ADVANCES IN APPLIED MATHEMATICS 14, 139–171 (1993)

Generalized Sequence Alignment and Duality

PAVEL A. PEVZNER

AND

MICHAEL S. WATERMAN

Departments of Mathematics and of Molecular Biology, University of Southern California, Los Angeles, California 90089-1113

Although a number of efficient algorithms for the longest common subsequence (LCS) problem have been suggested since the 1970s, there is no duality theorem for the LCS problem. In the present paper a simple duality theorem is proved for the LCS problem and for a wide class of partial orders generalizing the notion of common subsequence and sequence alignment. An algorithm for finding generalized alignment is suggested which has the classical dynamic programming approach for alignment problems as a special case. The algorithm covers both local and global alignment as well as a variety of gap functions. It is shown that the generalized LCS problem is closely associated with the minimal Hilbert basis problem. The Jeroslav–Schrijver characterization of minimal Hilbert bases gives an $O(n)$ estimation for the number of elementary edit operations for generalized LCS.

1. INTRODUCTION

A DNA molecule can be represented as a long string of letters from the four-letter alphabet $\{a, c, g, t\}$. Currently a large effort is being expended in the experimental determination and subsequent compilation of these genetic sequences from various organisms. The analysis of these sequences is usually based on ideas from evolution. Sequence features important to organisms are usually preserved over evolutionary time while those less important features change. Therefore the biologist asks what known sequences are closely related to a newly determined sequence. These primary events in sequence evolution are substitution, when one letter is replaced by another, and insertion or deletion of one or more letters. These are the edit operations in the computer science problem of minimum edit distance between two strings. In [A90] algorithms for minimum edit distance problems and finding patterns in sequences are reviewed. In

0196-8858/93 $9.00

this paper we study algorithms for sequence comparison motivated by biological problems. If the two sequences are written on the horizontal and vertical axes, the intersection points on the grid can represent the alignment of two letters, one from each sequence. If the letters are not equal, the node represents the substitution of one for the other. In this way, it is seen that sequence comparison or alignment is a path in a network. Finding optimal alignment is therefore a problem in combinatorial optimization.

Usually algorithms for combinatorial optimization problems are based on duality theorems. The Ford–Fulkerson algorithm, for example, is based on a duality theorem that states maximum network flow is equal to minimum cut. Duality theorems often give insight into the nature of the optimization problems. In this paper, we explore duality theorems and primal–dual algorithms for sequence alignment.

The simplest and most often studied alignment problem in computer science is the longest common subsequence (LCS) problem, which is to find a longest subsequence common to two sequences. The LCS problem is equivalent to the edit problem of finding the minimum number of inserted or deleted letters to transform one sequence into the other. In Section 2 we give a matrix generalization of LCS, A-LCS, for 2×2 matrices A. Several alignment problems of biological interest are included in this family of A-LCS problems. In Sections 3 and 4, we prove a duality result for this class of alignment problems. Each A-LCS problem has an associated partial order $\prec$ and a conjugate partial order $\prec^*$. The length of the A-LCS is the size of a minimum cover.

Recently [EGGI91] raised the problem of devising non-dynamic programming algorithms for sequence alignment. Utilizing the duality result, we give a primal-dual algorithm for A-LCS problems. This algorithm does not appear to be related to the usual dynamic programming algorithms for sequence comparison.

A-LCS is a path in a comparability graph for a partial order. The classical Needleman–Wunsch [NW70] dynamic programming algorithm decomposes each long arc in this graph into short arcs, thereby achieving its efficiency. In Sections 5 and 6, we study similar reductions for A-LCS problems. Further reduction is achieved in Section 7 which applies the theory of Hilbert bases to this problem. We give a geometric interpretation of elementary edit operations for A-LCS and demonstrate that the number of elementary edit operations equals the size of the minimum Hilbert basis in the corresponding cone.

Most algorithms currently used for DNA or protein sequence alignment have more complex weighting functions than those in the LCS problem. Each possible pair of alphabet letters can have different substitution weights, and insertions/deletions of blocks of letters can be weighted as a function of block length or even block composition. In Sections 8, 9, and 10, generalized alignment is defined and an algorithm is given for optimal generalized alignment. The algorithm contains many alignment algorithms as particular cases.

2. Examples and Definitions

A *partially ordered set* or briefly a *poset* is a pair $(P, \prec)$ such that P is a set and $\prec$ is a transitive and irreflexive binary relation on P, i.e., $p \prec q$ and $q \prec r$ imply $p \prec r$. A *chain* is a subset of P, where any two elements are comparable, and an *antichain* is a subset, where no two elements are comparable. A *sequence* in a poset is an ordered chain $p_1 \prec p_2 \prec \cdots \prec p_t$. Partial orders $\prec$ and $\prec^*$ are called *conjugate* [KT82] if for any two distinct $p_1, p_2 \in P$ the following condition holds:

$$p_1 \text{ and } p_2 \text{ are } \prec\text{-comparable} \Leftrightarrow p_1 \text{ and } p_2 \text{ are } \prec^*\text{-incomparable}$$

Let w be an arbitrary non-negative integer valued function on P:

$$w: P \to Z^+.$$

For a partial order $\prec$, a sequence $p_1 p_2 \dots p_k$ in P, maximizing

$$\sum_{i=1}^{k} w(p_i), \tag{1}$$

is called a *longest* $\prec$ *sequence*.

Let $I = \{1, 2, \dots, n\}$ and $J = \{1, 2, \dots, m\}$. As discussed in the Introduction, our interest is in the comparison of two sequences $\mathbf{s} = s_1 s_2 \dots s_n$ and $\mathbf{t} = t_1 t_2 \dots t_m$. For this reason we study $P \subseteq I \times J$ (often $p = (i, j) \in P$ denotes $s_i = t_j$). Let $p_1 = (i_1, j_1)$ and $p_2 = (i_2, j_2)$ be two arbitrary elements in $I \times J$. Denote

$$\Delta i = \Delta i(p_1, p_2) = i_2 - i_1,$$
$$\Delta j = \Delta j(p_1, p_2) = j_2 - j_1,$$
$$\Delta = \Delta(p_1, p_2) = (\Delta i, \Delta j).$$

Consider a few examples of partial orders on $I \times J$ (corresponding sequences are shown in Fig. 1):

- Common subsequences (CS):

$$p_1 \prec_1 p_2 \Leftrightarrow \Delta i > 0, \Delta j > 0;$$

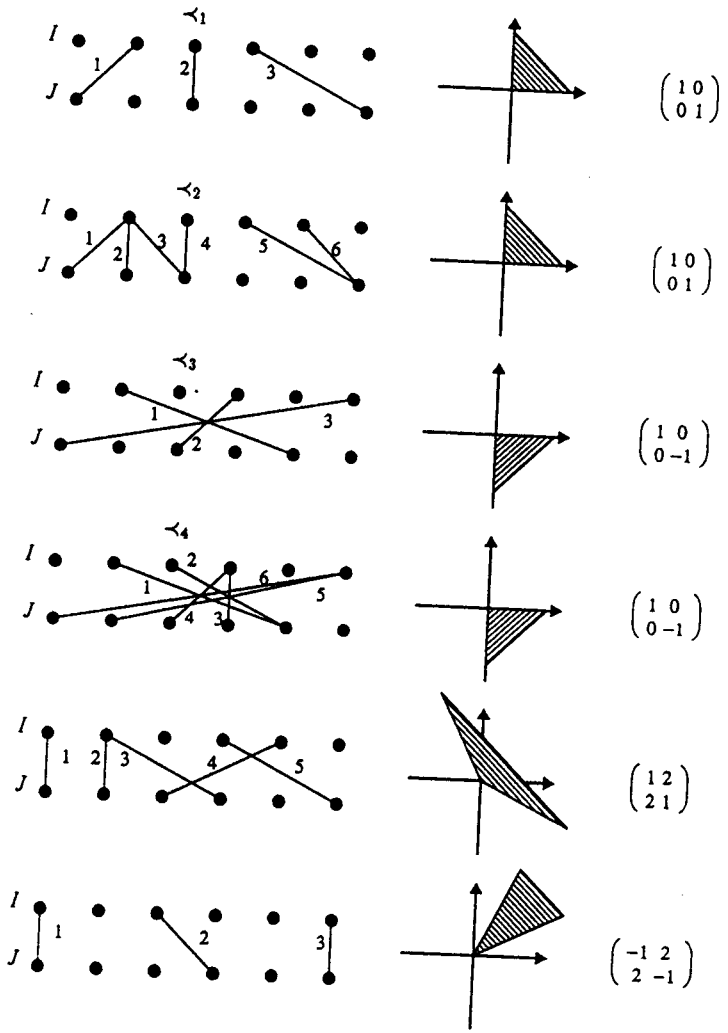

FIG. 1. Examples of sequences and the corresponding cones for various partial orders.

- Common forests (CF):

$$p_1 \prec_2 p_2 \Leftrightarrow \Delta i \geq 0, \Delta j \geq 0;$$

- Common inverted subsequences (CIS):

$$p_1 \prec_3 p_2 \Leftrightarrow \Delta i > 0, \Delta j < 0;$$

- Common inverted forests (CIF):

$$p_1 \prec_4 p_2 \Leftrightarrow \Delta i \geq 0, \Delta j \leq 0.$$

Partial orders $\prec_1$ and $\prec_3$ are particular cases of a partial order defined by an arbitrary 2×2 matrix $A = (a_{ij})$:

$$p_1 \prec_A p_2 \Leftrightarrow A\Delta^{\mathrm{T}} > \mathbf{0}. \tag{2}$$

For $A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ we have $\prec_1$, and for $A = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$ we have $\prec_3$. A partial order defined by A is called an *A-order*, and a sequence in A-order is called an *A-sequence*. Similarly, define *$\bar{A}$-order* and *$\bar{A}$-sequence* by the inequality

$$p_1 \prec_{\bar{A}} p_2 \Leftrightarrow A\Delta^{\mathrm{T}} \geq \mathbf{0}. \tag{3}$$

CFs are $\bar{A}$-sequences for $A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, and CIFs are $\bar{A}$-sequences for $A = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$.

The matrix A determines a *cone* in R^2; the cones for various A-matrices and corresponding A- and $\bar{A}$-sequences are shown at Fig. 1. The set of vectors Δ fulfilling (2) is designated cone(A), while the set of vectors Δ fulfilling (3) is denoted cone($\bar{A}$). For partial order A, an A-sequence $p_1 p_2 \ldots p_k$ in P maximizing (1) is called a *longest common sequence* for A or A-LCS ($\bar{A}$-LCS is defined similarly). For $P = I \times J$, $A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ and w defined for sequences s and t according to the rule

$$w(p) = w(i,j) = \begin{cases} 1, & s_i = t_j, \\ 0, & \text{otherwise}, \end{cases}$$

problem (1) coincides with the well-known *longest common subsequence problem*.

