Locating well-conserved regions within a pairwise alignment

Kun-Mao Chao, Ross C. Hardison [1,2] *and Webb Miller*[2]

Abstract

Within a single alignment of two DNA sequences or two protein sequences, some regions may be much better conserved than others. Such strong conservation may reveal a region that possesses an important function. When alignments are so long that it is infeasible, or at least undesirable, to inspect them in complete detail, it is helpful to have an automatic process that computes information about the varying degree of conservation along the alignment and displays the information in a graphical representation that is readily assimilated. This paper presents methods for computing several such "robustness measures" at each position of a given alignment. These methods are all very space-efficient; they use only space proportional to the sum of the two sequence lengths. To illustrate their effectiveness, one of the methods is used to locate particularly well-conserved regions in the β-globin gene locus control region and in the 5′ flank of the γ-globin gene.

Introduction

The utility of information about the reliability of different regions within an alignment is widely appreciated (e.g., Vingron and Argos, 1990; Zuker, 1991; Friemann and Schmitz, 1992). One approach to obtaining such information is to determine suboptimal alignments, i.e., some or all alignments that come within a specified tolerance of the optimum score (e.g., Waterman and Byers, 1985; Saqi and Sternberg, 1991). A number of these earlier papers modify the classic dynamic-programming algorithm for pairwise sequence alignment so that it computes such additional "robustness" information, and this strategy has been applied with some success to alignments of short protein sequences. However, since the traditional dynamic-programming formulation requires space proportional to the product of the sequence lengths, it is impractical for long protein sequences (the practical limit might be 1000 residues on a small workstation) and, especially, for long DNA sequences. In any case, the number of suboptimal alignments, or even alternative optimal alignments, can easily be so large as to preclude an exhaustive enumeration.

Our own work frequently involves aligning DNA sequences, primarily to learn about gene regulation or about evolution at the molecular level (Hardison and Miller, 1993). Sequence conservation has proved to be a reliable indicator of at least one class of regulatory elements. Specifically, regions of six or more consecutive nucleotides that are identical across a range of mammalian sequences, called "phylogenetic footprints" (Tagle *et al*., 1988), frequently correspond to binding sites for sequence-specific nuclear proteins (Gumucio *et al*., 1992). We also look for longer, imperfectly conserved (but strongly matching) regions, which may indicate other sorts of regulatory elements, such as a region that binds to a nuclear matrix or assumes some altered chromatin structure.

Departments of Computer Science and [1]Molecular and Cell Biology and [2]Center for Gene Regulation, The Pennsylvania State University, University Park, PA 16802

We often compute long alignments. For example, Chao *et al*. (1993) discuss a single alignment of over 100,000 base pairs, between chloroplast genomes of tobacco and liverwort. To determine the most strongly conserved regions of a long alignment, we routinely check for consistency with alignments involving DNA sequences from other species (Boguski *et al*., 1992: Miller, 1993). Even if sequence data for a third species are unavailable, we use simple tools to compute and display information about the distribution of gaps and the percentage of nucleotide identity between successive gaps of pairwise alignments (e.g., Figures 10 and 12, below).

This paper describes additional, more sophisticated measurements of the robustness of each position of a pairwise alignment. The first method computes, for each position i of the first sequence, the lower and upper limits of the positions in the second sequence to which it can be aligned and still come within a specified tolerance of the optimum alignment score. Delimiting suboptimal alignments this way, rather than enumerating all of them, allows the computation to run in only a small constant factor more time than the computation of a single optimal alignment. Another method determines, for each aligned pair (i.e., column) of an optimal alignment, the amount by which the optimum score must be lowered before reaching an alignment not containing that pair. In other words, if the optimum alignment score is s and the aligned pair is assigned the robustness-measuring number r, then any alignment scoring strictly greater than $s-r$ aligns those two sequence positions, while some alignment of score $s-r$ does not align them. As a special case, this value tells whether the pair is in all optimal alignments (namely, the pair is in all optimal alignments if and only if its associated value is non-zero). These computations are performed using dynamic-programming methods that require only space proportional to the sum of the two sequence lengths, in keeping with an algorithmic theme that we have consistently pursued (e.g., Myers and Miller, 1988; Huang *et al*., 1990; Huang *et al*., 1992). We also show how to efficiently handle the case where alignments are constrained so that each position, say position i, of the first sequence can be aligned only to positions between $L[i]$ and $R[i]$ of the second sequence, in harmony with another theme of our recent work (Chao *et al*., 1992; Chao *et al*., 1993). Note that in this case, arbitrary $L[i]$ and $R[i]$ are given as data for the problem, whereas the method mentioned at the start of this paragraph computes $L[i]$ and $R[i]$ that accomplish a specific purpose.

System and Methods

The programs described in this paper are written in C and were developed on Sun workstations running SunOS Unix. The code should be portable to a wide range of machines.

Algorithms

The widely used dynamic-programming method for sequence alignment (Needleman and Wunsch, 1970) can be interpreted as a method for determining a highest-scoring path in a certain graph, as follows. (For more details, see the survey by Pearson and Miller, 1992.) For aligning sequences of lengths M and N, form a rectangular grid of points with $M+1$ rows and $N+1$ columns. There are three edges entering the grid point (i, j) (i.e., the node lying at the intersection of row i and column j), as pictured in Figure 1. (Of course, row 0 and column 0 are special cases.) Weights are assigned to each edge, with edges that correspond to an identity or a conservative substitution being given a positive weight, and other weights being negative. The alignment problem is to find a maximum-score path from $(0, 0)$ to (M, N), i.e., to maximize the sum of edge weights.

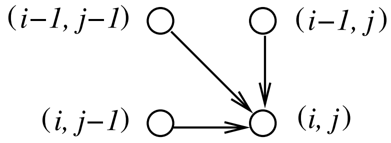

Fig. 1. Edges entering the node at grid point (i, j). Gaps of length k are penalized $r \times k$ for some constant r.

In practice, we don't want to explicitly construct this graph, since it is huge if the sequences are long. This economy is easy to achieve if we seek only the score of an optimal path (as opposed to producing the path itself). Define $Score^{-}(i, j)$ to be the optimal score of a path from $(0, 0)$ to (i, j). This score can be easily computed from the scores at $(i, j-1)$, $(i-1, j)$ and $(i-1, j-1)$, so scores for nodes in row i can be computed from the scores for row $i-1$. Since scores in row 0 are obvious, we can make a "forward pass" (row 0, row 1, row 2, $\cdots$) where we save only the scores in the previous row and the current row. Indeed, with a minor bit of cleverness, we require only space for the scores in one row plus an additional entry. $Score^{-}(M, N)$ is the solution to the "score-only" problem.

