sequence homology between a newly determined genomic sequence and a known protein sequence serves two purposes. First, it identifies a coding region of the genomic sequence. Second, it provides the first clues about the function of the gene coded by the region (Altschul *et al.*, 1990). Methods for comparing a DNA sequence and a protein sequence are a valuable tool for the analysis of DNA and protein sequences (Pearson, 1990).

We describe two alignment methods for comparing a DNA sequence and a protein sequence. The methods construct an optimal global alignment of, and an optimal local alignment between, the two sequences. The methods generalize that of States and Botstein (1991) by addressing the problem of introns in the DNA sequence. States and Botstein (1991) addressed the problem of frameshifts in the comparison of a mRNA sequence and a protein sequence. Because our methods rigorously solve the problems of frameshifts and introns in the DNA sequence, they are more sensitive than existing DNA–protein alignment methods. The memory requirements of our methods are proportional to the sum of the sequence lengths, and the time requirements are proportional to the product of the sequence lengths. The simplified versions of the methods were implemented as computer programs named NAP (Nucleotide-Amino acid alignment Program) and LAP (Local Alignment Program). The experimental results demonstrate that the programs are sensitive and powerful tools for finding genes by DNA–protein sequence homology.

Department of Computer Science, Michigan Technological University, 1400 Townsend Drive, Houghton, MI 49931 and [1]National Center for Biotechnology Information, National Library of Medicine, National Institutes of Health, Bethesda, MD 20894, USA

E-mail: huang@cs.mtu.edu

System and methods

The programs described in this paper were written in the C programing language. They have been compiled and tested on Sun workstations using the Sun C compiler. We think that the programs are portable to many platforms.

Algorithm

A global alignment model

An alignment of a DNA sequence and a protein sequence consists of substitutions, nucleotide insertion gaps and nucleotide deletion gaps. A substitution involves at most three nucleotides and one amino acid. A full substitution involves three nucleotides and a partial substitution involves one or two nucleotides. A full substitution is a match if it contains no gap and the codon codes for the amino acid. Gaps are defined relative to the DNA sequence. A nucleotide insertion gap is a gap where nucleotides correspond to no amino acids. A nucleotide deletion gap is a gap where no nucleotides are present. Only one nucleotide insertion gap is allowed to occur within a full substitution. No nucleotide insertion gap is allowed to occur within a partial substitution. The length of a gap is the number of nucleotides involved. Figure 1 shows an alignment of DNA and protein sequences. This alignment contains a match (ATG, Met) and two partial matches (A**, Arg) and (TG*, Cys), where an occurrence of the symbol * indicates that no nucleotide is present at the position.

Let the non-negative integers q and r be gap-open and gap-extension penalties. The score of a gap of length l is $-(q + l \times r)$. Let $t(b, \hat{b})$ be the score of substituting an amino acid $\hat{b}$ for an amino acid b. The score table σ for substitutions is computed from the table t as follows. Let $g(a_1a_2a_3)$ be the amino acid coded by a non-stop codon $a_1a_2a_3$, where each a_i is in {A, C, G, T}. The score of a full substitution involving a non-stop codon $a_1a_2a_3$ and an amino acid b is defined to be $\sigma(a_1a_2a_3, b) = t(g(a_1a_2a_3), b)$. The score of a full substitution involving a stop codon and an amino acid is defined to be the minimum value in the table t. For a partial substitution

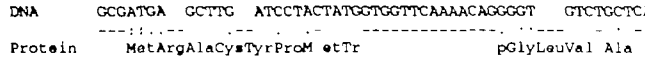

Fig. 1. An alignment of DNA and protein sequences. A match is indicated by three colons and a partial match by up to three fullstops. Gaps are indicated by dashes.

$(a_1a_2^*, b)$, $\sigma(a_1a_2^*, b)$ is defined to be the arithmetic average of those $\sigma(a_1a_2x, b)$ for x in {A, C, G, T} such that a_1a_2x is a non-stop codon. The score $\sigma(a_1^{**}, b)$ is defined to be the arithmetic average of those $\sigma(a_1xy, b)$ for x and y in {A, C, G, T} such that a_1xy is a non-stop codon. The scores of other partial substitutions are defined similarly. The score of a substitution with the codon containing Ns is the same as the score of the substitution with the Ns replaced by *s. For example, σ(ANT, Ser) = σ(A*T, Ser). The score of an alignment is simply the sum of scores of each substitution and each gap in the alignment.

Three adjustments to the scoring scheme are needed to produce a biologically meaningful alignment. First, as in Huang (1994), terminal gaps are given a score of zero. This adjustment handles well the case where the DNA sequence contains untranslated 5′ and 3′ regions. Second, a nucleotide insertion gap of length greater than k is given a constant penalty of $q + k \times r$, where k is a parameter specified by the user. The modification addresses the problem of long introns in the DNA sequence by favoring alignments with long internal nucleotide insertion gaps over alignments with many short nucleotide insertion gaps. Note that the alignment of the DNA sequence and the protein sequence may contain insertions of long introns of the DNA sequence. Third, a nucleotide insertion gap of length greater than k is given a 5′ bonus if it begins with GT and a 3′ bonus if it ends with AG. This adjustment encourages long nucleotide insertions to occur at splice sites. As a result, the exact locations of the introns are likely to be shown on a largest-scoring alignment. If k is set to 10, and the 5′ and 3′ bonuses are $3r$ each, then the score of the alignment in Figure 1 is:

$$\begin{aligned}\sigma(\text{ATG, Met}) &+ \sigma(\text{A}^{**}, \text{Arg}) - (q + 2r) + \sigma(\text{GCT, Ala})\\ &+ \sigma(\text{TG}^*, \text{Cys}) + \sigma(^*\text{AT, Tyr}) - (q + 2r)\\ &+ \sigma(\text{CCT, Pro}) + \sigma(\text{ATA, Met}) - (q + r)\\ &+ \sigma(\text{TGG, Trp}) - (q + 10r) + 3r + 3r\\ &+ \sigma(\text{GGT, Gly}) - (q + 3r) + \sigma(\text{GTC, Val})\\ &- (q + r) + \sigma(\text{GCT, Ala})\end{aligned}$$

Our global alignment model generalizes the local alignment model of States and Botstein (1991) by allowing a nucleotide insertion gap of any length within a full substitution, charging a constant penalty for any nucleotide insertion gap of length greater than k, and utilizing the GT and AG dinucleotides in the identification of RNA splice sites. These extensions address the problem of introns in the alignment of DNA and protein sequences.