Let $\mathscr{C} = \{C\}$ be a family of subsets of a set P. $\mathscr{C}' \subseteq \mathscr{C}$ is called a *cover* of a function w if:

$\forall p \in P$ there exist at least $w(p)$ subsets in family $\mathscr{C}'$ containing an element p.

For $w \equiv 1$ on P, $\mathscr{C}'$ is a cover if and only if each $p \in P$ is contained in at least one of the subsets $C \in \mathscr{C}'$. The number of elements in $\mathscr{C}'$ is called the *size* of the cover $\mathscr{C}'$ and a cover of minimum size is called a *minimum cover* of w by $\mathscr{C}$.

3. Duality for Longest $\prec$-Sequence Problems

Lemma 1. *Let $\prec$ and $\prec^*$ be conjugate partial orders on P. Then the length of a longest $\prec$-sequence in P equals the size of a minimum cover of w by $\prec^*$-sequences.*

Proof. According to Dilworth's theorem [D50], the length of a longest antichain in $\prec^*$ equals the size of a minimum cover of w by $\prec^*$-chains. As $\prec$ and $\prec^*$ are conjugate, each antichain in $\prec^*$ is a chain in $\prec$ and, vice versa, each chain in $\prec$ is an antichain in $\prec^*$. Therefore the length of a longest $\prec$-sequence in P equals the size of a minimum cover of w by $\prec^*$-sequences. □

Consider a binary relation on P defined by

$$p_1 \sqsubset p_2 \Leftrightarrow p_1 \prec p_2 \text{ or } p_1 \prec^* p_2.$$

Lemma 2. *$\sqsubset$ is a linear order on P.*

Proof. We prove that $p_1 \sqsubset p_2$ and $p_2 \sqsubset p_3$ implies $p_1 \sqsubset p_3$. If $p_1 \sqsubset p_2$ and $p_2 \sqsubset p_3$ then one of the following conditions holds:

(i) $p_1 \prec p_2$ and $p_2 \prec p_3$,
(ii) $p_1 \prec p_2$ and $p_2 \prec^* p_3$,
(iii) $p_1 \prec^* p_2$ and $p_2 \prec p_3$,
(iv) $p_1 \prec^* p_2$ and $p_2 \prec^* p_3$.

In case (i) $p_1 \prec p_2$ and $p_2 \prec p_3$ implies $p_1 \prec p_3$ and therefore $p_1 \sqsubset p_3$. In case (ii) $p_1 \prec p_2$ and $p_2 \prec^* p_3$ implies neither $p_3 \prec p_1$ nor $p_3 \prec^* p_1$. (In the first case $p_3 \prec p_1$ and $p_1 \prec p_2$ implies $p_3 \prec p_2$, contradicting $p_2 \prec^* p_3$. In the second case $p_2 \prec^* p_3$ and $p_3 \prec^* p_1$ implies $p_2 \prec^* p_1$, contradicting $p_1 \prec p_2$). Therefore $p_1 \prec p_3$ or $p_1 \prec^* p_3$, implies $p_1 \sqsubset p_3$. Note that cases (iii) and (iv) are symmetric to (ii) and (i), respectively, so we have shown that relation $\sqsubset$ is transitive. The lemma follows from the observation that for each pair p_1, p_2 either $p_1 \sqsubset p_2$ or $p_2 \sqsubset p_1$. □

Let $\mathscr{P} = p_1 p_2 \dots p_l$ be an arbitrary sequence of the members of P, and $\mathscr{P}_i = p_1 p_2 \dots p_i$. Let $\mathscr{C}_i = \{C_1, C_2, \dots, C_j\}$ be a cover of $\mathscr{P}_i$ by $\prec^*$-sequences and let $p_1^{\max}, p_2^{\max}, \dots, p_j^{\max}$ be the $\prec^*$-maximum elements in $C_1, C_2, \dots, C_j$ correspondingly. Consider an algorithm for constructing a cover $\mathscr{C}_{i+1}$ from $\mathscr{C}_i$. The algorithm is illustrated in Fig. 2.

Algorithm 1. Let k be the minimum index ($1 \le k \le j$) fulfilling the condition (Fig. 2)

$$p_k^{\max} \prec^* p_{i+1}, \tag{4}$$

and if the condition (4) fails for all k set $k = j + 1$.

If $k < j + 1$, add p_{i+1} to C_k and define

$$\mathscr{C}_{i+1} = \{C_1, C_2, \dots, C_{k-1}, C_k \cup \{p_{i+1}\}, C_{k+1}, \dots, C_j\}.$$

If $k = j + 1$ add $\{p_{i+1}\}$ as a new $\prec^*$-sequence to the cover $\mathscr{C}_{i+1}$:

$$\mathscr{C}_{i+1} = \{C_1, C_2, \dots, C_j, C_{j+1} = \{p_{i+1}\}\}.$$

Define also a *reference* ref(p) for p_{i+1} by

$$\text{ref}(p_{i+1}) = \begin{cases} p_{k-1}^{\max}, & \text{if } k > 1 \\ \varnothing, & \text{otherwise.} \end{cases}$$

Assume $\mathscr{C}_1$ consists only of the set $C_1 = \{p_1\}$ and ref(p_1) = $\varnothing$. Applying this algorithm $|P| - 1$ times, we will construct a cover $\mathscr{C}_l$ of P and a set of references ref(p) for each $p \in P$. The size of the cover $\mathscr{C}_l$ depends on the choice of the ordering of P. The size of the cover $\mathscr{C}_l$ is an upper bound for the length of longest $\prec$-sequence. The following lemma shows that if $\mathscr{P}$ is the ordering of P in $\sqsubset$, then Algorithm 1 gives a primal–dual algorithm for simultaneous solutions of (i) the longest $\prec$-sequence problem and (ii) the minimum $\prec^*$-cover problem (we suppose for simplicity that $w \equiv 1$).

Proposition 1. *If $\mathscr{P} = p_1 p_2 \dots p_l$ is the ordering of P in $\sqsubset$, then Algorithm 1 constructs a minimum cover $\mathscr{C}_l = \{C_1, C_2, \dots, C_t\}$ of P by $\prec^*$-sequences. A traceback of references* ref(p) *defines a longest $\prec$-sequence of length t for each $p \in C_t$.*

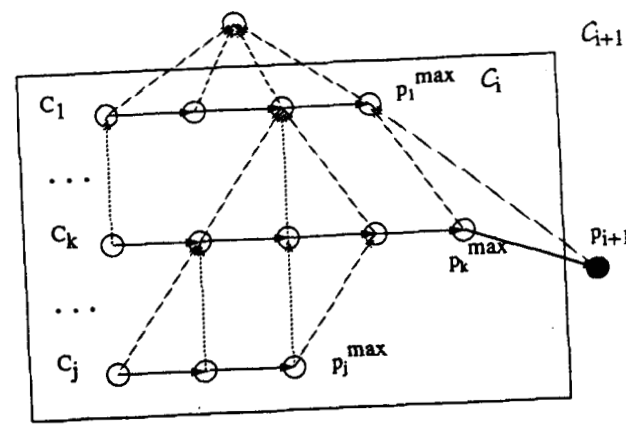

Fig. 2. Addition of p_{i+1} to the $\prec^*$-sequence C_k in the cover $\mathscr{C}_i = \{C_1, C_2, \dots, C_j\}$. The references ref($p$) = q correspond to the (dashed) arcs (p, q).

Proof. We show that for each i $(1 \leq i \leq l)$ the cover $\mathcal{C}_i = \{C_1, C_2, \ldots, C_j\}$ satisfies the condition

Applying Lemma 4 to the matrix $A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ (partial order $\prec_1$) and matrix $A^* = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$ (partial order $\prec_3$), we derive the following theorem:

THEOREM 1.

- *Length of longest CS equals the size of a minimum cover by CIF.*
- *Length of longest CF equals the size of a minimum cover by CIS.*
- *Length of longest CIS equals the size of a minimum cover by CF.*
- *Length of longest CIF equals the size of a minimum cover by CS.*

For the LCS problem and a fixed length alphabet, Algorithm 1 can be implemented in $O(nL)$ time, where L is the length of longest common sequence or in $O((r + n)\log n)$ time, where r is the total number of matches between the two input sequences. Improvements to the classical Needleman–Wunsch algorithm have been suggested by Hirschberg [H77] and Hunt and Szymanski [HS77]. In fact both Hirschberg [H77] and Hunt–Szymanski [HS77] algorithms can be viewed as implementations of Algorithm 1 with various data structures. $\prec^*$-chains in Algorithm 1 correspond to the *k-candidates* in Hirschberg's algorithm. Maximal elements of $\prec^*$-chains in Algorithm 1 correspond to the *dominant matches* in Apostolico's improvement [A86] of Hunt–Szymanski's algorithm. Other related algorithms for the LCS problem can be found in [M80, NKY82, HD84, A86, AG87, KR87, EGGI90]. We mention also that Algorithm 1 provides much more information than just the LCS length (compare Algorithm 1 with the Robinson–Schensted–Knuth algorithm [Saga91] for Young tableaux). The relationships between Algorithm 1, advanced LCS algorithms, and Young tableaux will be considered in detail elsewhere.

5. MAXIMUM PATHS IN GRAPHS AND A-LCS PROBLEM

Without loss of generality we suppose that P has a minimum element p_0 and $w(p_0) = 0$. A longest $\prec$-sequence problem can be reformulated as a "maximum path" problem (for the vertex p_0) in the weighted directed graph $G(P, E, w)$, where E and w are defined by the rule

$$(p_1, p_2) \in E \Leftrightarrow p_1 \prec p_2,$$
$$w(p_1, p_2) = w(p_2).$$

Note the abuse of notation ($w(p_1, p_2) = w(p_2)$) which we introduce for simplifying the arguments. The graph $G(P, E)$ is acyclic, so that a simple modification [L76] of a shortest path algorithm [D59] with running time $O(|P|^2)$ could be used for finding longest $\prec$-sequences.

For $P \subset I \times J$ defined by sequences $\mathbf{s} = atgcaa$ and $\mathbf{t} = agcta$ ($(i, j) \in P$ iff $s_i = t_j$), the corresponding graphs $G(P, E)$ for various A-orders are presented in Fig. 4. For "random" words $\mathbf{s}$ with n-letters and $\mathbf{t}$ with m-letters ($n \geq m$), the graph $G(P, E)$ for the classical LCS-problem has $O(n^2)$ vertices and $O(n^4)$ arcs; therefore Dijkstra's algorithm for this graph runs in $O(n^4)$ time. Sankoff and Sellers [SS73] were the first to consider LCS problem as optimization in the partial order. Sankoff [S72] first proposed an $O(n^2)$ algorithm for the LCS-problem in computer science, but a few authors developed closely related algorithms even earlier in speech processing [V68, VZ70] and in molecular biology [NW70]. As a matter of fact, the contribution of these authors is concerned with transformations of $G(P, E)$ to reduce computational complexity. They increased $|P|$ with a simultaneous significant decrease of $|E|$ by "decomposition" of each "long" arc into short arcs. This transformation does not change the length of longest path in G. As a result they reduced the maximum degree of vertices in G to three and the number of arcs in G was reduced to $O(n^2)$. Johnson's shortest path algorithm [J76] runs in $O(|E|\log|P|)$ time and gives $O(n^2 \log(n))$ complexity for such "sparse" graphs. For the case where arc weights are small integers, Wagner [W76] suggested an algorithm running in $O(|E|)$ time.