Hirschberg (1975) introduced a method that explicitly delivers an optimal path using temporary storage for only a couple of rows. The idea is to make a forward pass computing $Score^{-}$, stopping at row $M/2$ (round down for odd M), i.e., the middle row. Then make an analogous "backward pass" (row M, row $M-1$, $\cdots$, row $M/2$) computing $Score^{+}(i, j)$, defined as the optimal score of a path from (i, j) to (M, N). Adding the resulting values for nodes in row $M/2$ yields $Score(i, j) = Score^{-}(i, j) + Score^{+}(i, j)$ for $i = M/2$ and $0 \le j \le N$, where $Score(i, j)$ gives the optimal score of a path from $(0, 0)$ to (M, N) constrained to pass through (i, j). We simply pick $j_{\max}$ to maximize this value in row $M/2$, which gives a midpoint $(M/2, j_{\max})$ on an optimal path from $(0, 0)$ to (M, N), then recursively compute an optimal path from $(0, 0)$ to $(M/2, j_{\max})$ and an optimal path from $(M/2, j_{\max})$ to (M, N). This involves recomputing values at certain grid points, but the total time for computing all points on an optimal path is essentially twice that for determining the initial $(M/2, j_{\max})$, which in turn has a cost essentially equal to the cost of a single pass to compute $Score^{-}(M, N)$.

The general situation during this process is pictured in Figure 2. The next step is to apply forward and backward passes in the first nondegenerate rectangle along the optimal path being generated. Within a subproblem (i.e., rectangle) the scores of paths can be taken relative to the "start node" at the rectangle's upper left and the "end node" at the lower right. This means that a subproblem is completely specified by giving the coordinates of those two nodes. (In contrast, methods described below must maintain more information about each pending subproblem.)

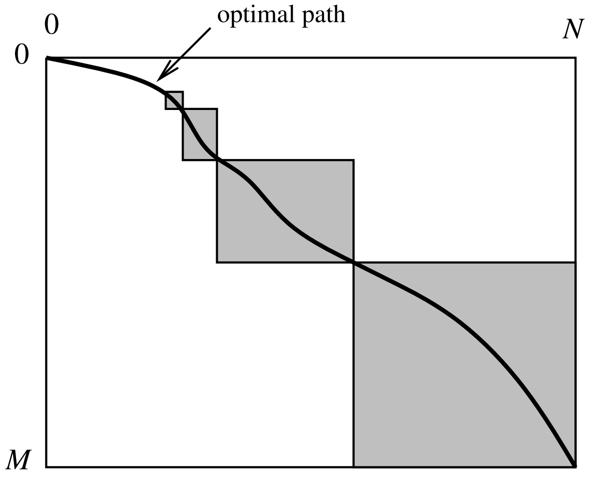

Fig. 2. The collection of pending subproblems at some point during execution of Hirschberg's method.

Hirschberg's original method, and the above discussion, apply to the case where the penalty for a gap is merely proportional to the gap's length, i.e., rk for a k-symbol gap. For applications in molecular biology, one wants penalties of the form $q + rk$, i.e., each gap is assessed an additional "gap-open" penalty q. (Actually, one can be slightly more general and substitute residue-dependent penalties for r.) In this case the relevant graph is more complicated (Myers and Miller, 1989). Now at each grid point (i, j) there are three nodes, denoted $(i, j)_S$, $(i, j)_D$ and $(i, j)_I$, and generally seven entering edges, as pictured in Figure 3. The alignment problem is to compute a highest-score path from $(0, 0)_S$ to $(M, N)_S$. A source of added complexity in this more general model is that now there is in general no leftmost (or rightmost) optimal path, which can happen because optimal paths might go through the same grid point but not the same node. See Figure 4.

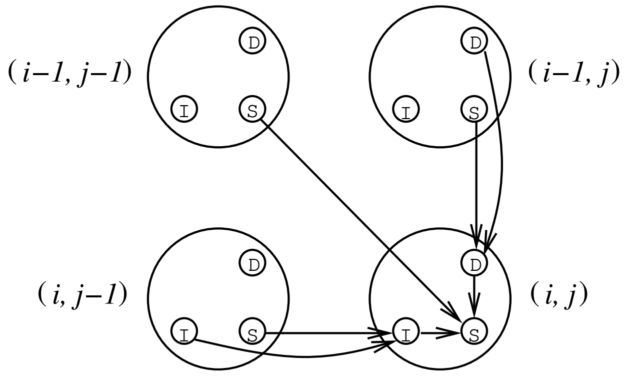

Fig. 3. Edges entering the nodes at grid point (i, j). Gaps of length k are penalized $q + r \times k$ for some constants q and r.

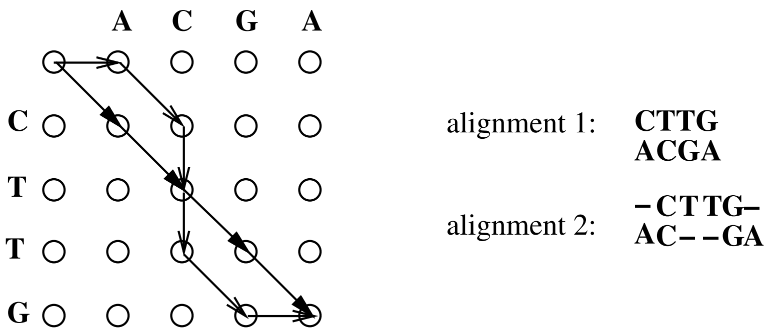

Fig. 4. The two optimal alignments of *CTTG* and *ACGA*. Matches score 1, mismatches score -1 and a gap of length k is penalized $1 + 0.75k$. Both alignments score -4. Note that the alignments don't actually intersect at the middle grid point: alignment 1 goes though $(2,2)_S$ and alignment 2 goes through $(2,2)_D$.

Fortunately, Hirschberg's strategy extends readily to this more general class of alignment scores (Myers and Miller, 1988). In essence, the main additional complication is that for each defining corner of a subproblem we need to specify one of the grid point's three nodes.