A global alignment algorithm

Let $A = a_1a_2\cdots a_m$ be a DNA sequence. Let $B = b_1b_2\cdots b_n$ be a protein sequence, where each b_j is an amino acid. The problem is to compute an alignment of A and B with the maximum score. This alignment is called an optimal alignment. We develop a dynamic programing algorithm for computing an optimal alignment of A and B. To obtain an efficient algorithm, we employ the techniques of Gotoh (1982) for treating the linear gap penalty and for treating the constant penalty for a gap of length greater than k. Let $S(i, j)$ be the maximum score of any alignment of A_i and B_j, where $A_i = a_1a_2\cdots a_i$ and $B_j = b_1b_2\cdots b_j$. In order to compute the matrix S efficiently, we need to introduce a number of additional matrices. First we define each additional matrix and give a recurrence for computing the matrix. Then we give the recurrence for computing the matrix S. Figure 2 shows the dependence of entry (i, j) in a matrix on some of the other entries in this matrix and other matrices.

Let $E(i, j)$ be the maximum score of any alignment of A_i and B_j that ends with a nucleotide insertion gap occurring outside codons. If the gap is a 3′-terminal gap, then the gap is not penalized. This is handled by a separate formula for the case where $j = n$. The matrix E is computed according to the recurrence:

$$\begin{aligned}E(i, j) &= -q \text{ for } i = 0 \text{ and } j > 0,\\ E(i, j) &= \max\{E(i-1, j) - r, S(i-1, j) - q - r\}\\ &\qquad \text{for } i > 0 \text{ and } 0 < j < n,\\ E(i, j) &= \max\{E(i-1, j), S(i-1, j)\}\\ &\qquad \text{for } i > 0 \text{ and } j = n\end{aligned}$$

Let $F(i, j)$ be the maximum score of any alignment of A_i and B_j that ends with a nucleotide deletion gap. A separate formula for the case where $i = m$ is introduced for not penalizing any 3′-terminal gap.

$$\begin{aligned}F(i, j) &= -q \text{ for } i > 0 \text{ and } j = 0,\\ F(i, j) &= \max\{F(i, j-1) - 3r,\\ &\qquad F(i-1, j-1) + \sigma(^*a_i^*, b_j) - q - 2r,\\ &\qquad S(i, j-1) - q - 3r,\\ &\qquad S(i-1, j-1) + \sigma(a_i^{**}, b_j) - q - 2r,\\ &\qquad S(i-1, j-1) + \sigma(^*a_i^*, b_j) - 2q - 2r,\\ &\qquad S(i-2, j-1) + \sigma(a_{i-1}a_i^*, b_j) - q - r\}\\ &\qquad \text{for } 0 < i < m \text{ and } j > 0,\\ F(i, j) &= \max\{F(i, j-1), S(i, j-1)\} \text{ for } i = m \text{ and } j > 0.\end{aligned}$$

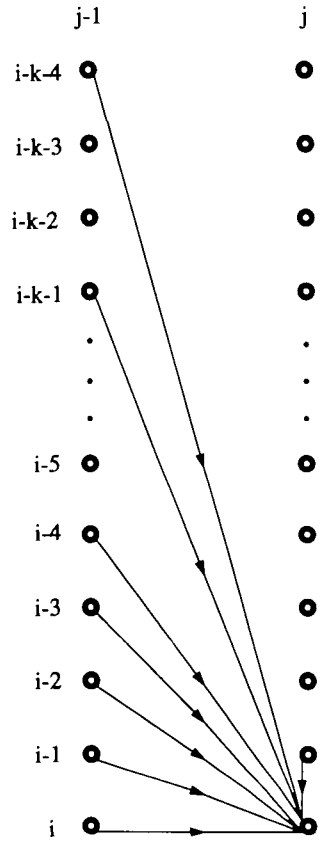

Fig. 2. The dependence of entry (i,j) in a matrix on some of the other entries in this and other matrices. The edge from entries $(i-1,j)$ to (i,j) denotes a nucleotide insertion gap. The edge from entries $(i,j-1)$ to (i,j) denotes a nucleotide deletion gap. The edges from entries $(i-1,j-1)$ and $(i-2,j-1)$ to (i,j) denote partial substitutions. The edge from entries $(i-3,j-1)$ to (i,j) denotes a full substitution. The edge from entries $(i-4,j-1)$ to (i,j) denotes a full substitution containing a nucleotide insertion gap. The edge from entries $(i-k-1,j-1)$ to (i,j) denotes a nucleotide insertion gap of length greater than k. The edge from entries $(i-k-4,j-1)$ to (i,j) denotes a full substitution containing a nucleotide insertion gap of length greater than k.

Let $G(i,j)$ be the maximum score of any alignment of A_i and B_j that ends with a nucleotide insertion gap of length greater than k occurring outside codons. The gap is given a penalty of $q + k \times r$. Let $I_5(i)$ be the 5′ bonus if $a_i a_{i+1} = \mathrm{GT}$ and zero otherwise.

$$G(i,j) = -q - k \times r \text{ for } i \leq k \text{ and } j > 0,$$

$$G(i,j) = \max\{G(i-1,j), S(i-k-1,j) + I_5(i-k) - q - k \times r\} \text{ for } i > k \text{ and } j > 0.$$

A partial score of an alignment ending with a full substitution is the score of the alignment minus the score of the last substitution. Let $C(i,j,b)$ be the maximum partial score of any alignment of A_i and B_j that ends with a full substitution containing a nucleotide insertion gap between the first and second bases of the codon for amino acid b. Let $lc(i,j,b)$ be the position of the first base of the codon if such an alignment exists and zero otherwise. If $lc(i,j,b)$ is not zero and the codon $a_{lc(i,j,b)} a_{i-1} a_i$ codes for amino acid b, then $C(i,j,b) + t(b,b_j)$ is the maximum score of any alignment of A_i and B_j that ends with a full substitution containing a nucleotide insertion gap between the first and second bases of the codon.