The classical Needleman–Wunsch algorithm for the LCS problem has running time $O(n^2)$ due to the special arrangement of vertices of G. Define a function g on the vertices of the graph $G(P, E)$ to be an *arrangement* if for each arc $(p_1, p_2) \in E$: $g(p_1) < g(p_2)$. Let $\mathbf{k} = (k_1, k_2) > \mathbf{0}$ be an arbitrary vector. For $p = (i, j)$ we now define a function f of interest:

$$f(p) = f(i, j) = \mathbf{k} \cdot A \cdot (i, j)^{\mathrm{T}}.$$

As a matter of fact the following lemma means that the vector $\mathbf{k}A$ determines a *systolic schedule* [K88] for the A-LCS problem. Systolic array designs for the LCS problem were suggested in [LL85, YL86, CPHW91].

LEMMA 5. *Let* $\mathbf{k} = (k_1, k_2) > \mathbf{0}$ *and* $f(p) = \mathbf{k} \cdot A \cdot (i, j)^{\mathrm{T}}$. *Then*

(i) $f(p)$ *is an arrangement of graph* $G(P, E)$, *defined by A-order, and*

(ii) *if* A *is non-singular, then* f *is an arrangement of the graph* $G(P, E)$, *defined by* $\bar{A}$*-order.*

Proof. For each $p_1, p_2 \in P$, observe that

$$f(p_1) + \mathbf{k}A\Delta^{\mathrm{T}} = f(p_1) + (f(p_2) - f(p_1)) = f(p_2). \qquad (8)$$

(i) If (p_1, p_2) is an arc in $G(P, E)$, defined by A-order, then according to (2) and (8), $f(p_1) < f(p_2)$.

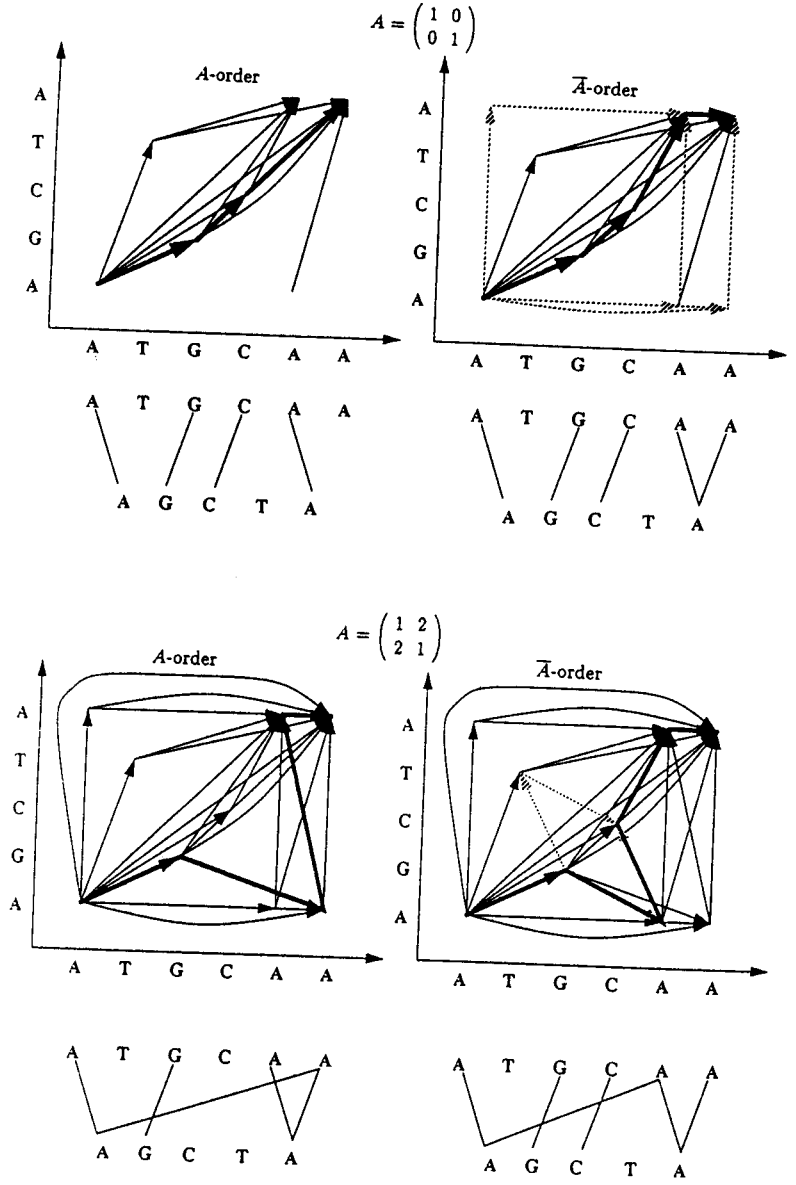

FIG. 4. Graphs $G(P, E)$ and optimal alignments for $s = ATGCA$, $t = AGCTA$ and for various A-orders and $\bar{A}$-orders. A-LCS are indicated by thick arcs; additional arcs introduced in $\bar{A}$-LCS are indicated by dotted arcs. Sequence alignments are shown below the graph $G(P, E)$.

(ii) If (p_1, p_2) is an arc in $G(P, E)$, defined by $\bar{A}$-order, then according to (3) and (8), $f(p_1) \le f(p_2)$ and $f(p_1) = f(p_2)$ iff $A\Delta^T = \mathbf{0}$. Since A is a non-singular matrix, $A\Delta^T = \mathbf{0}$ implies $p_1 = p_2$. Therefore $f(p_1) < f(p_2)$ for each arc (p_1, p_2). □

Arranging vertices allows implementation of the maximum path algorithm for acyclic graphs in $O(|E|)$ time and gives $O(n^2)$ running time for the LCS-problem. Unfortunately the Needleman–Wunsch transformation of $G(P, E)$ into the graph $G^*(I \times J, E^*)$ with $O(n^2)$ arcs is not valid for an arbitrary A-order. Below we describe a transformation of $G(P, E)$ that decreases the number of arcs significantly. The Needleman–Wunsch transformation is a particular case of this transformation.

6. Algorithms for A-LCS Problems

Consider the A-LCS problem and let L_1 and L_2 be the lines $a_{11}x + a_{12}y = 0$ and $a_{21}x + a_{22}y = 0$, respectively. If there is no integer point (x, y) on line L_k fulfilling the condition

$$|x| \le n, \qquad |y| \le m, \tag{9}$$

we slightly rotate L_k decreasing cone(A) until the "first" integer point of L_k fulfils condition (9), for $k \in \{1, 2\}$. After rotation (Fig. 5) we have lines L_1' and L_2' and cone$(A') \subset$ cone(A). Obviously A' and A define the same partial order on $I \times J$. Below we suppose that the first integer points $r = (i_1, j_1)$ and $s = (i_2, j_2)$ of L_1 and L_2 fulfil (9) and $A = \begin{pmatrix} j_1 & -i_1 \\ -j_2 & i_2 \end{pmatrix}$.

Let $v_1, v_2 \ldots, v_k$ be the set V of all non-zero integer vectors (or vertices) of the parallelogram Π (Fig. 5), defined by points $\mathbf{0}, r, s, r + s$. The number of elements of V, $\|V\|$, is given in the next proposition. The proof, which is straightforward, is omitted.

PROPOSITION 2. $\|V\| = |i_1 j_2 - i_2 j_1| + 2 = |\det(A)| + 2$.

Consider a graph $G^*(I \times J, E^*)$ with vertex set $I \times J$ and arc set E^* determined by V,

$$(p_1, p_2) \in E^* \Leftrightarrow (p_2 - p_1) \in V. \tag{10}$$

Define weighting functions w and $\bar{w}$ on E^* according to the rule:

$$w(p_1, p_2) = \begin{cases} w(p_2), & \text{if } p_2 \in P \text{ and } (p_2 - p_1) \ne r, s \\ 0, & \text{otherwise;} \end{cases} \tag{11}$$

$$\bar{w}(p_1, p_2) = \begin{cases} w(p_2), & \text{if } p_2 \in P \\ 0, & \text{otherwise.} \end{cases} \tag{12}$$

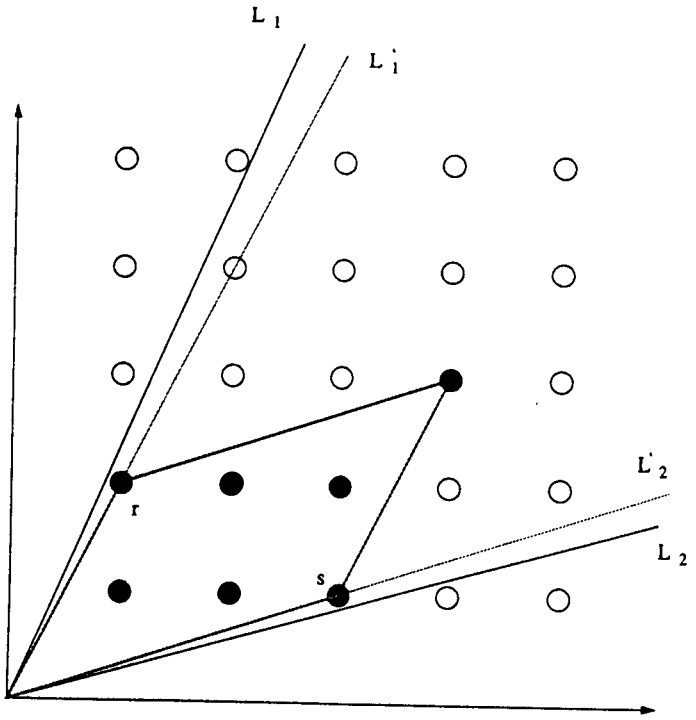

FIG. 5. Integer points on the lines L'_1 and L'_2 determine parallelogram Π and a set V of integer points in Π.

Theorems 2 and 3 below reduce A-LCS and $\bar{A}$-LCS problems to longest path problems.