It is worth noting that Hirschberg's strategy is not the only way to subdivide a dynamic-programming matrix to achieve linear-space performance. Chao *et al.* (1992, 1993) utilize a strategy that subdivides a problem into a variable number of subproblems, as opposed to Hirschberg's two equal-height subproblems.

The purpose of this paper is to extend Hirschberg's approach to compute additional information that in some sense indicates the degree to which the aligned residues are conserved. Two such methods are developed, as described in the Introduction. Moreover, it is shown how to proceed if the alignment is constrained *a priori* to lie in a restricted region of the dynamic-programming grid. All these methods run in linear space and score-only time, i.e., execution time that is at most a constant multiple of the time to produce the alignment's score.

The left extent of suboptimal alignments

Suppose we are given a threshold score that does not exceed the optimum score of a path. A path from $(0,0)_S$ to $(M,N)_S$ is *suboptimal* if its score is at least as large as the threshold score. For each row i, define $L[i]$ to be the index of the leftmost column where a suboptimal path intersects row i. We now show how to modify Hirschberg's method to compute L-values.

A general subproblem is pictured in Figure 5. The upper left corner is at a grid point $(s, L[s])$ and the lower right corner is at $(t, L[t])$ (with the special case (M,N) in the last row). For every node on the left and top borders of a pending subproblem, we save $Score^-$ (the optimal score of a path from $(0,0)_S$ to that node). Similarly, for every node on the right and bottom borders, we save $Score^+$. It is clear from Figure 2 that the sum of the lengths of the top borders is essentially bounded by N, and similarly for the other borders, so linear space is adequate for all pending subproblems. (We say "essentially" because the top borders for successive subproblems overlap in one position, giving a total of at most $\log_2 M$ extra positions.)

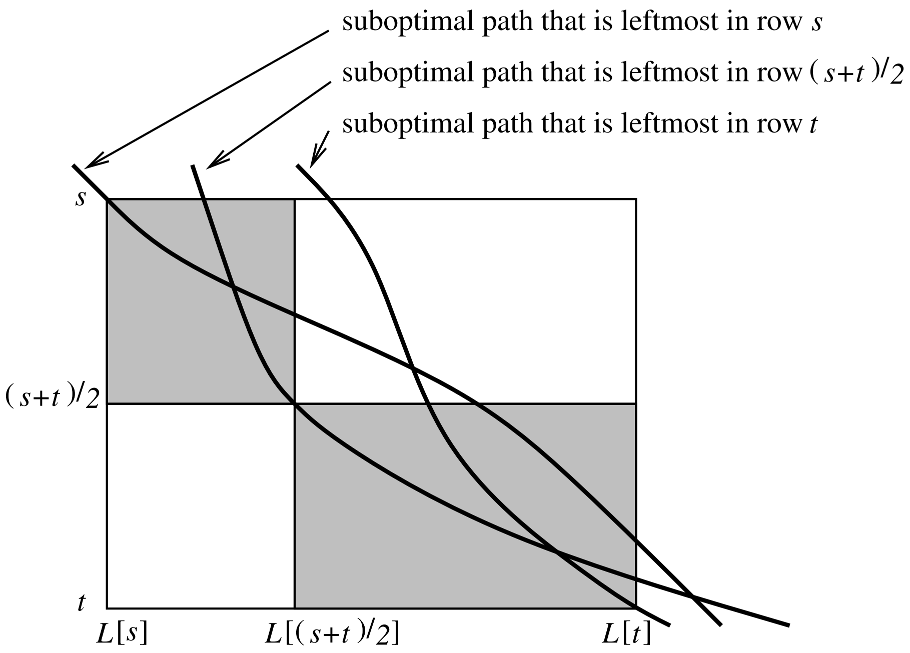

Fig. 5. General subproblem when computing the left extent of suboptimal alignments.

Problems with one column, or with one or two rows, can be solved directly. To subdivide larger subproblems, make forward and backward passes to propagate values of $Score^-$ and $Score^+$, respectively, to the middle row, whereupon $L[(s+t)/2]$ can be determined. Once $L[(s+t)/2]$ is known, we continue the forward and backward passes from row $(s+t)/2$ to propagate values $Score^-$ to borders of the lower subsubproblem and values $Score^+$ to the upper subsubproblem, which guarantees the invariant conditions.

The approach just described can be made more efficient, in terms of both time and space, by observing that it is not necessary to retain scores on the left and bottom boundaries of a subproblem, since suboptimal paths cannot enter or leave the rectangle through those borders. The resulting procedure is slightly harder to understand, however.

The robustness measure for each edge of an optimal path

The edge from $(i-1, j)_D$ to $(i, j)_D$ and the edge from $(i-1, j)_S$ to $(i, j)_D$ (Figure 3) are called *twins*, since they both correspond to the aligned pair that deletes the i^{th} entry of the first sequence. Similarly the edge from $(i, j-1)_I$ to $(i, j)_I$ and the edge from $(i, j-1)_S$ to $(i, j)_I$ are twins. For each edge of an optimal path, let the *robustness measure* of that edge be the difference between the optimal score and the highest score of all paths (i.e., from $(0,0)_S$ to $(M, N)_S$) that do not use the edge or its twin. A zero value means that the aligned pair is not essential to optimal alignments, while a positive value means that this column of an optimal alignment is required even with tiny changes in the scoring scheme.

We now show how to compute this value for all vertical and diagonal edges of an optimal path; horizontal edges can be handled simultaneously with a slightly more complex procedure. (We are not concerned with the edges to $(i, j)_S$ from $(i, j)_D$ or $(i, j)_I$, since they do not correspond to an aligned pair.) The *Score* of an edge is defined as the maximum score of all paths from $(0,0)_S$ to $(M, N)_S$ that use the edge. For each row i, define $RM[i] \geq 0$ to be the difference between the highest score over all edges that enter that row and the highest score of a non-twin edge. If two optimal paths enter row i along non-twin edges, then $RM[i] = 0$. Note that for any optimal path, the robustness measure for the edge entering row i is $RM[i]$.

As a first approximation to an algorithm to compute the RM-values, consider Figure 5, but use leftmost optimal paths in place of leftmost suboptimal paths. Clearly an optimal path that is leftmost at row $(s+t)/2$ must intersect that row within the rectangle. A forward and a backward pass lets us compute the score of any edge that enters row $(s+t)/2$ within the rectangle, which handles the optimal score. The only problem is that we cannot guarantee that the second highest scoring edge entering row $(s+t)/2$ must lie within the box.