$$C(i,j,b) = -\infty \text{ for } i \leq 3 \text{ and } j > 0,$$

$$C(i,j,b) = \max\{C(i-1,j,b) - r, S(i-4,j-1) - q - r\} \text{ for } i \geq 4, j > 0 \text{ and } a_{i-3} \text{ is base 1 of a codon for } b,$$

$$C(i,j,b) = C(i-1,j,b) - r \text{ for } i \geq 4, j > 0 \text{ and } a_{i-3} \text{ is not base 1 of any codon for } b.$$

$$lc(i,j,b) = 0 \text{ for } i \leq 3 \text{ and } j > 0,$$

$$lc(i,j,b) = i - 3 \text{ for } i \geq 4, j > 0, \text{ and } C(i,j,b) > C(i-1,j,b) - r,$$

$$lc(i,j,b) = lc(i-1,j,b) \text{ for } i \geq 4, j > 0, \text{ and } C(i,j,b) = C(i-1,j,b) - r.$$

Let $D(i,j,b)$ be the maximum partial score of any alignment of A_i and B_j that ends with a full substitution containing a nucleotide insertion gap between the second and third bases of the codon for amino acid b. Let $ld(i,j,b)$ be the position of the first base of the codon if such an alignment exists and zero otherwise. If $ld(i,j,b)$ is not zero and the codon $a_{ld(i,j,b)} a_{ld(i,j,b)+1} a_i$ codes for amino acid b, then $D(i,j,b) + t(b,b_j)$ is the maximum score of any alignment of A_i and B_j that ends with a full substitution containing a nucleotide insertion gap between the second and third bases of the codon. The recurrence for ld is similar to that for lc and is omitted.

$$D(i,j,b) = -\infty \text{ for } i \leq 3 \text{ and } j > 0,$$

$$D(i,j,b) = \max\{D(i-1,j,b) - r, S(i-4,j-1) - q - r\} \text{ for } i \geq 4, j > 0 \text{ and } a_{i-3} a_{i-2} \text{ is bases 1 and 2 of a codon for } b,$$

$$D(i,j,b) = D(i-1,j,b) - r \text{ for } i \geq 4, j > 0 \text{ and } a_{i-3} a_{i-2} \text{ is not bases 1 and 2 of any codon for } b.$$

Two matrices are introduced to handle a nucleotide insertion gap of length greater than k. Let $U(i,j,b)$ be the maximum partial score of any alignment of A_i and B_j that ends with a full substitution containing a nucleotide insertion gap of length greater than k between the first and second bases of the codon for amino acid b. Let $lu(i,j,b)$ be the position of the first base of the codon if such an alignment

exists and 0 otherwise.

$$U(i,j,b) = -\infty \text{ for } i < k+4 \text{ and } j > 0,$$

$$U(i,j,b) = \max\{U(i-1,j,b), S(i-k-4,j-1) + I_5(i-k-2) - q - k \times r\}$$
for $i \geq k+4$, $j > 0$ and a_{i-k-3} is base 1 of a codon for b,

$$U(i,j,b) = U(i-1,j,b) \text{ for } i \geq k+4,\ j > 0 \text{ and } a_{i-k-3}$$
is not base 1 of any codon for b.

$$lu(i,j,b) = 0 \text{ for } i < k+4 \text{ and } j > 0,$$

$$lu(i,j,b) = i-k-3 \text{ for } i \geq k+4,\ j > 0$$
and $U(i,j,b) > U(i-1,j,b)$,

$$lu(i,j,b) = lu(i-1,j,b) \text{ for } i \geq k+4,\ j > 0$$
and $U(i,j,b) = U(i-1,j,b)$.

Let $V(i,j,b)$ be the maximum partial score of any alignment of A_i and B_j that ends with a full substitution containing a nucleotide insertion gap of length greater than k between the second and third bases of the codon for amino acid b. Let $lv(i,j,b)$ be the position of the first base of the codon if such an alignment exists and zero otherwise. The recurrence for lv is similar to that for lu and is omitted.

$$V(i,j,b) = -\infty \text{ for } i < k+4 \text{ and } j > 0,$$

$$V(i,j,b) = \max\{V(i-1,j,b), S(i-k-4,j-1) + I_5(i-k-1) - q - k \times r\}$$
for $i \geq k+4$, $j > 0$, and $a_{i-k-3}a_{i-k-2}$ is bases 1 and 2 of a codon for b,

$$V(i,j,b) = V(i-1,j,b) \text{ for } i \geq k+4,\ j > 0$$
and $a_{i-k-3}a_{i-k-2}$ is not bases 1 and 2 of any codon for b.

Let $S(i,j)$ be the maximum score of any alignment of A_i and B_j. Let $I_3(i)$ be the 3′ bonus if $a_{i-1}a_i = \text{AG}$ and zero otherwise. The matrix S is computed according to the recurrence:

$$S(i,j) = 0 \text{ for } i = 0 \text{ or } j = 0,$$

$$S(i,j) = \max\{S(i-1,j-1) + \sigma(^{**}a_i, b_j) - q - 2r,$$
$$S(i-2,j-1) + \sigma(a_{i-1}{}^{*}a_i, b_j) - q - r,$$
$$S(i-2,j-1) + \sigma(^{*}a_{i-1}a_i, b_j) - q - r,$$
$$S(i-3,j-1) + \sigma(a_{i-2}a_{i-1}a_i, b_j),$$
$$F(i-1,j-1) + \sigma(^{**}a_i, b_j) - 2r,$$
$$F(i-2,j-1) + \sigma(^{*}a_{i-1}a_i, b_j) - r,$$
$$E(i,j), F(i,j), G(i,j) + I_3(i),$$
$$C(i,j,b^c) + t(b^c, b_j),$$
$$D(i,j,b^d) + t(b^d, b_j),$$
$$U(i,j,b^u) + t(b^u, b_j) + I_3(i-2),$$
$$V(i,j,b^v) + t(b^v, b_j) + I_3(i-1)\}$$
for $i > 0$ and $j > 0$,

where the maximum is taken over amino acids b^c, b^d, b^u and b^v such that $lc(i,j,b^c) \neq 0$ and the codon $a_{lc(i,j,b^c)}a_{i-1}a_i$ codes for amino acid b^c, $ld(i,j,b^d) \neq 0$ and the codon $a_{ld(i,j,b^d)}a_{ld(i,j,b^d)+1}a_i$ codes for amino acid b^d, $lu(i,j,b^u) \neq 0$ and the codon $a_{lu(i,j,b^u)}a_{i-1}a_i$ codes for amino acid b^u, $lv(i,j,b^v) \neq 0$ and the codon $a_{lv(i,j,b^v)}a_{lv(i,j,v^v)+1}a_i$ codes for amino acid b^v. Assume that the expressions with a negative index are removed from the recurrence for $S(i,j)$. For example, for $i = 2$, the expression involving $S(i-3, j-1)$ is not present in the recurrence.