THEOREM 2. *The length of an $\bar{A}$-LCS coincides with the length of a $\bar{w}$-longest path in G^*.*

Proof. Obviously, each path $p_1, p_2, \ldots, p_t$ in $G^*(I \times J, E^*, \bar{w})$ corresponds to an $\bar{A}$-sequence of the same length, since the vertices of this path with $w(p_k) > 0$ correspond to elements of an $\bar{A}$-sequence. To prove the theorem, it is sufficient to prove that each $\bar{A}$-sequence $p_1, p_2, \ldots, p_t$ has a corresponding path in the G^*-graph of at least the same length.

Let p_{k-1}, p_k be two arbitrary sequential elements of an $\bar{A}$-sequence. Since $\Delta = \Delta_k(p_{k-1}, p_k)$ belongs to cone($\bar{A}$), then

$$\Delta = xr + ys = \lfloor x \rfloor r + \lfloor y \rfloor s + \langle x \rangle r + \langle y \rangle s$$

($\lfloor x \rfloor$ is the integer part of x and $\langle x \rangle$ is the fractional part of x). The vector $p = \langle x \rangle r + \langle y \rangle s$ belongs to the parallelogram Π and is integer

($p = \Delta - \lfloor x \rfloor r + \lfloor y \rfloor s$). Therefore Δ is decomposed as a sum of $\lfloor x \rfloor + \lfloor y \rfloor + 1$ (or $\lfloor x \rfloor + \lfloor y \rfloor$ if $p = \mathbf{0}$) vectors defined by vertices from V. The decomposition for each pair p_{k-1}, p_k determines a path in G^* that visits vertices $p_1, p_2, \ldots, p_t$ and therefore has at least the same length as the $\bar{A}$-sequence $p_1 p_2 \ldots p_t$. □

THEOREM 3. *The length of an A-LCS coincides with the length of a w-longest path in G^*.*

Proof. Let $\mathscr{P} = p_1 p_2 \ldots p_t$ be an arbitrary path in G^*. Consider a subsequence of $\mathscr{P}$ defined by vertices: $\mathscr{P}' = \{p_k : p_k \in P, (p_k - p_{k-1}) \neq r, s\}$. Observe that $\mathscr{P}'$ is an A-sequence and according to (11) the length of this A-sequence coincides with the w-length of P:

$$\sum_{k=2}^{t} w(p_{k-1}, p_k) = \sum_{p_k \in \mathscr{P}'} w(p_{k-1}, p_k) = \sum_{p_k \in \mathscr{P}'} w(p_k).$$

To prove the theorem, it is sufficient to prove that each A-sequence $p_1, p_2, \ldots, p_t$ has a corresponding path in graph G^* with the same w-length. Let p_{k-1}, p_k be two arbitrary sequential elements of the A-sequence. As was proved in Theorem 2, $\Delta = \Delta_k(p_{k-1}, p_k) = \lfloor x \rfloor r + \lfloor y \rfloor s + p$. If $p \neq \mathbf{0}$, define $\mathscr{P}_k$ to be the path consisting of $\lfloor x \rfloor$ arcs r, $\lfloor y \rfloor$ arcs s, and ending with p. According to (11) only the last arc of this path has positive weight, equal to $w(p_k)$. If $p = \mathbf{0}$, then $\lfloor x \rfloor > 0$, $\lfloor y \rfloor > 0$, since otherwise p_{k-1} and p_k would be incomparable. Let $\mathscr{P}_k$ be the path consisting of $\lfloor x \rfloor - 1$ arcs r, $\lfloor y \rfloor - 1$ arcs s, and the arc $r + s$ at the end. According to (11) only the last arc of this path has positive weight, equal to $w(p_k)$. Thus each pair p_{k-1}, p_k determines a path $\mathscr{P}_k$ in $G^*(I \times J, E^*, w)$, and only the last arc of this path has positive weight $w(p_k)$. Therefore the length of the path $\mathscr{P}_1, \mathscr{P}_2, \ldots, \mathscr{P}_t$ equals the length of the A-sequence $p_1, p_2, \ldots, p_t$ □

7. A-LCS PROBLEMS AND HILBERT BASES

According to Theorems 2 and 3, finding longest A and $\bar{A}$ sequences requires about kn^2 operations, where k is the maximum vertex degree in G^*. ($k = |\det(A)| + 2$ equals the number of integer points in Π minus 1, by Proposition 2.) For the classical LCS problem, and for the longest CF, CIS, CIF problems, $\det(A) = 1$ and $k = 3$ (Figs. 6a, b) as in the usual Needleman–Wunsch algorithm. For CF and CIF, k can be reduced to two, where the vertex $(1, 1)$ of Π is decomposed as $(1, 0) + (0, 1)$. For $A = \begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix}$ and $A = \begin{pmatrix} -1 & 2 \\ 2 & -1 \end{pmatrix}$, $\det(A) = -3$ and $k = 5$ (Figs. 6c, d), but we

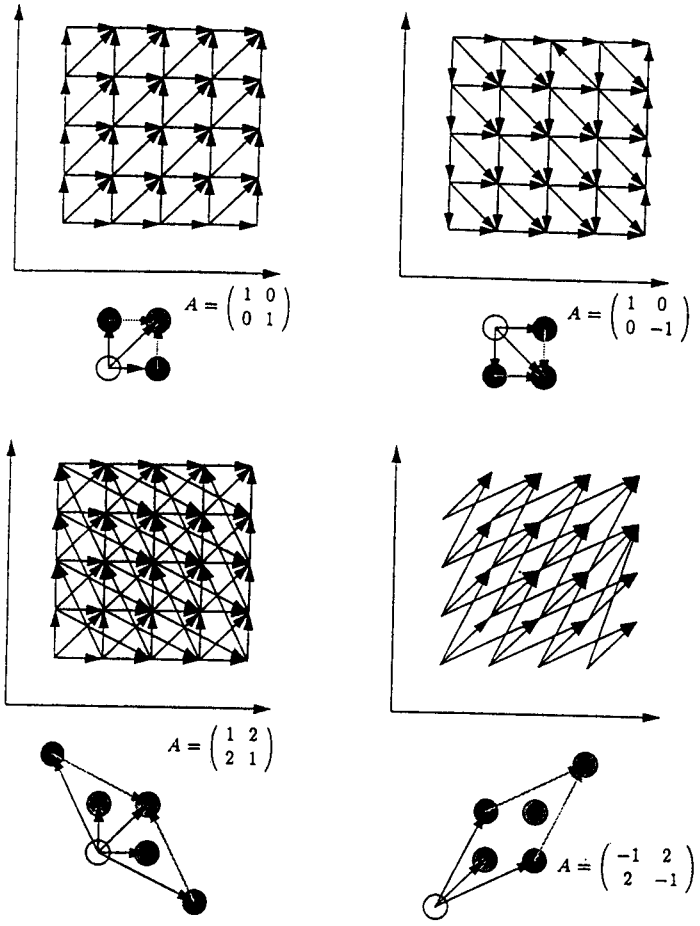

FIG. 6. Examples of parallelograms Π for various matrices A and the corresponding graphs G^*. The dark nodes in Π correspond to r and s.

can further decrease k as some points in Π are non-negative integer combinations of others.

We describe a procedure to eliminate arcs in the graphs G^* for A- and $\bar{A}$-LCS problems. A set H of integer vectors in the cone $K = \{x: Ax \geq \mathbf{0}\}$ is called a *Hilbert basis* of K if each integer vector in K is a non-negative integer linear combination of vectors in H. Let H be a Hilbert basis of the cone $\{x: Ax \geq \mathbf{0}\}$, and let $H(A)$ be the intersection of H and Π, where Π is the parallelogram defined by A. Note that each integer vector in Π is a non-negative integer linear combination of vectors from $H(A)$.

Observe that Theorem 2 still holds even if we define E^* by

$$(p_1, p_2) \in E^* \Leftrightarrow (p_2 - p_1) \in H(A)$$

instead of by (10). Similarly Theorem 3 still holds even if we define E^* by

$$(p_1, p_2) \in E^* \Leftrightarrow (p_2 - p_1) \in H(A) \cup \{r + s\}$$

instead of by (10). These observations allow further transformations of G^* excluding arcs which do not belong to the Hilbert basis.

For example, when $A = \begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix}$, $H(A) = \{(2, -1), (1, 0), (0, 1), (-1, 2)\}$. (The vertex (1, 1) from Π equals (1, 0) + (0, 1).) For $A = \begin{pmatrix} -1 & 2 \\ 2 & -1 \end{pmatrix}$, $H(A) = \{(1, 2), (1, 1), (2, 1)\}$. Therefore we can reduce the maximum degree of G^* for these matrices to four and three, respectively. For matrix $A = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$ $k = 2$, but $H(A) = \{(1, 1)\}$, reducing the maximum degree of G^* to one.

Sometimes we can reduce the maximum degree of G^* even if there exist $O(n^2)$ integer points in Π. For example, let n be even and

$$A = \begin{pmatrix} -(n/2 - 1) & n - 1 \\ n - 1 & -(n/2 - 1) \end{pmatrix}.$$

For this matrix, Π contains $|\det(A)| + 3 = 0.75n^2 - n + 3$ vertices by Proposition 2 but $H(A) = \{(n/2 - 1, n - 1), (1, 2), (1, 1), (2, 1), (n - 1, n/2 - 1)\}$ with only five vertices (Fig. 7).

Unfortunately we cannot guarantee that $H(A)$ contains $O(1)$ integer points for an arbitrary A-matrix. When $A = \begin{pmatrix} n & -1 \\ -1 & m \end{pmatrix}$, the parallelogram Π contains almost all vertices from $I \times J$; $|\det(A)| + 3 = nm + 2$ and $H(A) = \{(1, 1), (1, 2), \ldots, (1, n)\} \cup \{(2, 1), (3, 2), \ldots, (m, 1)\}$ contains $n + m - 1$ vectors.

Some interesting questions arise:

1. Given A, find $H(A)$ of minimal size. (This is the minimal Hilbert basis for a plane lattice generated by vectors r and s.)

2. Find the maximum size of a minimal Hilbert basis $H(A)$ for matrix $A = (a_{ij})$ with $|a_{ij}| \leq n$.

Fortunately using the Jeroslav–Schrijver characterization of Hilbert bases, we can find $H(A)$ in $O(n)$ time and prove that the maximum size of minimal Hilbert basis of $H(A)$ is $O(n)$.