This problem can be solved by maintaining values $HR[i]$, defined to be the highest score over all edges entering row i that are not contained within a pending subproblem. $RM[(s+t)/2]$ is readily computed using $HR[(s+t)/2]$ and information about edges entering row $(s+t)/2$ within the rectangle. The only remaining problem is to update HR in linear space and score-only time when a problem is subdivided.

Figure 6 depicts the strategy. When the problem is divided, $HR[i]$ may increase because row i intersects the new pending problem in a smaller range of columns. Namely, we must now account for the scores of edges in the region denoted Δ, and we must do this for all i between s and $(s+t)/2$ in linear space and in time proportional to the area of the upper right region being discarded from the problem. The critical observation is that any path through Δ must pass through the regions denoted Γ, so every score of an edge in Δ is the score of an edge in Γ. We make a forward pass to propagate values of $Score^-$ to nodes along the vertical segment from (s, j_{best}) to $((s+t)/2, j_{\text{best}})$. (Note that we don't know j_{best} until the original forward pass reaches row $(s+t)/2$, necessitating a second pass.) A backward pass beginning at row $(s+t)/2$ propagates values of $Score^+$ to the right and bottom borders of Γ, so edges across Γ can be scored. Given the maximum edge score for Γ, we can compute the new maximum when i is raised to $i+1$ in constant time, since the enlarged Γ contains only three new edges. Thus, after the backward pass to reach Γ's borders, we need only conduct a simple counter-clockwise scan around Γ to update $HR[i]$ for all i between s and $(s+t)/2$.

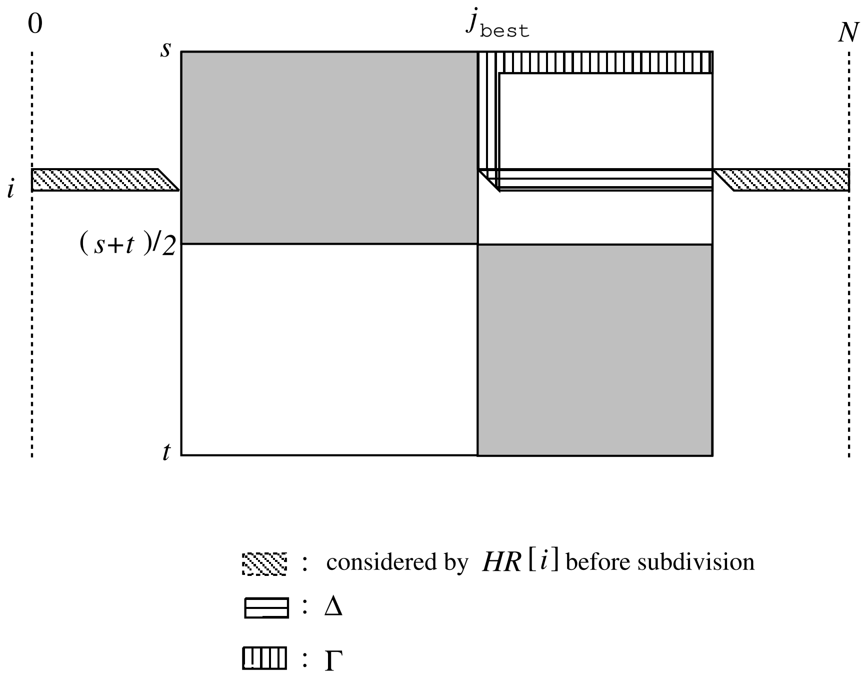

Fig. 6. General subproblem when computing the robustness measure.

Computation within a region

Suppose we have computed the left and right extents of suboptimal paths relative to a given threshold score and that we now want to repeat the process for a larger threshold. For efficiency reasons, we would like to delimit the smaller region of suboptimal alignments by performing a computation that considers only those grid points bounded by the original left and right extents. More generally, assume we are given values $L[i] \le R[i]$ for each row i. Further assume that $L[0] = 0 \le R[0]$, $L[M] \le N = R[M]$, and L and R are monotonic in the sense that, e.g., $L[i-1] \le L[i]$ for $1 \le i \le M$. We want to apply the algorithms discussed above to the region consisting of grid points (i, j) where $L[i] \le j \le R[i]$, where the running time should be proportional to the number of grid points in the region. The problem with a direct attack, which does forward and backward passes to the middle row of the region, is that for narrow regions the total running time can exceed the score-only time by a factor $\log_2 M$ (see Chao *et al.*, 1992).

Our strategy is to cover the region with a small number of non-intersecting upright rectangles whose total area does not exceed twice that of the rectangle's intersection with the given region. By "area" we mean the number of grid points in the interior or on the border. These rectangles are positioned in a greedy fashion; we repeat the process of placing a new rectangle that covers the first exposed row and extends as far as possible. See Figure 7, where the dark line segments indicate the grid points of the top row of a rectangle that can be reached along a single edge from the previous rectangle. The algorithm of Figure 8 computes the set I containing the indices of first rows of all rectangles except the first.

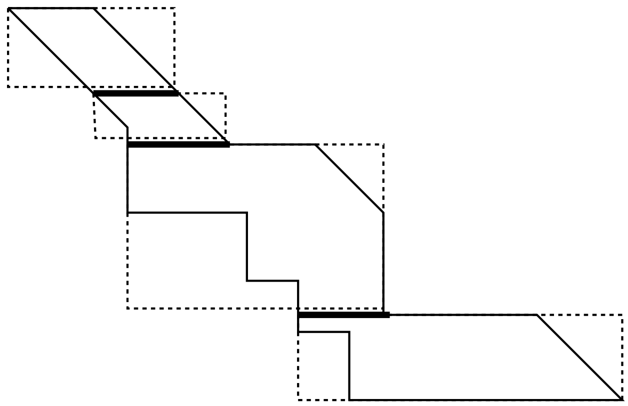

Fig. 7. Covering the region defining a constrained alignment problem with non-intersecting rectangles.

$I \leftarrow$ empty
$i_0 \leftarrow 0$
$sum\_widths \leftarrow R[0] - L[0] + 1$
for $i \leftarrow 1$ **to** M **do**
  { $width \leftarrow R[i] - L[i] + 1$
    $sum\_widths \leftarrow sum\_widths + width$
    if $(R[i] - L[i_0] + 1) \times (i - i_0 + 1) > 2 \times sum\_widths$ **then**
      { $I \leftarrow I \cup \{i\}$
        $i_0 \leftarrow i$
        $sum\_widths \leftarrow width$
      }
  }

Fig. 8. Algorithm to partition a region of grid points.