The score of an optimal alignment of A and B is $S(m,n)$. The score $S(m,n)$ can be obtained in linear space by computing the matrices in order of columns. We directly compute an optimal alignment of score $S(m,n)$ in linear space using a divide-and-conquer technique (Hirschberg, 1975; Myers and Miller, 1988; Huang, 1994). In this technique, the midpoint of the optimal alignment is computed by a forward and a reverse pass, and then the alignments on both sides of the midpoint are computed recursively. Myers and Miller (1988) extended the algorithm of Hirschberg (1975) to handle the linear gap penalty. Huang (1994) further generalized the technique to compute an optimal alignment of two sequences of the same type, where terminal gaps are not penalized and long gaps are given a constant penalty. We modify the procedure of Huang (1994) to compute an optimal alignment of DNA and protein sequences of lengths m and n in $O(mn)$ time and $O(m+n)$ space.

The algorithm described above is not very efficient because the matrices $C(i,j,b), D(i,j,b), U(i,j,b)$ and $V(i,j,b)$ have to be computed for each possible amino acid b. Below we give a simplified version of the algorithm, which achieves a greater efficiency with a little compromise on alignment optimality. The simplification involves combining $C(i,j,b)$ for all amino acids b into a single matrix $C(i,j)$. This simplification is also performed for $U(i,j,b)$ and $V(i,j,b)$. The matrix $D(i,j,b)$ is no longer needed since its simplification is the same as that of $C(i,j,b)$. The new recurrences for the matrices S, C, lc, U, lu and V are given below. Two expressions involving $S(i-4, j-1)$ are included in the recurrence for S to compute accurately the maximum score of alignments of A_i and B_j that end with a full substitution containing a nucleotide insertion gap of length one. Note that the simplified algorithm approximately computes the maximum score of alignments of A_i and B_j that end with a full

substitution containing a nucleotide insertion gap of length greater than one. The recurrence for lv is similar to that for lu and is omitted. No change is made to the recurrences for the matrices E, F and G.

$$S(i,j) = 0 \text{ for } i = 0 \text{ or } j = 0,$$

$$\begin{aligned} S(i,j) = \max\{ & S(i-1,j-1) + \sigma(^{**}a_i, b_j) - q - 2r, \\ & S(i-2,j-1) + \sigma(a_{i-1}{}^{*}a_i, b_j) - q - r, \\ & S(i-2,j-1) + \sigma(^{*}a_{i-1}a_i, b_j) - q - r, \\ & S(i-3,j-1) + \sigma(a_{i-2}a_{i-1}a_i, b_j), \\ & S(i-4,j-1) + \sigma(a_{i-3}a_{i-1}a_i, b_j) - q - r, \\ & S(i-4,j-1) + \sigma(a_{i-3}a_{i-2}a_i, b_j) - q - r, \\ & F(i-1,j-1) + \sigma(^{**}a_i, b_j) - 2r, \\ & F(i-2,j-1) + \sigma(^{*}a_{i-1}a_i, b_j) - r, \\ & E(i,j), F(i,j), G(i,j) + I_3(i), \\ & C(i,j) + \sigma(a_{lc(i,j)}a_{i-1}a_i, b_j), \\ & C(i,j) + \sigma(a_{lc(i,j)}a_{lc(i,j)+1}a_i, b_j), \\ & U(i,j) + \sigma(a_{lu(i,j)}a_{i-1}a_i, b_j) + I_3(i-2), \\ & V(i,j) + \sigma(a_{lv(i,j)}a_{lv(i,j)+1}a_i, b_j) + I_3(i-1)\} \\ & \text{for } i > 0 \text{ and } j > 0, \end{aligned}$$

where $lc(i,j) \neq 0$, $lu(i,j) \neq 0$ and $lv(i,j) \neq 0$.

$$\begin{aligned} C(i,j) &= -\infty \text{ for } i \leq 3 \text{ and } j > 0, \\ C(i,j) &= \max\{C(i-1,j) - r, S(i-4,j-1) - q - r\} \\ &\quad \text{for } i \geq 4 \text{ and } j > 0. \end{aligned}$$

$$\begin{aligned} lc(i,j) &= 0 \text{ for } i \leq 3 \text{ and } j > 0, \\ lc(i,j) &= i - 3 \text{ if } i \geq 4,\ j > 0, \\ &\quad \text{and } C(i,j) > C(i-1,j) - r, \\ lc(i,j) &= lc(i-1,j) \text{ if } i \geq 4,\ j > 0, \\ &\quad \text{and } C(i,j) = C(i-1,j) - r. \end{aligned}$$

$$\begin{aligned} U(i,j) &= -\infty \text{ for } i < k+4 \text{ and } j > 0, \\ U(i,j) &= \max\{U(i-1,j), S(i-k-4,j-1) \\ &\quad + I_5(i-k-2) - q - k \times r\} \\ &\quad \text{for } i \geq k+4 \text{ and } j > 0. \end{aligned}$$

$$\begin{aligned} lu(i,j) &= 0 \text{ for } i < k+4 \text{ and } j > 0, \\ lu(i,j) &= i - k - 3 \text{ for } i \geq k+4, j > 0 \\ &\quad \text{and } U(i,j) > U(i-1,j), \\ lu(i,j) &= lu(i-1,j) \text{ for } i \geq k+4, j > 0 \\ &\quad \text{and } U(i,j) = U(i-1,j). \end{aligned}$$

$$\begin{aligned} V(i,j) &= -\infty \text{ for } i < k+4 \text{ and } j > 0, \\ V(i,j) &= \max\{V(i-1,j), S(i-k-4,j-1) \\ &\quad + I_5(i-k-1) - q - k \times r\} \\ &\quad \text{for } i \geq k+4 \text{ and } j > 0. \end{aligned}$$

An alignment of score $S(m,n)$ is computed in linear space using a divide-and-conquer technique.

A local alignment algorithm

If there is no correspondence between the entire protein sequence and a portion of the DNA sequence, then it is inappropriate to use a global alignment program to compare the two sequences. The global alignment may not contain a local similarity between the two sequences even if the local similarity exists. A local alignment algorithm should be used to find a local similarity between two sequences (Smith and Waterman, 1981).