THEOREM 4 [J78], [S81]. *Let A be an integer matrix. If $K = \{x: Ax \geq 0\}$ has the property that $x \neq 0$ and $x \in K$ implies $-x \notin K$, then the intersection of a family of Hilbert bases is a Hilbert basis. (This intersection is called the minimal Hilbert basis.)*

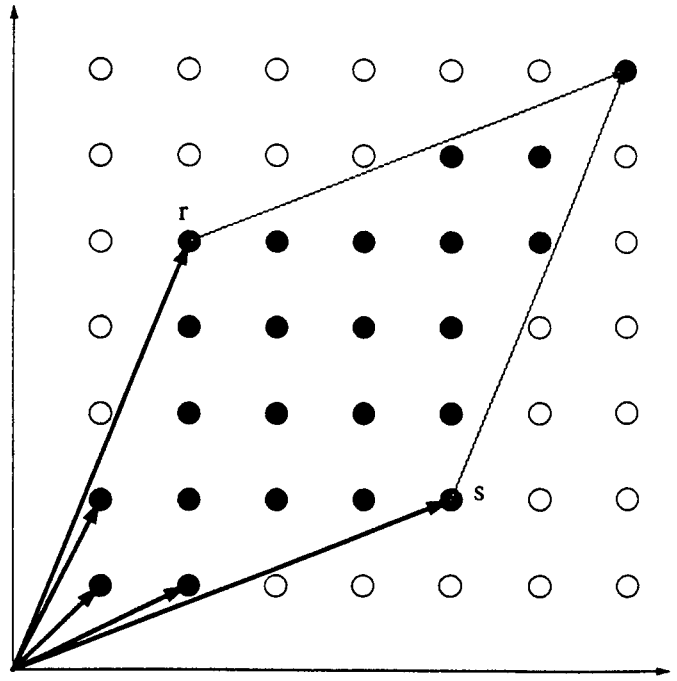

FIG. 7. For matrix $A = \begin{pmatrix} -(n/2-1) & n-1 \\ n-1 & -(n/2-1) \end{pmatrix}$ with $n = 6$ parallelogram Π contains $|\det(A)| + 3 = 0.75n^2 - n + 3 = 24$ vertices but the minimal Hilbert basis contains only five vertices.

Theorem 4 implies the following observation of Schrijver [S81].

COROLLARY 1. *A set of all integer vectors in K that are not non-negative integer linear combinations of other integer vectors in K is the minimal Hilbert basis in K.*

The corollary implies a sufficient condition for a point (i, j) to lie outside the minimal Hilbert basis of the plane lattice Π: if all points in one-neighbourhood of (i, j) (the set $\{(i', j'): |i' - i| \leq 1, |j' - j| \leq 1\}$) belong to Π, then (i, j) does not belong to the minimal Hilbert basis of Π. Therefore (i, j) can belong to the minimal Hilbert basis only if its one-neighbourhood intersects the boundary of Π. This implies that cardinality of the minimal Hilbert basis of Π is at most $O(n + m)$ and that it is easy to find the minimal Hilbert basis in $O(n)$ time. Therefore the number of arcs in G can be reduced to $O(n^3)$, and this yields $O(n^3)$ A- and $\bar{A}$-LCS algorithms for an arbitrary A-matrix.

The LCS problem is often discussed in terms of the two elementary edit operations, insertions and deletions. Generalized LCS problems require at least $|H(A)|$ elementary edit operations; each operation corresponds to a vector from the Hilbert basis. It is worth noting that, although the number of elementary edit operations can be as large as $O(n)$, an integer analog of the Caratheodory theorem [CFS86, S91] implies that each arc between comparable elements can be decomposed as a sum of only four elementary edit operations.

It is worth noting that for arbitrary 2×2 non-singular matrices A and B each A-LCS problem can be reduced to a B-LCS by the transformation $\begin{pmatrix} i' \\ j' \end{pmatrix} = B^{-1} \cdot A \begin{pmatrix} i \\ j \end{pmatrix}$. For arbitrary A and $B = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ this transformation reduces the A-LCS problem for n-letter words to the classical LCS problem for $O(n^2)$ words and yields a $O(n^4)$ algorithm for the A-LCS problem. Nevertheless, the Hunt–Szymanski algorithm applied after such transformation yields an A-LCS algorithm with running time at most $O(n^2 \log n)$.

8. GENERALIZED ALIGNMENTS

Most existing methods for DNA or protein sequence comparison treat more complex weighting functions than Eq. (1). For a 2×2 matrix A, we define *A-maximum alignment* of words $\mathbf{q} = q_1 q_2 \dots q_n$ and $\mathbf{t} = t_1 t_2 \dots t_m$ as an A-sequence $p_1 p_2 \dots p_t$ maximizing a function

$$\sum_{i=1}^{t} w_s(p_i) - \sum_{i=2}^{t} w_g(p_{i-1}, p_i) - w_{in}(p_1) - w_{ter}(p_t), \qquad (13)$$

where

- $w_s(p) = w_s(i, j)$ is the *substitution* weight of aligning q_i and t_j;
- $w_g(p', p'') = w_g((i', j'), (i'', j''))$ is the weight of a *gap* defined as an insertion/deletion between (i', j') and (i'', j'');
- $w_{in}(p) = w_{in}(i, j)$ is the weight of an *initial gap* (for the usual alignment problem, this is a gap between $(0, 0)$ and (i, j));
- $w_{ter}(p) = w_{ter}(i, j)$ is the weight of a *terminal gap* (for the usual alignment problem, this is a gap between (i, j) and $(n + 1, m + 1)$).

$\bar{A}$-maximum alignment is defined similarly. By initial and terminal gap functions, we mean the gap functions at the leftmost and rightmost ends of the sequence alignment. These have often been weighted differently from gaps in the remaining alignment.

Let r and s be the vertices of the parallelogram Π (Fig. 5) defined by matrix A, and V be the set of all integer points of Π, except $\mathbf{0}$. For an arbitrary vector $p \in \text{cone}(A)$, let $p = x_r(p)r + x_s(p)s$ be the decomposi-

tion of p into vectors r and s. Define

$$\|x\| = \begin{cases} x - 1, & \text{if } x \text{ is integer and } x \neq 0, \\ \lfloor x \rfloor, & \text{otherwise.} \end{cases} \quad (14)$$

For the usual alignment problem, $\|x_r\|$ and $\|x_s\|$ are treated as the gap lengths in **q** and **t** correspondingly.

Next we consider a few special cases of the functions $w_s, w_g, w_{in}, w_{ter}$:

1. Substitution weights w_s are usually defined by an $l \times l$ matrix $(d(a, b))$, where l is the size of alphabet, by the rule $w_s(p) = w_s(i, j) = d(q_i, t_j)$. Some common examples are

(a)

$$d(a, b) = \begin{cases} 1, & a = b \\ 0, & a \neq b, \end{cases}$$

and

(b) $d(a, b)$ is an arbitrary matrix.

A few matrices for protein sequence comparison are discussed in [A91].

2. Gap weights $w_g(p', p'')$ are defined as functions of variables $\|x_r\| = \|x_r(p'' - p')\|$ and $\|x_s\| = \|x_s(p'' - p')\|$. For the usual alignment problems, the sequences $p_1 p_2 \ldots p_t$ giving a maximum in (13) usually have the property that $\|x_r\| = 0$ or $\|x_s\| = 0$. We suppose that the gap weight is a sum of the same gap function for $\|x_r\|$ and $\|x_s\|$:

(a) additive gap functions $w_g(p', p'') = v \cdot \|x_r\| + v \cdot \|x_s\|$, where v is an arbitrary constant.

(b) linear gap functions $w_g(p', p'') = (u + v \cdot \|x_r\|) + (u + v \cdot \|x_s\|)$, where u and v are arbitrary constants. If $\|x_r\| = 0$ or $\|x_s\| = 0$, the corresponding term in the sum is eliminated.

(c) the sum of piecewise linear concave gap functions of $\|x_r(p'' - p')\|$ and of $\|x_s(p'' - p')\|$.

3. Initial (terminal) gap weights w_{in} (w_{ter}):

(a) $w_{in}(p) = w_g((0, 0), p)$; $w_{ter}(p) = w_g(p, (n + 1, m + 1))$.

(b)

$$w_{in}(p) = w_{in}(i, j) = \begin{cases} 0, & i = 0 \text{ or } j = 0, \\ \infty, & \text{otherwise,} \end{cases}$$

$$w_{ter}(p) = w_{ter}(i, j) = \begin{cases} 0, & i = n \text{ and } j = m, \\ \infty, & \text{otherwise.} \end{cases}$$

(c) $w_{in}(p) = w_{ter}(p) = 0$.

(d) $w_{in}(p)$ and $w_{ter}(p)$ are arbitrary functions.

For $A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, particular cases of problem (13) are studied in many papers, a few of which are listed below:

- w_s (1a), w_g (2a), w_{in}, w_{ter} (3a). Evolutionary (edit) distance problem [S74, FW74] and classical LCS problem (for $v = 0$).
- w_s (1b), w_g (2a), w_{in}, w_{ter} (3b). Classical global optimal alignment problem [NW70]
- w_s (1b), w_g (2a), w_{in}, w_{ter} (3c). Local alignment with additive gap function [S79]
- w_s (1b), w_g (2b), w_{in}, w_{ter} (3a). Global alignment with linear gap function [FS83]
- w_s (1b), w_g (2b), w_{in}, w_{ter} (3c). Local alignment with linear gap function [SW81, G82, AE86].
- w_s (1b), w_g (2c), w_{in}, w_{ter} (3b). Global alignment with piecewise linear gap function [WSB76].
- w_s (1b), w_g (2c), w_{in}, w_{ter} (3c). Local alignment with piecewise linear gap function [SW81, W84, MM88, GG89, G90].
- w_s (1b), w_g (2c), w_{in}, w_{ter} (3d). Global alignment with piecewise linear gap function and arbitrary beginning/end gap function [R84].

A few remarks about the relationships between these papers are in order. [NW70] presents a similarity method while [S74, FW74] present distance methods. For global alignment there is an easy correspondence between these algorithms [SWF81]. For local alignment, [S79] is based on distance while [SW81] uses similarity. Here no equivalence between similarity and distance exists, and similarity is usually used for local alignment. Finally we mention that we have included various work under "piecewise linear concave gap functions." We refer the reader to [MM88, GG89], where $O(n^2 \cdot \log K)$ algorithms are presented for concave K-piecewise gap functions.

Note that the problem of finding $\bar{A}$-maximum alignment for $A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ with an objective function similar to (13) was studied in speech processing [SC78], geology [WR87], and DNA physical mapping [WSK84, MBR91, HW92].

In the next section we prove that all these examples of optimal alignment problems are particular cases of Theorem 5 that give $O(Kn^2)$ algorithms for optimal alignment problems with arbitrary K-piecewise linear concave function w_g and arbitrary initial/terminal gap functions w_{in} and w_{ter}.

9. Generalized Alignments and K-Strata Graphs

The goal of this and the next section is to give an A-alignment algorithm for generalized alignment. The situation here is much more difficult than the A-LCS problem due to concave gap functions, arbitrary initial/terminal gap functions, and local or global alignments. Instead of treating the alignment problems from Section 8 in a case-by-case fashion, we would like to give a general optimum path algorithm. In this section we define K-strata graphs and in the next section we derive a general optimum path algorithm.