The following technical lemma is essential to our approach, since it implies that the total length of the dark segments in Figure 7 is linear in N. Its proof is relatively uninteresting and is relegated to the Appendix.

Lemma. The sum of the segment widths $R[i] - L[i] + 1$ for $i \in I$ is at most $3N$.

This observation allows us to efficiently apply the earlier algorithms to constrained alignment problems. The algorithm of Figure 8 is first used to break the problem into sections. A backward pass through the entire region allows us to compute and retain values $Score^{+}$ for all nodes on dark line segments of Figure 7, which by the Lemma involves only linear space. The sections of the region are then treated from top to bottom. Each section is first reduced to a special case of the general subproblem depicted in Figure 5, then one of the earlier algorithms is applied, with some extra code that keeps all computations to feasible grid points.

The inductive hypothesis is that $Score^{-}$ is given for the first row of the current section. That way, the leftmost position, $L[s]$ of a suboptimal or optimal path (depending on which problem is being solved) in the section's first row, s, can be determined. A forward pass then propagates values of $Score^{-}$ through the section. Values of $Score^{-}$ can be retained along the line segment denoted a in Figure 9. (They are unnecessary for computing the left extent of suboptimal paths, as mentioned above.) When the section's last row is reached, $L[t]$ for the section's last row, t, can be determined. $Score^{+}$ for row t and line segment c is calculated in a backward pass starting from line segment b and passing over the portion to the right of column $L[t]$, inclusive. The reduced section delimited by rows s and t and by columns $L[s]$ and $L[t]$ is exactly a restricted version of Figure 5 and the required $Score^+$ and $Score^-$ values are now known. Thus either of the earlier algorithms can be applied to the reduced current section. (For robustness measures, we also need values $HR[i]$, but they pose no problem.) To keep the inductive hypothesis valid for the next section, the forward pass sketched above is extended one row further, to line segment b.

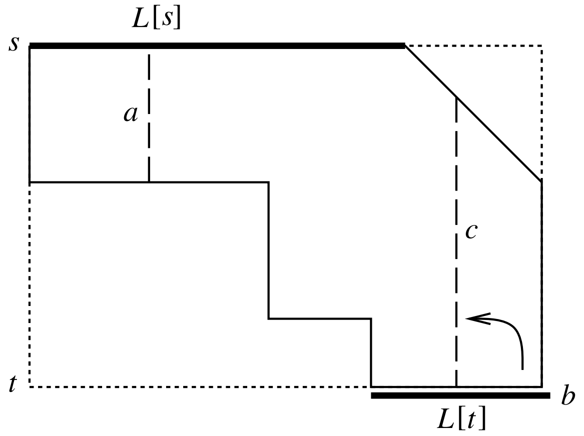

Fig. 9. Conversion of a section of the region to a special case of Figure 5 (see text).

For computing either extents of suboptimal solutions or robustness measures, there is a constant, call it k, such that the total number of grid points evaluated when running the algorithm on a rectangle's intersection with the region is at most k times the area of the enclosing rectangle, hence at most $2k$ times the area of the region's section. It follows readily that the entire process (breaking into sections and solving the pieces) takes time proportional to the area of the constraining region. In brief, the algorithms attain "score-only" running times on rectangular problems, so we make a preliminary pass to divide the region into "nearly rectangular" sections.

Implementation

We have implemented the above method for determining the region containing suboptimal alignments, i.e., alignments whose score lies within a fixed departure from optimality. The program for determining both the left and right boundaries, called *left_right*, has a measured running time of approximately 5.6 times that of a program that computes merely the optimal cost. It uses temporary array storage for $4M+6N+10$ integers.

Another program, called *robust*, computes the robustness measure for all edges (including horizontal edges) of an optimal alignment. It runs in approximately 5.4 times the score-only time and uses array storage for $8M+8N+16$ integers. Certain economies are possible when one wants only a bit telling if the aligned pair is in all optimal alignments. A program called *unique* does this in 2.2-3.8 times the score-only time and $4M+8N+12$ integers of array storage. The strong dependence of *unique*'s execution time on its input stems mostly from the fact that if there is a unique optimal edge entering the middle row, then recomputation within the subproblems is not needed to complete the division step. Thus *unique* runs substantially faster with unique optimal alignments than under conditions of global non-uniqueness.

Examples

To illustrate the utility of the methods developed above, the program for computing the left and right extents of suboptimal alignments was applied to alignments between regulatory regions of the β-like globin gene clusters of humans and rabbits. The general plan is to (1) compute an optimal alignment, (2) lower the optimal alignment score, s, slightly to get a threshold $t < s$, (3) compute the left and right extents of all suboptimal alignments scoring at least t and (4) determine the subregions where the left and right extents are unusually close together (e.g., the pinched regions in Figures 10 and 12). These are the sections where even somewhat suboptimal alignments must match sequence positions in almost the same way as is done by an optimal alignment; this robustness property is interpreted as indicating particularly well-conserved sections of the alignment. Two regions of interest were studied this way, as described next.

Hypersensitive site 4 (HS4) of the β-globin locus control region

The locus control region (LCR) of the β-globin gene cluster is a dominant, *cis*-acting regulatory sequence that is thought to function in opening an active chromatin domain (Orkin, 1990). It is marked by a set of at least four DNase hypersensitive sites (Tuan *et al*., 1985), named HS1 through HS4, located distal to the gene cluster. DNA fragments from this region can confer position-independent, high-level expression of linked genes in transgenic mice and transfected cells (Grosveld *et al*., 1987; Tuan *et al*., 1989). Although the structure and function of HS2 and HS3 have been studied in considerable detail in several different species (e.g. Li *et al*., 1990; Hug *et al*., 1992; Hardison *et al*., 1993), much less is known about HS4. This DNA segment is as effective as HS2, a strong enhancer, in conferring position-independent expression on linked genes in transgenic mice (Fraser *et al*., 1990), but more precise functional mapping has been limited to analysis of sequences that will generate DNase hypersensitive sites in transfected cells (Lowrey *et al*., 1992). The latter study has implicated binding sites for the transcription factors NFE2/AP1 and GATA1 in HS4 function, but more data are needed to show that these sequences are necessary and to investigate the possible function of other sequences in this region.