A local alignment between a DNA sequence and a protein sequence is an alignment of a region of the DNA sequence and a region of the protein sequence. An optimal local alignment is one with the maximum score, and the maximum score is the similarity score between the two sequences. We develop an algorithm for computing an optimal local alignment between the DNA and protein sequences. Following the method of Smith and Waterman (1981), we obtain the recurrences of the local alignment algorithm by modifying those of the global alignment algorithm given in the previous subsection. The modification is to include the term zero in the recurrence for $S(i,j)$. The technique of Huang *et al.* (1990) is used to obtain an optimal local alignment in linear space. In this technique, the end-points of an optimal local alignment are first determined, and then the optimal local alignment is constructed by applying the linear-space global alignment procedure to the corresponding regions of the two sequences.

If the set of the simplified recurrences is used with the term zero added to the recurrence for $S(i,j)$, then we have a simplied local alignment algorithm. This algorithm achieves a higher efficiency with a little loss of alignment optimality.

Implementation

The simplified algorithm for computing a global alignment of DNA and protein sequences was implemented as a portable computer program named NAP. The simplified local alignment algorithm was implemented as a portable computer program named LAP. The programs were written in the C programing language. The NAP and LAP programs have an option to compare the reverse complement of the DNA

sequence with the protein sequence. The parameters used in the experiments described below are $q = 10$, $r = 2$, $k = 15$. Setting k to 15 means that each nucleotide insertion gap of 15 bp or more receives a penalty of 40. The selection of a small value for k encourages the programs to identify short exons correctly by aligning regions of the protein sequence with the short exons of the DNA sequence because the gaps before and after a short exon are not heavily penalized. If a large value is used for k, and hence long gaps are heavily penalized, then the programs tend to combine the two long gaps before and after a short exon into a single gap, resulting in a miss of the short exon. Note that the penalty of a nucleotide insertion gap of length greater than k is the constant $q + k \times r$. The BLOSUM62 matrix was used as the score table t (Henikoff and Henikoff, 1992). All the experiments involving NAP and LAP were performed on a Sun Sparcstation 5 with 32 megabytes of memory. The computational times reported below for NAP and LAP do not include the time to compare the reverse complement of the DNA sequence with the protein sequence.

We performed a test to determine the effect of alignment similarity on the accuracy of NAP. For this test, we selected a human α globin DNA sequence of 900 bp (GenBank Accession V00491), which contains three exons. The locations of the three exons were determined by aligning the DNA sequence with the human α globin protein sequence with NAP. As expected, NAP produced an alignment that consists only of exact matches, which shows the exact location of each exon. The comparison of the starting and ending positions of the exons shown by the NAP alignment with the positions of the exons given in the GenBank annotation of the DNA sequence indicates that three positions in the 'CDS' field of the GenBank annotation are off by 1–3 bp. We extracted a total of 508 protein entries whose 'DE' field begins with 'HEMOGLOBIN' from the Swiss-Prot protein database (Release 32). The NAP program was used to align the human α globin DNA sequence with each of the 508 protein sequences. Figure 3 shows an alignment of the human α globin DNA sequence and a pig θ globin protein sequence.

For each of the 508 alignments, we calculated the percent similarity and error rate of the alignment. The percent similarity of an alignment is the number of matches divided by the length of the protein sequence. The error rate of an alignment is the number of protein residues that are aligned with non-exonic regions of the DNA sequence divided by the protein sequence length. The 508 alignments were partitioned into 20 groups by similarity, with group 1 consisting of the alignments whose similarity is <5%, group 2 consisting of the alignments whose similarity is at least 5%, but is <10%, and so on. Table I shows the number of alignments, the average score of alignments, and the average error rate of alignments in each group. The results in Table I indicate that NAP is able to identify the correlation between DNA and protein

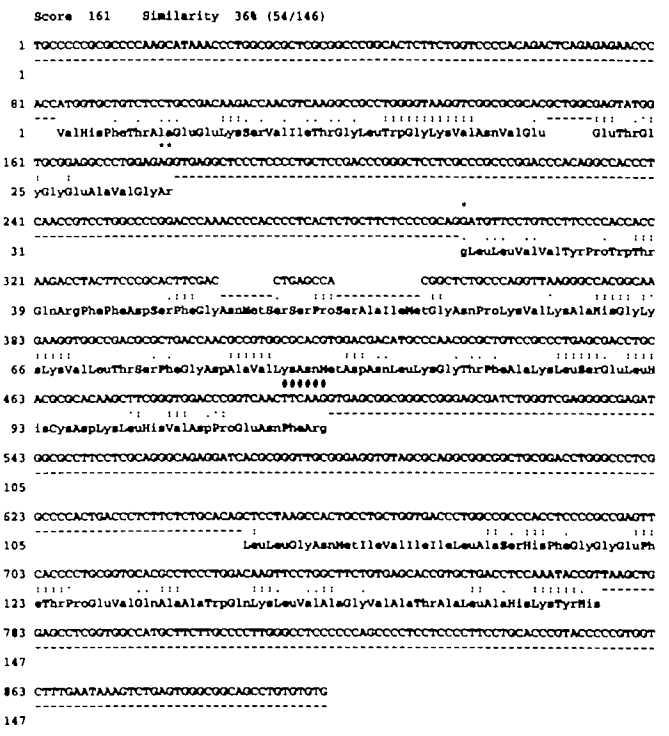

Fig. 3. A global alignment of a human α globin DNA sequence (GenBank Accession V00491) and a pig θ globin protein sequence (Swiss-Prot Accession P04246). The NAP program identified the exact locations of the two introns. The regions of interest are indicated by asterisks and pound signs.

sequences if the two sequences have a similarity of 30% or more.

NAP can be used to annotate coding region features in a genomic sequence automatically because it has several special features that were designed accurately to align a genomic DNA sequence to a related protein sequence. Long introns do not distort the DNA–protein alignment because a constant penalty is charged for any gap exceeding a user-specified length. The 5′ and 3′ untranslated regions do not affect the alignment because terminal gaps are not penalized. The program also identifies the donor and receptor splice sites for each intron so that the results are consistent with the intron splicing process. As an example, we aligned a human genomic sequence for the cytokine α subunit to a mouse macrophage inflammatory α1 protein sequence. The length of the DNA sequence is 4788 bp and the length of the protein sequence is 92 amino acids. The result is shown in Figure 4. NAP took 6.5 s. The alignment projects a three-exon coding region in the human genomic sequence. The first exon is from base 2217 to base 2292, the second from base 2978 to base 3092, and the third from base 3514 to base 3601. The size of the first intron is 685 bp and that of the second intron is 421 bp. At the junction of the second and the third exons, NAP was able to recognize a spliced codon where the first two bases of the codon reside in the second exon, while the