Let w_g be a piecewise linear concave non-negative function with K pieces defined on $\mathbf{R}^+$ (Fig. 8a):

$$w_g(y) = w_g(y_k) + u_k(y - y_k) \qquad \text{if } y_k \le y < y_{k+1}, \qquad 1 \le k \le K$$

(here $y_{K+1} = \infty$). Since w_g is concave and non-negative,

$$\forall z > y_k: \qquad w_g(z) \le w_g(y_k) + u_k(z - y_k), \tag{15}$$

and

$$\forall z_1, z_2, \ldots z_l: \qquad w_g(z_1 + z_2 + \ldots + z_l) \le w_g(z_1) + w_g(z_2) + \ldots + w_g(z_l). \tag{16}$$

Next we specify a graph for A-alignment, where the edges of the graph have a clear interpretation in terms of the pieces of w_g. The vertices of the graph will be represented by *strata* $1, 2, \ldots, K$, each stratum having two *layers*, r and s. Layer r of the ith stratum corresponds to the gaps between p' and p'' with $\|x_r\|$ fulfilling the condition $y_i \le \|x_r\| \le y_{i+1}$. Layer s of the ith stratum corresponds to gaps with $\|x_s\|$ fulfilling the condition $y_i \le \|x_s\| \le y_{i+1}$.

Formally, consider the K-stratum graph $G^K(W, E, w)$ (Fig. 9) with vertex set

$$W = I \times J \times \{0, 1, \ldots, K\} \times \{r, s\} \cup \{p_{\text{in}}, p_{\text{ter}}\}.$$

We designate the vertices of W by triples (p, i, a), where $p \in I \times J$, $i \in \{0, 1, \ldots, K\}$, $a \in \{r, s\}$. p_{in} and p_{ter} correspond to initial and terminal gaps, stratum zero corresponds to substitutions, stratum $i > 0$ corresponds to gaps between p' and p'' fulfilling

$$\|x_r(p'' - p')\| \ge y_i \qquad \text{(layer } r\text{)}$$

or

$$\|x_s(p'' - p')\| \ge y_i \qquad \text{(layer } s\text{)}.$$

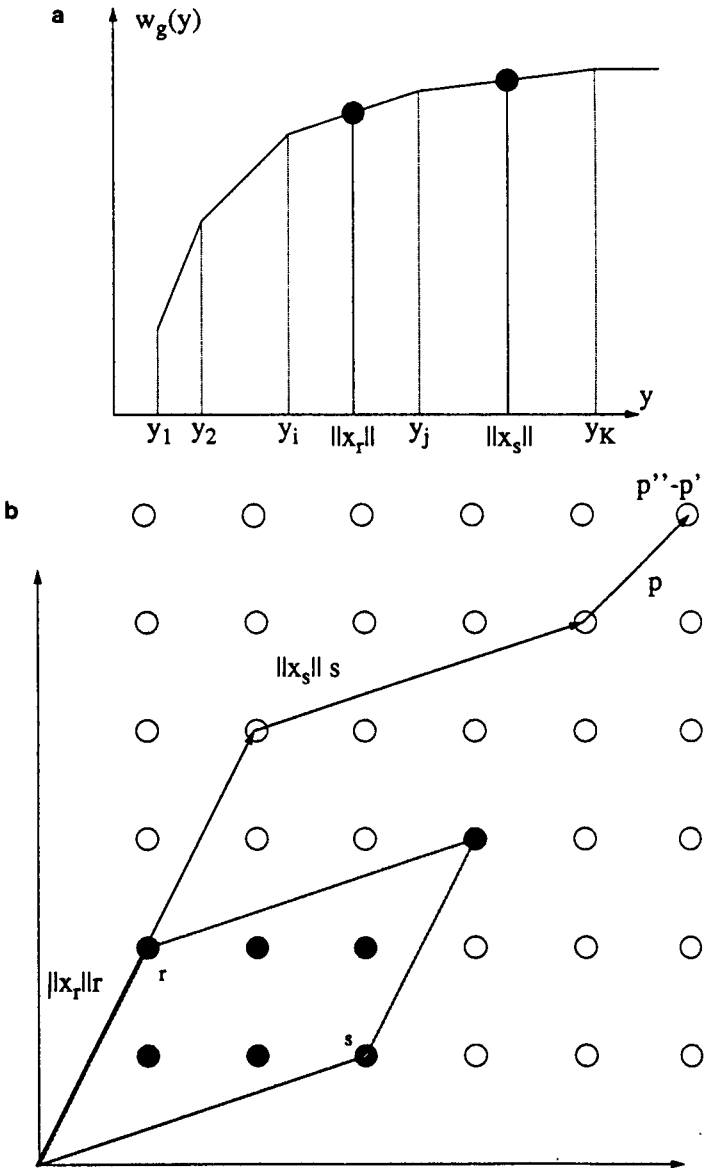

FIG. 8. (a) A piecewise linear concave function w_g; (b) the decomposition $p'' - p' = \|x_r\|r + \|x_s\|s + p$, where p belongs the parallelogram Π.

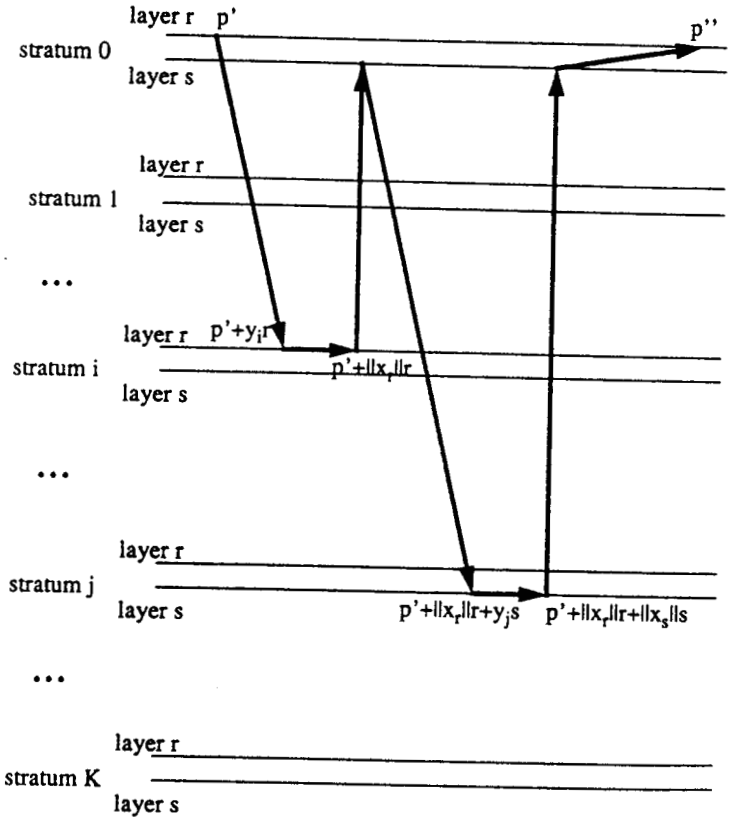

FIG. 9. The gap between p' and p'' corresponds to the path between $(p', 0, r)$ and $(p'', 0, r)$ in G^K consisting of seven arcs. Each horizontal line represents a copy of the set $I \times J$.

Each stratum has two layers, r and s, corresponding to r and s components of the gaps.

A gap in the A-alignment between p' and p'' can be decomposed as $p'' - p' = \|x_r\|r + \|x_s\|s + p$ (Fig. 8b), where p belongs to the parallelogram Π. (See the proof of Theorem 3.) This gap can be represented in the graph as a path between $(p', 0, r)$ and $(p'', 0, r)$ in G^K, consisting of seven arcs and passing through ith and jth strata (Fig. 9). Indices i and j are determined by the conditions (Fig. 8a)

$$y_i \le \|x_r\| \le y_{i+1}$$

and

$$y_j \le \|x_s\| \le y_{j+1}.$$

The first three arcs below give the weight of the r-component of the gap and the next three arcs give the weight of the s-component. The last arc gives the weight of the substitution p'':

- $(p', 0, r) \to (p' + y_i r, i, r)$ (r-component of the gap, beginning)
- $(p' + y_i r, i, r) \to (p' + \|x_r\|r, i, r)$ (r-component of the gap, continuing)
- $(p' + \|x_r\|r, i, r) \to (p' + \|x_r\|r, 0, s)$ (r-component of the gap, ending)
- $(p' + \|x_r\|r, 0, s) \to (p' + \|x_r\|r + y_j s, j, s)$ (s-component of the gap, beginning)
- $(p' + \|x_r\|r + y_j s, j, s) \to (p' + \|x_r\|r + \|x_s\|s, j, s)$ (s-component of the gap, continuing)
- $(p' + \|x_r\|r + \|x_s\|s, j, s) \to (p' + \|x_r\|r + \|x_s\|s, 0, s)$ (s-component of the gap, ending)
- $(p' + \|x_r\|r + \|x_s\|s, 0, s) \to (p'', 0, r)$ (substitution p'').

Formally the arc set of G^K is defined by the rules:

- Arcs inside zeroth stratum (substitution arcs): $a = ((p_1, 0, s), (p_2, 0, r))$,

$$a \in E \text{ and } w(a) = w_s(p_2) \Leftrightarrow (p_2 - p_1) \in V \text{ and } (p_2 - p_1) \ne r, s.$$

- Arcs inside zeroth stratum (bridges between r and s layers of the zero-stratum; $\|x_r\| = 0$): $a = ((p, 0, r), (p, 0, s))$,

$$a \in E \text{ and } w(a) = 0.$$

- Arcs from zero-stratum to kth stratum for $k > 0$ (beginning gap):

$a = ((p_1, 0, r), (p_2, k, r))$: $a \in E$ and $w(a) = -w_g(y_k) \Leftrightarrow (p_2 - p_1) = y_k r$,

or

$a = ((p_1, 0, s), (p_2, k, s))$: $a \in E$ and $w(a) = -w_g(y_k) \Leftrightarrow (p_2 - p_1) = y_k s$.

• Arcs inside kth stratum for $k > 0$ (continuing gap):

$a = ((p_1, k, r), (p_2, k, r))$: $a \in E$ and $w(a) = -u_k \Leftrightarrow (p_2 - p_1) = r$,
or
$a = ((p_1, k, s), (p_2, k, s))$: $a \in E$ and $w(a) = -u_k \Leftrightarrow (p_2 - p_1) = s$.

• Arcs from k-stratum to zeroth stratum for $k > 0$ (ending gap):

$$a = ((p, k, r), (p, 0, s)),\ a \in E \text{ and } w(a) = 0,$$

or

$$a = ((p, k, s), (p, 0, s)): a \in E \text{ and } w(a) = 0.$$

• Arcs from p_{in} to zeroth stratum (initial gap): $a = (p_{in}, (p, 0, r))$,

$$a \in E \text{ and } w(a) = -w_{in}(p) + w_s(p).$$

• Arcs from zeroth stratum to p_{ter} (terminal gap): $a = ((p, 0, r), p_{ter})$,

$$a \in E \text{ and } w(a) = -w_{ter}(p).$$

10. Algorithms for the A-Maximum Alignment Problems

First we prove a few lemmas about paths in G^K beginning and ending in the zeroth stratum. Then Theorem 5 gives the general alignment algorithm. Let $p', p'' \in I \times J$ and let Q be an arbitrary path of length $w(Q)$ in G^K between $(p', 0, r)$ and $(p'', 0, s)$ or between $(p', 0, s)$ and $(p'', 0, s)$.