Evolutionary conservation is a very helpful guide to identifying candidate regulatory sequences. DNA sequences from the HS4 region are available only from human (Li *et al*., 1985) and rabbit (Hardison *et al*., 1993), thus pairwise alignments can be made, but the discriminating power of multiple alignments is not available. As previously reported (Hardison *et al*., 1993), almost all of the sequenced 1300 bp region from rabbit aligns with human, and the percent identity varies little across the alignment (Figure 10, top). The region required to establish a hypersensitive site in transfected cells (the HS4 core) is indicated along the horizontal axis (human DNA) and the binding sites for NFE2/AP1 and GATA1 in both sequences are also shown. However, the percent identity calculated for this HS4 core is no greater than for most other segments in this region. These two binding sites do stand out in the plot of extents of suboptimal alignments (Figure 10, bottom); the third and fourth constrictions contain the NFE2/AP1 site and the GATA1 site, respectively (Figure 11). A match to the CDP (CCAAT displacement factor

(Superti-Furta *et al*., 1988)) binding site found in both sequences 5′ to the NFE2/AP1 site is not in a pinched region (Figures 10 and 11). Thus the extent of suboptimal alignments support the proposal that the NFE2/AP1 and GATA1 sites are important for HS4 function.

Other notable constrictions in Figure 10 are found outside the HS4 core. The constriction at human sequence positions around 772-878 (Figure 11) contains within it a gap-free alignment of 61 identical nucleotides out of the 81 pairs in this segment (75.3% identity); again, not an unusually high percent identity. Presumably the more important factor in its showing as a constriction in the plot is the inflexibility in placing the gaps on each side of this aligning segment. The gaps are relatively long (10 and 17 positions) and have little terminal redundancy that would allow other alignments, using different gap placement, to score almost as high. This limited flexibility in gap placement results in a more robust alignment. The point that the constrictions are not simply a function of high percent identity is illustrated by the alignment involving human positions 1073 to 1092 (Figure 11). This section matches exactly with the rabbit sequence at 19 of 20 positions (95% identity), the highest scoring segment in the plot of Figure 10 (upper). However, this is not at a constriction point, presumably because the gap at the 5′ end is short and in a repeating pyrimidine sequence.

Figure 10 goes here.

Fig. 10. The left and right extents of suboptimal alignments between the HS4 portion of the β-globin locus control regions of human and rabbit, as computed by the *left-right* program described in this paper. The top panel shows the location of an optimal local alignment computed by the program *sim* (Huang *et al.*, 1990); short perpendicular lines indicate conserved matches to a small library of transcription factor binding sites. Identically matching nucleotides scored 1, mismatches scored -1 and k-symbol gaps were penalized $6 + 0.2k$. The core of the HS4 region in the human sequence is shown as an open box on the horizontal axis, and the filled triangle indicates an Alu repeat. The center panel shows the percentages of identical nucleotides in each segment of the alignment lying between successive gaps, as a function of position in the human sequence. The lower panel depicts the optimal alignment as a light line and the left and right boundaries as darker lines. The optimal alignment scored 247.4 and the threshold for suboptimal alignment scores was 217.4. The computation by *left-right* required 9.5 seconds on a Sun SparcStation 2. The diagrams were drawn with the *laps* program (Schwartz *et al.*, 1991), and vertical dotted lines and a few annotations were added by hand. The human sequence is from Li *et al.* (1985) and the rabbit sequence is from Hardison *et al.* (1993), GenBank accession number L05835.

Figure 11 goes here.

Fig. 11. Alignments around constriction points in the comparison of HS4 between human and rabbit. Notable constrictions in Figure 10 are overlined in the human sequence, and matches to binding sites for nuclear factors are underlined in both sequences and labeled (CDP = CCAAT displacement factor; NFE2 = NFE2/AP1). The aligning segment corresponding to human positions 1073-1092 (double underlined beneath the rabbit sequence) does not have a constriction in Figure 10. Identically matching pairs of nucleotides are marked by vertical lines and transitions are marked by colons.

Sequences flanking the 5′ end of the γ-globin gene

The promoter and sequences further 5′ to the γ-globin genes have been intensively studied because several mutations that lead to persistent expression of fetal hemoglobin ($\alpha_2\gamma_2$) in adult life map in this region. Virtually all the single-copy region between ε- and γ-globin genes aligns in pairwise comparisons among many mammals (reviewed in Hardison and Miller, 1993), and multiple alignments have been generated for the sequences from many primates and rabbits (e.g. Gumucio *et al*., 1992). These multiple alignments reveal "phylogenetic footprints" of short segments that are invariant in the sequences aligned, and in the vast majority of cases examined these correspond to sites for sequence-specific binding proteins (Gumucio *et al*., 1992). As shown in Figure 12 (upper panel), the nonrepetitive sequences 5′ to the human ${}^{G}\gamma$-globin gene and the rabbit γ-globin gene align, albeit with some variation in the percent identity. The plot of extents of suboptimal alignments in the lower panel of Figure 12 shows several prominent constrictions, i.e. positions where the alignments are highly constrained. The constriction around human positions 33409-33470 (Figure 13) contains (1) the phylogenetic footprint at −1250 (human positions 33409 to 33416) that is a binding site for CSBP1 (Gumucio *et al*., 1992), (2) a match to the GATA1 binding site, and (3) a perfect match of the segment TCCCAGCTGT that includes an E box (CANNTG), which is the binding site for a variety of heterodimeric helix-loop-helix proteins, of which MyoD is a prominent example (Blackwell and Weintraub, 1990). The matches to the GATA1 site and the E box are not in the footprints of Gumucio *et al*. (1992), partly because of differences in the alignments generated in that region and because of mismatches with the galago sequence. As further shown in Figure 13, the constriction around human positions 33651-33740 marks a region containing two phylogenetic footprints (at positions −960 and −930 in the multiple alignment of Gumucio *et al*. (1992)) and two co-aligning binding sites for the CACC-binding factor and one for CDP (which is in the −960 phylogenetic footprint). This latter constriction is also bounded by firmly placed gaps, as discussed above. The region corresponding to human positions 33651-33740 was previously noted as a strongly aligning region with multiple potential binding sites in the comparison of the rabbit γ and human ${}^{A}\gamma$ sequences (Huang *et al*., 1990); the ${}^{A}\gamma$ and ${}^{G}\gamma$ human sequences are very similar in this region. This earlier work also noted a strong match to a GATA1 site in the human/rabbit comparison (human positions 33593 to 33598, Figure 13). This region does not stand out in the analysis of Gumucio *et al*. (1992), but they use a different, and clearly suboptimal, alignment of the rabbit sequence with primate sequences in this region. If the alignment from Figure 13 is used, a clear phylogenetic footprint is revealed around this GATA1 site. The convergence of these two completely different approaches to identifying conserved sequences (i.e., phylogenetic footprints and extents of suboptimal alignments) in identifying these interesting regions 5′ to the γ-globin genes supports the hypothesis that these sequences play some important role in gene function or regulation. With this in mind, the other constriction points further upstream, around human positions 31640 and 31820, may also identify candidates for regulatory sequences.

