Multiple sequence alignment using anchor points through generalized dynamic programming

J. Gracy
gracy@embl-heidelberg.de

J. Sallantin
js@lirmm.fr

[1] European Molecular Biology Laboratory,
Meyerhofstraße 1 - Postfach 102209 - 69012 Heidelberg - Germany.
[2] Laboratoire d'Informatique, de Robotique
et de Micro-électronique de Montpellier,
161 rue Ada - 34392 Montpellier Cedex 5 - France.

Abstract

A generalization of the dynamic programming algorithm applied to the multiple alignment of protein sequences is proposed. The algorithm has two main procedures: (i) local correspondences between sequences - hereafter called anchor points - are selected according to a criterion that combines local and global simlilarity values, (ii) the alignment is constructed recursively by choosing and linking together the optimal anchor points. This multiple sequence alignment algorithm achieves a good compromise between the $O(L^N)$ complexity of the exhaustive dynamic programming approach applied to N sequences of length L and the poor quality of the alignments obtained with methods based on a hierarchical clustering of the sequences.

1 Introduction

Distantly related proteins often display ambiguous relationships that cannot be detected by pairwise sequence alignment algorithms. On another hand, the local relationships between two proteins can be highlighted by using simultaneous comparison of more than two sequences. Therefore, multiple sequence alignments almost always yield to more accurate structural, functional or phylogenetic inferences.
Such a situation can be illustrated on a simple example. Let us consider the three pentapeptides of the figure 1. Although the pairs (S_1, S_2) and (S_2, S_3) share three and two residues respectively, S_1 and S_3 do not display direct similarity. In this way, the completely different polypeptides S_1 and S_3 can be related by chaining the comparisons $S_1 - S_2$ and $S_2 - S_3$.

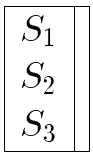

Figure 1: Three pentapeptides.

Therefore, the quality of a local similarity must be estimated using a simultaneous comparison of every pairs of sequences. The major difficulty of the multiple sequence alignment problem comes from the prohibitive complexity of an exhaustive search of the optimal solution. Dynamic programming allows to align sequential objects with a computational time-complexity proportional to the product of their lengths. This method is used to align two protein sequences [10]. Theoretically, multiple alignment of protein sequences can be implemented using the same principle [9]. However, if we have N sequences to align, this generalization leads to a search in an N-dimensional space. Consequently, such an exhaustive solution to the alignment problem can be applied in practice to no more than three sequences. Over three sequences, the existing methods always break down the alignment process in order to obtain a polynomial complexity. A frequently used method consists in reducing the alignment of N sequences to $N-1$ alignments of two sequences [3]. During a preliminary step, the sequences are aligned by pairs. Then, estimation of the similarity between the sequences allows a hierarchical clustering: at each step of the clustering, the two most similar proteins are replaced by a "multiple" sequence formed by the result of their alignment. The process is repeated until the final alignment corresponding to the root of the clustering tree is obtained. When two sequences created during the clustering process are aligned, the mutation costs are averaged over the elementary costs given by every possible pair of residues.
This method has two main weaknesses :

- It only uses $N-1$ out of $N(N-1)/2$ possible sequence comparisons,
- Every error in the alignment occurring at any step of the process will be transmitted to the remaining hierarchical clustering steps.

A cyclical adjustment of the produced alignments has been proposed to solve these problems [13]; however, the updating cycle implies a rather heavy computational cost and convergence to the theoretical optimum is not proved.
Finally, the problem is to find an efficient compromise between the intractable optimality of the exhaustive dynamic programming algorithm and the low reliability of hierarchical clustering methods. In this direction, we describe in the following sections a new algorithm organized in two consecutive steps (figure 2):

- A list of anchor points is selected using local and global similarity criteria based on the simultaneous comparison of all the sequences.
- A multiple sequence alignment which is optimal relatively to the previously selected anchor points is generated using a combination of hierarchical and exhaustive dynamic programming.