Lemma 6. *Let $p'' - p' = zr$ (or $p'' - p' = zs$) and assume Q has no vertices in zeroth stratum, except the first and the last ones. Then $w_g(z) \le -w(Q)$.*

Proof. All intermediate vertices of Q belong to a kth stratum for k fulfilling the conditions $1 \le k \le K - 1$ and $y_k \le z$. (See the definition of arcs in G^K.) Observe that each arc from the zeroth stratum to the kth stratum has weight $-w_g(y_k)$ and that each arc inside the kth stratum has weight $-u_k$. Taking into account (15),

$$w_g(z) \le w_g(y_k) + (z - y_k)u_k = -w(Q). \quad \square$$

Lemma 7. *Let $p'' - p' = zr$ (or $p'' - p' = zs$) and assume Q has no arcs from layer s to layer r inside the zeroth stratum. Then $w_g(z) \le -w(Q)$.*

Proof. Let $(p', 0) = (p_1, 0), (p_2, 0), \ldots, (p_l, 0) = (p'', 0)$ be the vertices of Q belonging to the zeroth stratum, where the indices for layers r and s are omitted, and let Q_i be the subpath of Q between $(p_{i-1}, 0)$ and $(p_i, 0)$ for $i = 2, 3, \ldots, l$. Observe that $p_i - p_{i-1} = z_i r$ (or $p_i - p_{i-1} = z_i s$) for integer z_i, and that Q_i fulfils the conditions of Lemma 6. Therefore $w_g(z_i) \le -w(Q_i)$. On the other hand, according to (16),

$$\begin{aligned} w_g(z_2 + \ldots + z_l) &\le w_g(z_2) + \ldots + w_g(z_l) \\ &\le -w(Q_1) - \ldots - w(Q_{l-1}) = -w(Q). \quad \square \end{aligned}$$

Lemma 7 implies

Lemma 8. *Let Q have no arcs from layer s to layer r inside the zeroth strata. Then*

$$w_g(\| x_r(p'' - p') \|) + w_g(\| x_r(p'' - p') \|) \le -w(Q).$$

The main result of this section is given in the next theorem.

Theorem 5. *The length of an A-maximum alignment coincides with the length of a w-longest path from p_{in} to p_{ter} in the K-strata graph G^K.*

Proof. Let $Q = \{p_{in}, q_1 = (p_1, k_1), q_2 = (p_2, k_2), \ldots, q_t = (p_t, k_t), p_{ter}\}$ be an arbitrary path of length $w(Q)$ between p_{in} and p_{ter} in G^K. Again the indices for layers r and s are omitted. We denote by $Q(p_i, p_j)$ the subpath of Q between (p_i, k_i) and (p_j, k_j).

We now prove that there exists an A-sequence with alignment score (13) equal at least $w(Q)$. Consider the projection $\mathscr{P} = p_1 p_2 \ldots p_t$ of Q onto $I \times J$. For consecutive elements of $\mathscr{P}$ either $p_i - p_{i-1} = 0$ for arcs from the kth stratum to the zeroth stratum, or $p_i - p_{i-1} = zs$ or zr for arcs to or inside the kth stratum, or $p_i - p_{i-1} \in V$ for substitution arcs inside the zeroth stratum. Therefore $\mathscr{P}$ is an $\bar{A}$-sequence and $p_j - p_i \in \text{cone}(\bar{A})$ for each $1 \le i \le j \le t$. Moreover, $p_j - p_i \in \text{cone}(A)$ if a subpath of Q between p_i and p_j contains a substitution arc from the zeroth stratum. (See the definition of arcs inside the zeroth stratum.) Therefore the subsequence of $\mathscr{P}$,

$$\mathscr{P}' = \{p_1\} \cup \{p_i: ((p_{i-1}, k_{i-1}), (p_i, k_i)) \text{ is a substitution arc in the zero-stratum}\} \cup \{p_t\}$$

is an A-path. For each pair of consecutive elements p_i and p_j in $\mathscr{P}'$, the last arc of $Q(p_i, p_j)$ is a substitution arc in the zeroth stratum. Set

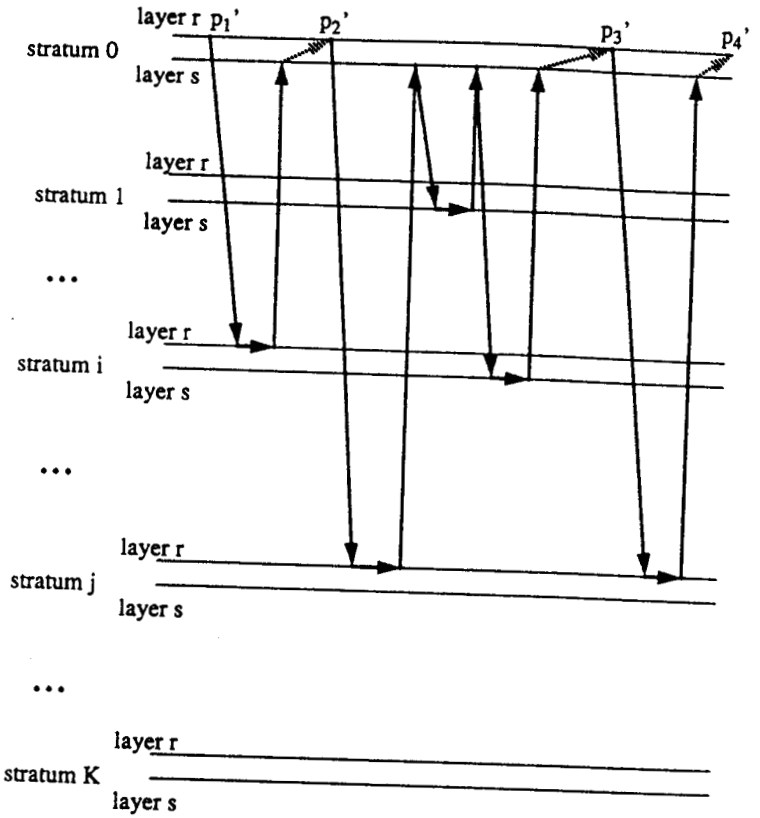

FIG. 10. A path Q in the K-strata graph with the vertices p_{in} and p_{ter} omitted. The substitution arcs of Q inside the zeroth stratum (dashed line) define a sequence $\mathscr{P}' = (p_1', p_2', p_3', p_4')$. The subpaths $Q_2 = Q(p_1', p_2')$, $Q_3 = Q(p_2', p_3')$, and $Q_4 = Q(p_3', p_4')$ fulfill the conditions of Lemma 8.

$\mathscr{P}' = p_1' p_2' \dots p_l'$, $l \le t$, and $\forall i$: $p_i' = p_j$ for a $j \ge i$, and let $Q_i = Q(p_{i-1}', p_i')$ be a subpath of Q joining p_{i-1}' and p_i' (Fig. 10).

Observe that only the last arc e_i of the path Q_i belongs to the zeroth stratum and goes from layer s to layer r. Therefore $Q_i \setminus e_i$ fulfils the conditions of Lemma 8. Thus

$$w_g(z_r) + w_g(z_s) \le -w(Q_i \setminus e_i),$$

where $z_r = \|x_r(p_i' - p_{i-1}')\|$, $z_s = \|x_s(p_i' - p_{i-1}')\|$. Since $w(e_i) = w_s(p_i')$ and $w_g(p_i' - p_{i-1}') = w_g(z_r) + w_g(z_s)$, we have

$$w_g(p_i' - p_{i-1}') - w_s(p_i') \le -w(Q_i) = -w(Q_i \setminus e_i) - w(e_i).$$

Therefore the alignment score (13) for the A-sequence $\mathscr{P}$ has score at least $w(Q)$:

$$\begin{aligned} w(Q) &= w(p_{in}, p_1) + \sum_{i=2}^{l} w(Q_i) + w(p_t, p_{ter}) \\ &\le (-w_{in}(p_1') + w_s(p_1')) + \sum_{i=2}^{l} (w_s(p_i') - w_g(p_{i-1}', p_i')) \\ &\quad - w_{ter}(p_l'). \end{aligned}$$

To complete the proof of the theorem it is sufficient to construct for each A-sequence $\mathscr{P} = p_1 p_2 \dots p_t$ a corresponding path in G^K between p_{in} and p_{ter} with length equal to the alignment score of $\mathscr{P}$.

Following the proof of Theorem 3, denote

$$p_k - p_{k-1} = x_r r + x_s s = \|x_r\| r + \|x_s\| s + p.$$

Observe that $p \in V$ and let $y_i \le \|x_r\| < y_{i+1}$, $y_j \le \|x_s\| < y_{j+1}$ (Fig. 8b). It follows from the definition of arcs in G^K that the sequence of vertices (Fig. 9),

$$\begin{aligned} &(p_{k-1}, 0, r), \\ &\left.\begin{aligned} &(p_{k-1} + y_i r, i, r), \\ &(p_{k-1} + (y_i + 1) r, i, r), \\ &\dots, \\ &(p_{k-1} + \|x_r\| r, i, r), \end{aligned}\right\} \text{stratum } k \\ &(p_{k-1} + \|x_r\| r, 0, s), \\ &\left.\begin{aligned} &(p_{k-1} + \|x_r\| r + y_j s, j, s), \\ &(p_{k-1} + \|x_r\| r + (y_j + 1) s, j, s), \\ &\dots, \\ &(p_{k-1} + \|x_r\| r + \|x_s\| s, j, s), \end{aligned}\right\} \text{stratum } l \\ &(p_{k-1} + \|x_r\| r + \|x_s\| s, 0, s), \\ &(p_{k-1} + \|x_r\| r + \|x_s\| s + p, 0, r) = (p_k, 0, r), \end{aligned}$$

is a path Q_i between $(p_{k-1}, 0, r)$ and $(p_k, 0, r)$ in G^K having the length $w(Q_k) = -w_g(p_{k-1}, p_k) + w_s(p_k)$. Therefore, the length of the path $Q =$

$p_{in}, Q_2, Q_3, \ldots, Q_t, p_{ter}$ in G^K equals the weight of the alignment score (13) of $\mathscr{P}$. □

Remark 1. The time complexity of the algorithm is determined by the number of arcs in the K-stratum graph and equal $O((K + \|\det(A)\|)n^2)$. Using Hilbert bases the time complexity can be reduced to $O((K + \|H(A)\|)n^2)$. The transformation $\begin{pmatrix} i' \\ j' \end{pmatrix} = A\begin{pmatrix} i \\ j \end{pmatrix}$ reduces the A-LCS alignment problem to the classical LCS problem and allows one to use the algorithms from [MM88, GG89].