Table I. Effect of alignment similarity on the accuracy of NAP

Similarity range (%)	Number of alignments	Average score	Average error rate (%)[a]
0–5	4	1	100.00
5–10	2	13	83.39
10–15	0		
15–20	0		
20–25	1	72	37.93
25–30	17	111	34.77
30–35	19	120	3.42
35–40	125	156	0.86
40–45	94	184	0.62
45–50	15	283	0.09
50–55	11	314	0.13
55–60	43	364	0.10
60–65	9	358	0.00
65–70	33	450	0.00
70–75	12	473	0.00
75–80	4	521	0.00
80–85	44	563	0.00
85–90	46	590	0.00
90–95	13	617	0.00
95–100	16	652	0.00

[a]The average error rate of alignments in a similarity range is the total number of protein residues that are aligned with non-exonic regions of the DNA sequence divided by the total protein length.

last one is in the third exon. The prediction corresponds perfectly with the original annotation in GenBank. Without utilizing the pattern information about the splice sites, the program would produce a slightly different result in the junction of the first and the second exons.

The NAP program can handle long sequences because of its low computer memory requirement. As an example, NAP was used to compare a human DNA sequence (GenBank Accession X52889) for cardiac β myosin heavy chain with a *Dictyostelium discoideum* myosin heavy chain protein sequence (PIR Accession A26655). The length of the DNA sequence is 25 000 bp and the length of the protein sequence is 2116 amino acids. NAP produced a huge alignment of 25 210 bp in 13.2 min. The alignment has a match percentage of 32%. The alignment was examined to see how well NAP identified the splice sites. To visualize the positions of the splice sites of the DNA sequence, we also used NAP to align the human DNA sequence with the protein sequence that is coded by the DNA sequence. As expected, NAP produced a perfect alignment, which shows the positions of 74 splice sites. The perfect alignment was compared with the alignment of the human DNA and *D.discoideum* protein sequences. Out of the 74 splice sites, 46 splice sites were exactly identified by NAP, 19 were approximately identified by NAP (off by at most 19 bp), and nine were missed by NAP. NAP completely missed two exons; one is exon 1, which is only 10% similar to the corresponding part of the protein sequence, and the other is exon 38, which is only 18 bp long. The results are satisfactory, considering that the overall similarity between the exon sequences and the protein sequence is only 32%.

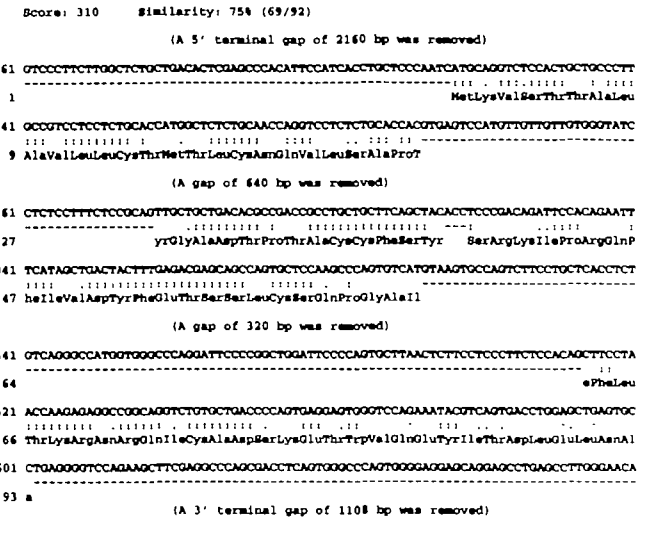

Fig. 4. A global alignment of a human genomic sequence for the cytokine α subunit (GenBank Accession M24110) and a mouse macrophage inflammatory α1 protein sequence (Swiss-Prot Accession P10855). The NAP program identified the exact locations of the two long introns.

The local alignment program LAP should be used if the DNA and protein sequences are not globally similar, but have similar regions. This situation occurs when the DNA and protein sequences contain several domains, only some of which are similar. For example, a human DNA trk proto-oncogene sequence of 2701 bp contains the five domains according to the GenBank annotation: (1) a putative signal peptide; (2) an amino-terminal moiety rich in consensus sites for *N*-glycosylation; (3) a transmembrane domain; (4) a kinase catalytic region highly related to that of other tyrosine kinases; (5) a very short carboxy-terminal tail. A mouse protein sequence of 1115 amino acids for proto-oncogene tyrosine-protein kinase receptor ret precursor contains the three domains according to the Swiss-Prot annotation: (1) an extracellular region (residues 29–637); (2) a transmembrane region (residues 638–659); (3) a protein kinase domain (residues 725–1017). The two sequences only align at the transmembrane domain and at the protein kinase domain. The LAP program was used to compare the two sequences. LAP produced a local alignment between the two sequences at the kinase domain in 33.3 s. The alignment is shown in Figure 5.

The NAP and LAP programs can tolerate sequencing errors that result in frameshifts in a coding region. As an example, we used LAP to align a DNA sequence of 2501 bp from *Saccharomyces cerevisiae* chromosome VIII and a hypothetical yeast protein sequence of 473 amino acids. This coding region of *S.cerevisiae* chromosome VIII was discovered by a fast database search program we recently developed (X.Huang, in preparation). The coding region contains a number of frameshifts. LAP produced a local alignment between the DNA and protein sequences in 22.8 s. The

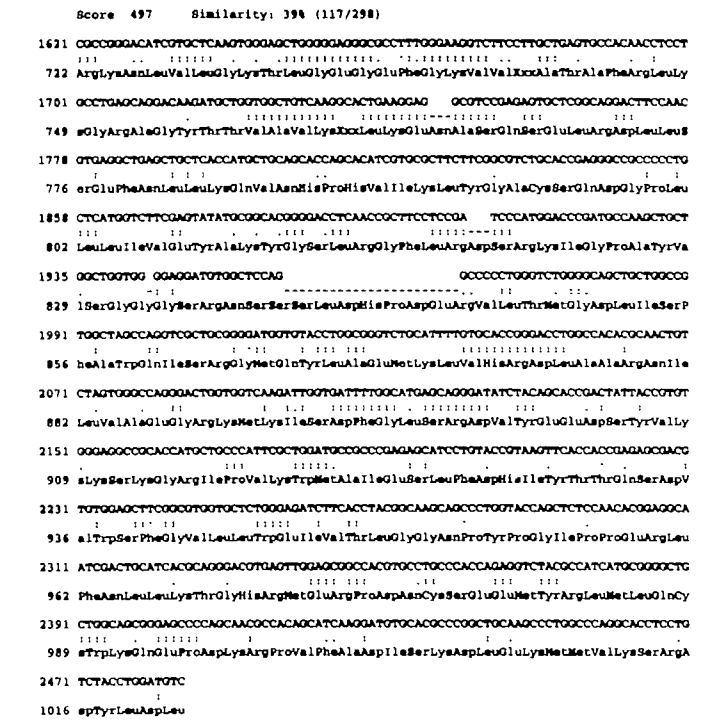

Fig. 5. A local alignment between a human trk proto-oncogene DNA sequence (GenBank Accession M23102) and a mouse protein sequence for proto-oncogene tyrosine-protein kinase receptor ret precursor (Swiss-Prot Accession P35546). The two sequences have several domains. The alignment shows the correlation between the sequences at a common domain. The alignment was produced by the local alignment program LAP.