Figure 12 goes here.

Fig. 12. The left and right extents of suboptimal alignments between the 5′ flanking region of the γ globin genes of human and rabbit, computed and displayed as with Figure 10. Open boxes with pointed ends indicate L1 repeats, and genes are drawn with filled-in exons. The optimal alignment scored 762 and the threshold for suboptimal alignment scores was 712. The computation by *left-right* required 110 seconds on a Sun SparcStation 2. The human sequence is from Collins and Weissman (1984) and Li *et al.* (1985), and the rabbit sequence is from Margot *et al.* (1989).

Figure 13 goes here.

Fig. 13. Alignment of part of the 5′ flanking sequence between human $^{G}\gamma$ and rabbit γ-globin genes. The matching segments containing human positions 33,409-33,470 and 33,651-33,740 (overlined) correspond to constrictions in Figure 12. Matches to "phylogenetic footprints" identified by Gumucio *et al.* (1992) are underlined in both sequences, as are matches to binding sites for known transcription factors (GATA1, E box, CAC BP and CDP).

Discussion

The constrictions observed in the extents of suboptimal alignments are produced by a variety of factors, including the percent identity in the aligned segments and the flexibility in placement of gaps in the alignment. Aligning segments bounded by inflexible gaps tend to appear as constrictions that are landmarks for notable sequences. The use of extents of suboptimal alignments to identify well-conserved segments within longer alignments is a novel approach that identifies some previously recognized important sequences (such as exons of the γ-globin genes, or core hypersensitive sites in the LCR). It also gives useful information beyond the simple strength of matches (percent match) when only two sequences are available. In one test case, HS4, previously recognized binding sites were identified along with other regions flanking the core. When compared with the results of an extensive multiple alignment in the 5′ flank of γ-globin genes, the constrictions of the region containing suboptimal alignments fell close to some prominent phylogenetic footprints and reveals other potential binding sites, thereby validating the efficacy of the suboptimal-alignment approach and reinforcing the potential regulatory roles of these regions. Obviously these are important DNA segments to subject to functional tests for effects on expression of globin genes.

Availability

The C source code described in the Implementation section can be obtained over the Internet via anonymous ftp from groucho.cs.psu.edu. The authors can be contacted by electronic mail at webb@cs.psu.edu.

Acknowledgements

K.-M. C. and W. M. were supported by grant R01 LM05110 from the National Library of Medicine. R. C. H. was supported by PHS grant R01 DK27635 and an RCDA KO4 DK01589.

References

Blackwell, T. K., and Weintraub, H. (1990) Differences and similarities in DNA-binding preferences of MyoD and E2A protein complexes revealed by binding site selection. *Science*, **250**, 1104-1110.

Boguski, M., Hardison, R. C., Schwartz, S., and Miller, W. (1992) Analysis of conserved domains and sequence motifs in cellular regulatory proteins and locus control regions using new software tools for multiple alignment and visualization. *The New Biologist*, **4**, 247-260.

Chao, K.-M., Pearson, W. R., and Miller, W. (1992) Aligning two sequences within a specified diagonal band. *CABIOS*, **8**, 481-487.

Chao, K.-M., Hardison, R. C., and Miller, W. (1993) Constrained sequence alignment. *Bull. Math. Biol.*, in press.

Collins, F. S., and Weissman, S. M. (1984) The molecular genetics of human hemoglobin. *Prog. Nucleic Acid Res. Mol. Biol.*, **31**, 315-462.

Fraser, P., Hurst, J., Collis, P., and Grosveld, F. (1990) DNase I hypersensitive sites 1, 2 and 3 of the human β-globin dominant control region direct position-independent expression. *Nucl. Acids Res.*, **18**, 3503-3508.

Friemann, A., and Schmitz, S. (1992) A new approach for displaying identities and differences among aligned amino acid sequences. *CABIOS*, **8**, 261-265.

Grosveld, F., van Assendelft, G. B., Greaves, D., and Kollias, G. (1987) Position-independent, high-level expression of the human β-globin gene in transgenic mice. *Cell*, **51**, 975-985.

Gumucio, D. L., Heilstedt-Williamson, H., Gray, T. A., Tarle, S. A., Shelton, D. A., Tagle, D., Slightom, J., Goodman, M., and Collins, F. S. (1992) Phylogenetic footprinting reveals a nuclear protein which binds to silencer sequences in the human γ and ε globin genes. *Mol. Cell. Biol.*, **12**, 4919-4929.

Hardison, R. C., and Miller, W. (1993) Use of long sequence alignments to study the evolution and regulation of mammalian globin gene clusters. *Mol. Biol. Evol.*, in press.

Hardison, R. C., Xu, J., Jackson, J., Mansberger, J., Selifonova, O., Grotch, B., Petrykowska, H., Biesecker, J., and Miller, W. (1993) Comparative analysis of the locus control region of the rabbit β-like globin gene cluster: HS3 increases transient expression of an embryonic ε-globin gene. *Nucleic Acids Res.*, submitted.

Hirschberg, D. S. (1975) A linear space algorithm for computing maximal common subsequences. *Comm. ACM*, **18**, 341-343.

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

Huang, X., Miller, W., Schwartz, S., and Hardison, R. C. (1992) Parallelization of a local similarity algorithm. *CABIOS* **8**, 155-165.

Hug, B. A., Moon, A. M., and Ley, T. J. (1992) Structure and function of the murine β-globin locus control region 5′ HS-3. *Nucl. Acids Res.*, in press.

Li, Q., Powers, P., and Smithies, O. (1985) Nucleotide sequence of 16-kilobase pairs of DNA 5′ to the human ε-globin gene. *J. Biol. Chem.*, **260**, 14901-14910.