Remark 2. We omitted consideration of functions w_g depending on context, i.e., functions not only the size of a gap but of the letters in a gap as well [S74, R84]. These models can be incorporated in our approach.

Remark 3. We assumed that the gap function is a sum of the same gap function for $\|x_r(p'' - p')\|$ and $\|x_s(p'' - p')\|$. The case of

- distinct and context dependent gap functions that arise for alignment of a protein with unknown tertiary structure with the protein with known tertiary structure, see [BSST87, SS92];
- the functions of the variables $(\|x_r(p'' - p')\| + \|x_s(p'' - p')\|)$ that arise for RNA secondary structure;
- the functions of $(\|x_r(p'' - p')\| - \|x_s(p'' - p')\|)$ that arise for DNA physical mapping can be incorporated in our approach.

Acknowledgments

The research was supported in part by the National Science Foundation (DMS 90-05833) and the National Institute of Health (GM-36230). We are grateful to Dan Gusfield, Gian-Carlo Rota, Anatoly Rubinov, and Martin Vingron for helpful discussions as well as Andras Sebo and Alexander Vainshtein for useful comments on Hilbert bases.

References

[A90] A. V. Aho, Algorithms for finding patterns in strings, *in* "Handbook of Theoretical Computer Science" (J. van Leeuwen, Ed.), pp. 256–300, Elsevier Science, Amsterdam, 1990.

[A91] S. Altschul, Amino acid substitution matrices from an information theoretic perspective, *J. Mol. Biol.* **219** (1991), 555–565.

[AE86] S. F. Altschul and B. W. Erickson, Optimal sequence alignment using affine gap costs, *Bull. Math. Biol.* **48** (1986), 603–616.

[A86] A. Apostolico, Improving the worst-case performance of the Hunt–Szymanski strategy for the longest common subsequence of two strings, *Inform. Process. Lett.* **23** (1986), 63–69.

[AG87] A. Apostolico and C. Guerra, The longest common subsequence problem revisited, *Algorithmica* **2** (1987), 315–336.

[BSST87] T. L. Blundell, B. L. Sibanda, M. J. Sternberg, and J. M. Thornton, Knowledge-based prediction of protein structures and the design of novel molecules, *Nature* **326** (1987), 347–352.

[CPHW91] E. T. Chow, T. Hunkapiller, J. C. Peterson, B. A. Zimmerman, and M. S. Waterman, A systolic array processor for biological information signal processing, *in* "Proceedings, International Conference on Supercomputing (ICS-91), June 17–21, 1991," to appear.

[CFS86] W. Cook, J. Fonlupt, and A. Schrijver, An integer analogue of Caratheodory's theorem, *J. Combin. Theory Ser. B* **40** (1986), 63–70

[D59] E. W. Dijkstra, A note on two problems in connection with graphs, *Numer. Math.* **1** (1959), 269–271.

[D50] R. P. Dilworth, A decomposition theorem for partially ordered sets, *Ann. Math.* **51** (1950), 161–165.

[DM41] B. Dushnik, E. W. Miller, Partially ordered sets, *Am. J. Math.* **63** (1941), 600–610.

[EGGI90] D. Eppstein, Z. Galil, R. Giancarlo, and G. F. Italiano, Sparse dynamic programming, *in* "Proceedings, First ACM-SIAM SODA," pp. 513–522.

[EGGI91] D. Eppstein, Z. Galil, R. Giancarlo, and G. Italiano, Efficient algorithms for sequence analysis, *in* "Proceedings, Second Workshop on Sequences: Combinatorics, Compression, Security, Transmission, Positano, Italy," to appear.

[F85] P. C. Fishburn, "Interval Orders and Interval Graphs: A Study of Partially Ordered Sets," Wiley, New York, 1985.

[FW74] M. J. Fisher and R. Wagner, The string to string correction problem, *J. Assoc. Comput. Mach.* **21** (1974), 168–178.

[FS83] W. M. Fitch and T. F. Smith, Optimal sequence alignments, *Proc. Nat. Acad. Sci. USA* **80** (1983), 1382–1386.

[GG89] Z. Galil and R. Giancarlo, Speeding up dynamic programming with applications to molecular biology, *Theor. Comput. Sci.* **64** (1989), 107–118.

[G82] O. Gotoh, An improved algorithm for matching biological sequences, *J. Mol. Biol.* **162** (1982), 705–708.

[G90] O. Gotoh, Optimal sequence alignment allowing for long gaps, *Bull. Math. Biol.* **52** (1990), 359–373.

[H77] D. S. Hirschberg, Algorithms for the longest common subsequence problem, *J. Assoc. Comput. Mach.* **24** (1977), 664–675.

[HD84] W. J. Hsu and M. W. Du, New algorithms for the LCS problem, *J. Comput. System Sci.* **29** (1984), 133–152.

[HW92] X. Huang and M. S. Waterman, Dynamic programming algorithms for restriction map comparison, *Comput. Appl. Biosci.*, **8** (1992), 511–520.

[HS77] J. W. Hunt and T. G. Szymanski, A fast algorithm for computing longest common subsequences, *Commun. ACM* **20** (1977), 350–353.

[J78] R. G. Jeroslav, Some basic theorems for integral monoids, *Math. Oper. Res.* **3** (1978), 145–154.

[J76] D. B. Johnson, Efficient algorithms for shortest paths in sparse networks, *J. Assoc. Comput. Mach.* **24** (1976), 1–13.

[KT82] D. Kelly and W. T. Trotter, Dimension theory for ordered sets, *in* "Ordered Sets" (I. Rival, Ed.), Reidel, Dordrecht/Boston, 1982.

[KR87] S. K. Kumar and C. P. Rangan, A linear space algorithm for the LCS problem, *Acta Inform.* **24** (1987), 353–362.

[K88] S. Y. Kung, "VLSI Array Processors," Prentice-Hall, Englewood Cliffs, NJ, 1988.

[L76] E. L. Lawler, "Combinatorial Optimization: Networks and Matroids," Holt, Rinehart, Winston, New York, 1976.

[LL85] R. S. Lipton and D. Lopresti, A systolic array for rapid string comparison, *in* "1985 Chapel Hill Conference on VLSI."

[L72] C. L. Liu, "Topics in Combinatorial Mathematics," Math. Assoc. Amer., Washington, DC, 1972.

[MM88] W. Miller and E. W. Myers, Sequence comparison with concave weighting functions, *Bull. Math. Biol.* **50** (1988), 97–120.

[MBR91] W. Miller, J. Barr, and K. E. Rudd, Improved algorithm for searching restriction maps, *Comput. Appl. Bio. Sci.*, 7 (1991), 447–456.

[M80] A. Mukhopadhyay, A fast algorithm for the longest-common-subsequence problem, *Inform. Sci.* **20** (1980), 69–82.

[NKY82] N. Nakatsu, Y. Kambayashi, and S. Yajima, A longest common subsequence algorithm suitable for similar text strings, *Acta Inform.* **18** (1982), 171–179.

[NW70] S. B. Needleman and C. D. Wunsch, A general method applicable to the search for similarities in the amino acid sequence of two proteins, *J. Mol. Biol.* **48** (1970), 443–453.

[R56] J. T. Robacker, "Min-Max Theorems of Shortest Chains and Disjunct Cuts of a Network," The RAND Corporation, RM-1660-PR, 1956.

[R84] M. A. Roytberg, "Algorithm of Homology Search for Primary Structures," Scientific Computer Centre Academy of Science, USSR, Pushino, 1–26, 1984. [Russian]

[SC78] H. Sacoe and S. Chiba, Dynamic-programming algorithm optimization for spoken-word recognition, *IEEE Trans. Acoust. Speech Signal Process.* **ASSP-26** (1978), 43–49.

[Saga91] B. E. Sagan, "The Symmetric Group. Representations, Combinatorial Algorithms and Symmetric Functions," Wadsworth & Brooks/Cole, Belmont, CA, 1991.

[S72] D. Sankoff, Matching sequences under deletion-insertion constraints, *Proc. Nat. Acad. Sci. USA* **69** (1972), 4–6.

[SS73] D. Sankoff and P. H. Sellers, Shortcuts, diversions, and maximal chains in partially ordered sets, *Discrete Math.* **4** (1973), 287–293.

[S81] A. Schrijver, On total dual integrality, *Linear Algebra Appl.* **38** (1981), 27–32.

[S91] A. Sebo, Hilbert bases, Caratheodory's theorem and combinatorial optimization, *in* "IPCO" (B. Pulleyblank, Ed.), Watcom, Waterloo, to appear.

[S74] P. H. Sellers, On the theory and computation of evolutionary distance, *SIAM J. Appl. Math.* **26** (1974), 787–793.

[S79] P. H. Sellers, Pattern recognition in genetic sequences, *Proc. Nat. Acad. Sci. USA* **76** (1979), 3041.

[SS92] R. F. Smith and T. F. Smith, Pattern-induced multi-sequence alignment (PIMA) algorithm employing secondary structure-dependent gap penalties for use in comparative protein modelling, *Protein Eng.*, **5** (1992), 35–42.

[SW81] T. F. Smith and M. S. Waterman, Identification of common molecular subsequences, *J. Mol. Biol.* **147** (1981), 195–197.

[SWF81] T. F. Smith, M. S. Waterman, and W. M. Fitch, Comparative biosequence metrics, *J. Mol. Evol.* **18** (1981), 38–46.

[VZ70] V. M. Velichko and N. G. Zagoruyko, Automatic recognition of 200 words, *Int. J. Man-Mach. Stud.* **2** (1970), 223–234.

[V68] T. K. Vintsyuk, Speech discrimination by dynamic programming, *Cybernetics* **4** (1968), 52–57.

[W76] R. A. Wagner, A shortest-path algorithm for edge-sparse graphs, *J. Assoc. Comput. Mach.* **23** (1976), 50–57.

[WSB76] M. S. Waterman, T. F. Smith, and W. A. Beyer, Some biological sequence matrices, *Adv. Math.* **20** (1976), 367–387.

[W84] M. S. Waterman, Efficient sequence alignment algorithms, *J. Theor. Biol.* **108** (1984), 333–337.

[WR87] M. S. Waterman and R. J. Raymond, The match game: New stratigrafic correlation algorithms, *Math. Geol.* **19** (1987), 109–127.

[WSK84] M. S. Waterman, T. F. Smith, and H. L. Katcher, Algorithms for restriction map comparisons, *Nucleic Acids Res.* **12** (1984), 237–242.

[YL86] C. B. Yang and R. C. T. Lee, The mapping of 2-D array processors to 1-D array processors, *Parallel Comput.* **3** (1986), 217–229.