alignment shown in Figure 6 provides a clear view of the relationship between the two sequences.

Comparisons with other alignment systems

In addition to the alignment model of States and Botstein (1991), two other alignment systems have recently been developed (Gelfand *et al.*, 1996; Guan and Uberbacher, 1996). As discussed previously, our alignment model generalizes that of States and Botstein (1991) by allowing a long nucleotide insertion gap within a codon. The alignments in Figures 3 and 4 demonstrate that NAP can exactly identify the location of an intron within a spliced codon. However, this is not the case with the model of States and Botstein since no long nucleotide insertion gap within a codon is allowed on their model. In addition, the NAP and LAP programs require space proportional to the sum of the sequence lengths, while the program of States and Botstein requires space proportional to the product of the sequence lengths. Thus, the NAP and LAP programs can handle much longer DNA and protein sequences. Note that our alignment model keeps the capability to handle frameshifts in the DNA sequence from the model of States and Botstein.

Guan and Uberbacher (1996) developed an alignment algorithm that compares a protein sequence with the 3-frame

Fig. 6. A local alignment between the reverse complement of a region (202 000–204 500) of the *S.cerevisiae* chromosome VIII sequence (Johnston *et al.*, 1994) and a yeast protein sequence (Swiss-Prot Accession P36032). The alignment contains many gaps that result in frameshifts. The alignment was produced by the local alignment program LAP.

translations of a DNA sequence. The algorithm allows the alignment to shift from one frame translation to another frame translation. Since the algorithm of Guan and Uberbacher uses only the 3-frame translated sequences, it cannot accurately handle situations where a nucleotide insertion/deletion gap occurs within a codon. The algorithm approximately handles the situations using gaps that occur outside any codon. As a consequence, the algorithm cannot exactly identify an intron that occurs within a codon. In addition, the algorithm of Guan and Uberbacher requires quadratic space, so it cannot process huge sequences. Note that the types of alignment configurations that are available on the model of Guan and Uberbacher are also available on our alignment model.

We compared our programs with the program of Guan and Uberbacher (1996) on the three pairs of DNA and protein sequences: the globin DNA and protein sequences, the yeast DNA and protein sequences, and the myosin heavy chain DNA and protein sequences. The alignment parameters used

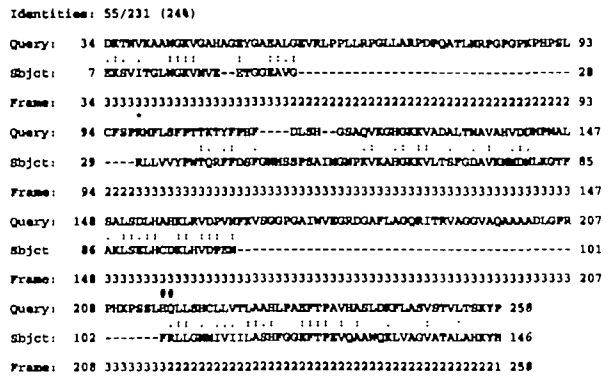

Fig. 7. A local alignment between the globin DNA and protein sequences produced by the program of Guan and Uberbacher (1996). The query sequence is the translation of the DNA sequence. The regions of interest are indicated by an asterisk and pound signs. The program was not able to identify the spliced codon containing intron 1 (see the column indicated by an asterisk). The program misplaced two residues F and R of the protein sequence at the region indicated by pound signs.

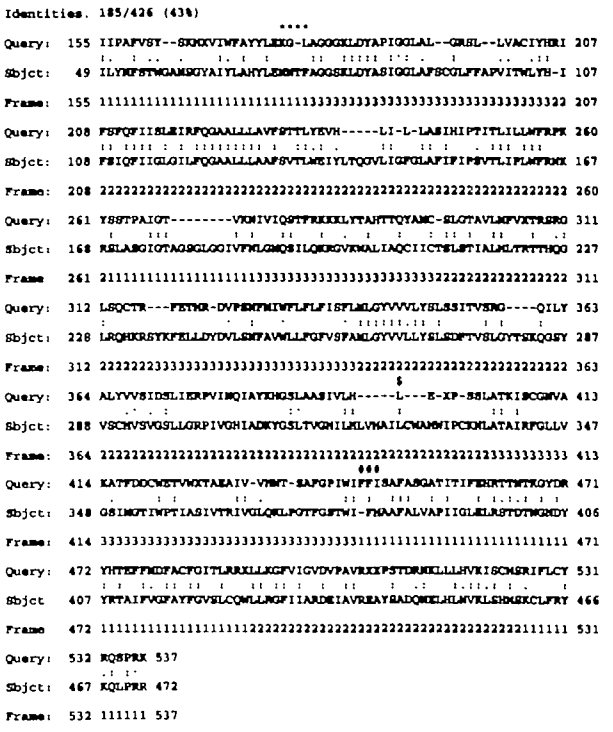

Fig. 8. A local alignment between the yeast DNA and protein sequences produced by the program of Guan and Uberbacher (1996). The query sequence is the translation of the DNA sequence. The alignment does not show the correlation between the two sequences at the nucleotide level. For example, it is not clear whether an extra residue T at the region indicated by asterisks corresponds to a deletion of one, two or three nucleotides. A similar comment pertains to an extra residue F at the region indicated by pound signs. The program misplaced a translated residue L of the DNA sequence at the position indicated by a dollar sign.

by the program of Guan and Uberbacher are: the substitution matrix, BLOSUM62; gap open penalty, 10; gap extension penalty, 1; frameshift penalty, 10. Because of its high memory requirement, the program of Guan and Uberbacher was not able to compare the myosin heavy chain DNA sequence of 25 000 bp and the protein sequence of 2116 amino acids.