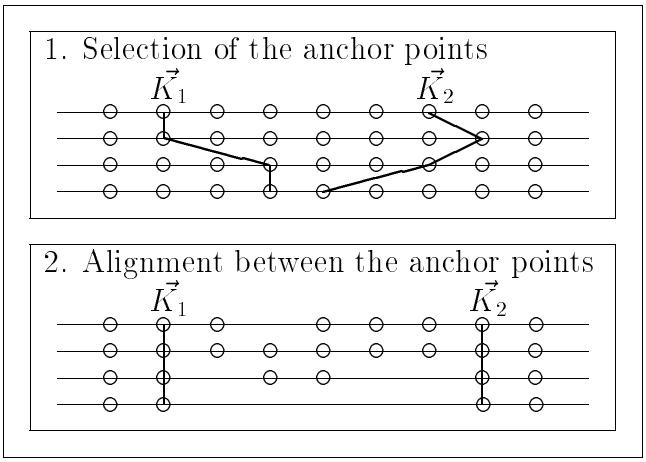

Figure 2: The main steps of the proposed algorithm.

The *a priori* selection of anchor points adds local constraints on the possible alignments and, therefore, leads to a reduction of the search space. This selection speed up dramatically the alignment procedure.
On another hand, the alignment does not result from an iterative clustering of the sequences, but involves a simultaneous comparison of all the sequences. In this way, the preliminary selection step allows the following dynamic programming step to focuse the search onto the most relevant local similarities. This "selection-optimization" process improves of the quality of the alignments as will be shown on weakly homologous domains of the immunoglobulin superfamily.

2 Methods

The selection of the anchor points is performed using two criteria: a local similarity criterion and a global one.

2.1 The local similarity criterion

The principle of the dynamic programming algorithm forces the scoring function to be locally decomposable [11]. This constraint explains the frequent use of additive criteria of homology. However, one can be easily convinced that a cumulated score calculated with mutation and insertion tables does not directly measure the statistical significance of an alignment. In fact, the expected value of a match between two randomly chosen residues is negative for the PAM matrices [4, 1]. Consequently, matches limited to few residues are often over-evaluated with an additive scoring function in comparison to long range similarities.
Using a Monte-Carlo procedure, the statistical likelihood of an alignment can be estimated by the distribution of the obtained scores when the sequences are randomly scrambled. The score can then be normalized using the mean value and the standard deviation of the distribution. Theoretical results about the statistical distribution of the scores have been published [6]. They are limited to alignments without gap.
It is possible to compute a priori the distribution of scores for particular sequences and matrix

of substitution. It is given for N polypeptides of given length L by the L-th power $P_N^L(x)$ of the characteristic polynomial:

$$P_N^1(x) = \sum_s p_N(s).x^s,$$

where $p_N(s)$ is the probability of obtaining an elementary score equal to s when N residues are randomly chosen on each sequence with respect to their composition [2]. The coefficient of the term xs of the polynomial $P_N^L(x)$ corresponds to the probability that the score s will occur on an alignment of n random polypeptides of length L.
There are $\prod_{n=1}^{N}(L_n - L + 1)$ possible alignments of length L on N sequences of length L_n. Let $Q_{N,L}(s)$ be the distribution function of scores on segments of length L, the probability that an alignment of length L randomly chosen in the N sequences will have a score greater than s is:

$$R_{N,L}(s) = 1 - (1 - Q_{N,L}(s))^{\prod_{n=1}^{N}(Ln-L+1)}$$

The likelihood of the local alignment is finally expressed by the local score:

$$LSc(N, s, L) = -ln[R_{N,L}(s)].$$

The following table shows that the statistical significance of a similarity has no linear relationship with the cumulated score s. For example, a similarity score equal to 8 between two segments of length 1 or 15 has the same statistical significance (=12) as a score equal to 11 between two segments of length 5.