Li, Q., Zhou, B., Powers, P., Enver, T., and Stamatoyannopoulos, G. (1990) β-Globin locus activation regions: Conservation of organization, structure and function. *Proc. Natl. Acad. Sci. USA.*, **87**, 8207-8211.

Lowrey, C. H., Bodine, D. M., and Nienhuis, A. W. (1992) Mechanism of DNase I hypersensitive site formation within the human globin locus control region. *Proc. Natl.*

Acad. Sci. USA., **89**, 1143-1147.
Margot, J. B., Demers, G. W., and Hardison, R. C. (1989) Complete nucleotide sequence of the rabbit β-like globin gene cluster: analysis of intergenic sequences and comparison with the human β-like globin gene cluster. *J. Mol. Biol.*, **205**, 15-40.
Miller, W. (1993) Building multiple alignments from pairwise alignments. *CABIOS*, in press.
Myers, E. W., and Miller, W. (1988) Optimal alignments in linear space. *CABIOS*, **4**, 11-17.
Myers, E. W., and Miller, W. (1989) Approximate matching of regular expressions. *Bull. Math. Biol.*, **51**, 5-37.
Needleman, S. B., and Wunsch, C. (1970) A general method applicable to the search for similarities in the amino acid sequences of two proteins. *J. Mol. Biol.* **48**, 444-453.
Orkin, S. (1990) Globin gene regulation and switching: Circa 1990. *Cell*, **63**, 665-672.
Pearson, W. R., and Miller, W. (1992) Dynamic programming algorithms for biological sequence comparison. *Meth. Enz.*, **210**, 575-601.
Saqi, M., and Sternberg, M. (1991) A simple method to generate non-trivial alternative alignments of protein sequences. *J. Mol. Biol.*, **219**, 727-732.
Schwartz, S., Miller, W., Yang, C.-M., and Hardison, R. C. (1991) Software tools for analyzing pairwise alignments of long sequences. *Nucl. Acids Res.*, **17**, 4663-4667.
Superti-Furta, G., Barberis, A., Schaffner, W., and Busslinger, M. (1988) The −117 mutation in Greek HPFH affects the binding of three nuclear factors to the CCAAT region of the γ-globin gene. *EMBO J.*, **7**, 3099-3107.
Tagle, D., Koop, B., Goodman, M., Slightom, J., Hess, D., Jones, R. (1988) Embryonic ε and γ globin genes of a prosimian primate (*Galago crassicaudatus*): Nucleotide and amino acid sequences, developmental regulation and phylogenetic footprints. *J. Mol. Biol.*, **203**, 439-455.
Tuan, D., Solomon, W., Li, Q., and London, I. (1985) The β-like globin gene domain in human erythroid cells. *Proc. Natl. Acad. Sci. USA.*, **82**, 6384-6388.
Tuan, D., Solomon, W., London, I., and Lee, D. (1989) An erythroid-specific, developmental-stage-independent enhancer far upstream of the human "β-like globin" genes. *Proc. Natl. Acad. Sci. USA.*, **86**, 2554-2558.
Vingron, M., and Argos, P. (1990) Determination of reliable regions in protein sequence alignment. *Protein Engineering*, **3**, 565-569.
Waterman, M., and Byers, T. (1985) A dynamic programming algorithm to find all solutions in a neighborhood of the optimum. *Math. Biosciences*, **77**, 179-185.
Zuker, M. (1991) Suboptimal sequence alignment in molecular biology: alignment with error analysis. *J. Mol. Biol.*, **221**, 403-420.

Appendix

Lemma. Suppose $i \in I$, where I is computed by Figure 8. Let the intersection of row i with the given region extend from column j to column k, i.e., $j = L[i]$ and $k = R[i]$. Define $c[i]$ to be $k - j + 1$ and

$$C[i] = \Sigma\{ c[i] \text{ for } q \in I \text{ and } q \le i \}$$

Then $C[i] \le j + 2k$.

Proof. (Induction on i.) If i is the smallest element of I, then $C[i] = c[i] \le k+1 \le 2k$. Otherwise, let i', j' and k' be the values for the previous entry of I, and suppose $C[i'] \le j' + 2k'$. There are two cases to consider.

First suppose that less than one half of the previous line segment overlaps the current segment. More precisely, suppose $j > m$, where $m = (k' + j')/2$, rounded up (midpoint of the previous line segment). See Figure 14. It follows that $k' - j + 1 \le j - j'$. Then

$$
\begin{aligned}
C[i] &= C[i'] + k - j + 1 \\
&\le j' + 2k' + (k - k') + (k' - j + 1) \\
&\le j' + 2k' + 2(k - k') + j - j' \text{ (since } k - k' \ge 0) \\
&= j + 2k.
\end{aligned}
$$

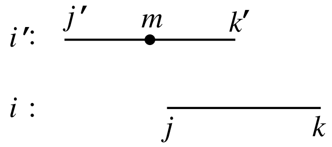

Fig. 14. The case where $j > m$.

Finally, suppose $j \le m$. Rounding *up* in the definition of m implies that $k' - m \le m - j'$. Also, in Figure 15, the area of the outer rectangle is actually more than twice the enclosed area between L and R (by the choice of elements of I), which by monotonicity of L and R is at least the shaded area. Thus the unshaded area of the outer rectangle exceeds the shaded area. Taking a horizontal slice along a row, this means that:

$$
\begin{aligned}
(k - k') + (j - j') &\ge k' - j + 1 + 1 \\
&= k' - m + 1 + m - j + 1 \\
&\ge m - j' + m - j + 1
\end{aligned}
$$

Hence $k - k' \ge 2(m - j) + 1$.

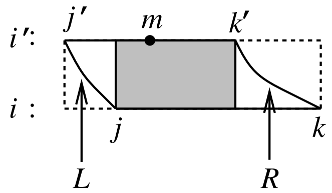

Fig. 15. Picture for the argument that $k - k' \ge 2(m - j) + 1$.

These inequalities yield the desired conclusion, as follows.

$$
\begin{aligned}
C[i] &= C[i'] + k - j + 1 \\
&= C[i'] + (m - j) + (k' - m + 1) + k - k' \\
&\le C[i'] + (m - j) + j - j' + m - j + 1 + k - k' \\
&= C[i'] + j - j' + 2(m - j) + 1 + k - k' \\
&\le j' + 2k' + j - j' + k - k' + k - k' \\
&= j + 2k.
\end{aligned}
$$