On the globin DNA and protein sequences, NAP identified the exact locations of the two introns in the DNA sequence (Figure 3). NAP aligned a residue R of the protein sequence with a spliced codon AGG with intron 1 being between bases 2 and 3 of the codon. Introl 2 of the DNA sequence is outside codons. The alignment generated by the program of Guan and Uberbacher on the globin sequences is shown in Figure 7. The program of Guan and Uberbacher was not able to identify the spliced codon containing intron 1. The program also failed to identify the exact location of intron 2. The program misplaced two residues F and R of the protein sequence near the acceptor site of intron 2. An alignment of larger score would be obtained if the two residues were placed near the donor site of intron 2. NAP took 1.9 s on a Sun Sparcstation 5, and the program of Guan and Uberbacher took 1 s on a Sun Sparcstation 10.

On the yeast DNA and protein sequences, LAP produced a detailed view of the correlation between the two sequences (Figure 6). In contrast, the alignment produced by the program of Guan and Uberbacher does not show the correlation between the two sequences at the nucleotide level (Figure 8). For instance, the alignment in Figure 8 shows an extra residue T of the protein sequence at the region indicated by asterisks. However, the LAP alignment in Figure 6 shows that a single base of the DNA sequence is missing at the region indicated by asterisks. As another example, the alignment in Figure 8 shows an extra translated residue F of the DNA sequence at the region indicated by pound signs, while the alignment in Figure 6 shows a single extra base A of the DNA sequence at the region indicated by pound signs. The alignment in Figure 8 is not optimal because a translated residue L of the DNA sequence was placed between two gaps at the position indicated by a dollar sign. An alignment of larger score would be obtained if the two gaps were combined into one by aligning the translated residue L with another protein residue L at the end of the gap. LAP took 22.8 s on a Sun Sparcstation 5, and the program of Guan and Uberbacher took 28 s on a Sun Sparcstation 10.

Gelfand *et al.* (1996) proposed an algorithm for solving the problem of introns in the DNA sequence. In addition to DNA and protein sequences, the algorithm takes as input a set of regions of the DNA sequence. The algorithm computes a subset of ordered non-overlapping regions such that the concatenation of the regions and the protein sequence have the maximum similarity score. The effectiveness and efficiency of the algorithm depend on selection of the set of regions. Selection of a small set of regions increases the efficiency of the algorithm, but compromises the effectiveness of the algorithm since the set may not contain all the exons. On the other hand, selection of a large set of regions

improves the effectiveness of the algorithm, but decreases the efficiency of the algorithm. Note that the number of potential regions can be in the order of the square of the DNA sequence length. In contrast, the use of dynamic programing in our algorithms allows us to consider all the promising regions of the DNA sequence efficiently.

Discussion

We have described two sensitive methods for comparing a DNA sequence and a protein sequence. Because of their high time requirements, it may be impractical to employ our methods, on a conventional computer, to compare a DNA sequence against a database of protein sequences. However, the methods can be used to perform the database search on a high-performance computer. This search can identify weak homology between the DNA sequence and a protein sequence in the database, which increases the success rate of finding the coding region of the DNA sequence. Our methods can also be used to refine results produced by fast database search programs such as TFASTA and BLASTX (Pearson and Lipman, 1988; Altschul *et al.*, 1990). Because of frameshifts and introns in the DNA sequence, the fast database search programs produce several small alignments between the DNA sequence and a protein sequence in the database. Our methods construct one large alignment between the two sequences.

Availability

The source codes of NAP and LAP are freely available for academic use on the WWW at http://www.cs.mtu.edu/faculty/huang.html and via anonymous ftp at cs.mtu.edu in directory /pub/huang. For commercial use, contact the Intellectual Property Office of Michigan Technological University.

Acknowledgements

The authors thank David Lipman and James Ostell for critical reading of the manuscript, the reviewers for suggestions that significantly improved the paper, and Xiaojun Guan and Edward Uberbacher for providing the alignments. The work of X.H. was supported in part by the Michigan Research Excellence Fund.

References

Altschul,S.F., Gish,W., Miller,W., Myers,E.W. and Lipman,D.J. (1990) Basic local alignment search tool. *J. Mol. Biol.*, **215**, 403–410.

Gelfand,M.S., Mironov,A.A. and Pevzner, P.A. (1996) Gene recognition via spliced sequence alignment. *Proc. Natl Acad. Sci. USA*, **93**, 9061–9066.

Gotoh,O. (1982) An improved algorithm for matching biological sequences. *J. Mol. Biol.*, **162**, 705–708.

Guan,X. and Uberbacher,E.C. (1996) Alignments of DNA and protein sequences containing frameshift errors. *Comput. Applic. Biosci.*, **12**, 31–40.

Henikoff,S. and Henikoff,J.G. (1992) Amino acid substitution matrices from protein blocks. *Proc. Natl Acad. Sci USA*, **89**, 10915–10919.

Hirschberg,D.S. (1975) A linear space algorithm for computing maximal common subsequences. *Commun. Assoc. Comput. Mach*, **18**, 341–343.

Huang,X. (1994) On global sequence alignment. *Comput. Applic. Biosci.*, **10**, 227–235.

Huang,X., Hardison,R.C. and Miller,W. (1990) A space-efficient algorithm for local similarities. *Comput. Applic. Biosc.*, **6**, 373–381.

Johnston,M. *et al.* (1994) Complete nucleotide sequence of *Saccharomyces cerevisiae* chromosome VIII. *Science*, **265**, 2077–2082.

Myers,E.W. and Miller,W. (1988) Optimal alignments in linear space. *Comput. Applic. Biosci.*, **4**, 11–17.

Pearson,W.R. (1990) Rapid and sensitive sequence comparison with FASTP and FASTA. *Methods Enzymol*, **183**, 63–98.

Pearson,W.R. and Lipman,D. (1988) Improved tools for biological sequence comparison. *Proc. Natl Acad. Sci. USA*, **85**, 2444–2448.

Smith,T.F. and Waterman,M.S. (1981) Identification of common molecular subsequences. *J. Mol. Biol.*, **147**, 195–197.

States,D.J. and Botstein,D. (1991) Molecular sequence accuracy and the analysis of protein coding regions. *Proc. Natl Acad. Sci. USA*, **88**, 5518–5522.

Received on March 4, 1996; revised and accepted on August 28, 1996