s	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	-7	-2	0	1	2	3	4	5	6	6	7	7	8	9	9	9	10	10	10	13	13
2	-15	-3	-1	0	2	3	4	5	6	7	8	9	10	11	11	12	12	13	14	14	15
3	-23	-5	-3	0	1	3	4	5	6	8	9	10	10	11	12	13	14	15	15	16	17
4	-31	-7	-4	-2	0	2	4	5	6	8	9	10	11	12	13	14	15	16	17	17	18
5	-39	-9	-5	-3	0	1	3	5	6	7	9	10	11	12	13	14	15	16	17	18	19
6	-47	-11	-7	-4	-1	1	3	4	6	7	9	10	11	12	14	15	16	17	18	19	20
7	-54	-12	-8	-5	-2	0	2	4	5	7	8	10	11	12	14	15	16	17	18	19	20
8	-61	-14	-9	-6	-3	-1	1	3	5	6	8	9	11	12	14	15	16	17	18	20	21
9	-66	-16	-11	-7	-4	-2	0	2	4	6	8	9	11	12	13	15	16	17	19	20	21
10	-72	-17	-12	-8	-5	-3	-1	2	4	5	7	9	10	12	13	15	16	17	19	20	21
11	-77	-19	-14	-10	-7	-4	-1	1	3	5	7	8	10	12	13	15	16	17	19	20	21
12	-81	-21	-15	-11	-8	-5	-2	0	2	4	6	8	10	11	13	14	16	17	19	20	21
13	-86	-22	-16	-12	-9	-6	-3	-1	1	3	5	7	9	11	12	14	15	17	18	20	21
14	-90	-24	-18	-13	-10	-7	-4	-2	1	3	5	7	9	10	12	14	15	17	18	20	21
15	-95	-26	-19	-15	-11	-8	-5	-3	0	2	4	6	8	10	12	13	15	16	18	19	21

Figure 3: Part of the table of significance scores $-ln((1 - Q_{2,L}(s))/Q_{2,L}(s))$ for $N = 2$. The statistical scores are tabulated for different values of the length L of the compared polypeptides (vertical axis) and of the cumulated score s calculated with the substitution matrix $PAM120$ (horizontal axis).

2.2 The local selection procedure

For N given sequences, the selection procedure consists in finding the anchor points $(k_1, ..., k_N)$ and their length L that maximize the local similarity measure LSc. As suggested by the KMR

algorithm used to find repeated words in strings [7, 8], this selection is performed recursively using a growing number of sequences. This recursive organization reduces the computational time-complexity of the search.
Let the optimal anchor points $(k_1, ..., k_m, L)$ relative to the m first sequences be stored in a heap H_1. During the recurrent step, all the partial anchor points in H_1 are compared to the sequence $m+1$. For each extended anchor point $(k_1, ..., k_m, k_{m+1}, L)$, the boundaries $(k_1, ..., k_{m+1})$ and $(k_1+L-1, ..., k_{m+1}+L-1)$ are locally tuned by adding or subtracting to them the $(m+1)$-tuple $(l, ..., l)$ in order to maximize the score LSc. The best extended anchor points are transferred in a heap H_2. At the end of each recurrent step, H_1 and H_2 are exchanged until all the sequences have been processed.
The heaps are implemented as arrays of pointers [2]. The entry indexed by the score s is pointing on the linked list of anchor points with a local score s. By this way, the anchor points are rapidly retrieved and directly sorted. The size of the heaps is proportional the number of stored anchor points. This number has been empirically fixed to 400 in order to obtain the majority of the optimal anchor points.

2.3 The global similarity criterion

The similarity is variable along the sequences. Two completely different polypeptides can be aligned if their match is consistent with strong similarities occurring in their sequential neighborhood. Consequently, the criterion used to select of the anchor points has to integrate some long range similarity score that measures the quality of alignments with allowed gaps.
In the case of pairwise sequence alignment, the retained similarity measure about an anchor point (i, j) is defined as the score of the optimal alignment constrained to align position i and j. The standard algorithm for pairwise sequence alignment is based on the principle of dynamic programming [10]. Using a local recursive formula, this algorithm permits the construction of the optimal alignment paths joining the beginning of the sequences to a pair of given sequential positions. At the end of the process, optimal alignment path is backtracked starting from the end of the sequences. However, this algorithm can be modified in order to obtain all the scores of the sub-optimal alignments [14]. This complete information is obtained by executing the algorithm in both forward and backward directions onto the sequences.
Let $FSc(i, j)$ be the partial score of the optimal path joining the beginning of the sequences to the position (i, j), let $BSc(i, j)$ be the optimal score calculated from the end of the sequences to the position (i, j), the global score of the optimal path constrained to align positions i and j is :

$$GSc(i, j) = FSc(i, j) + m(i, j) + BSc(i, j)$$

where $m(i, j)$ is the score of the correspondence between positions i and j.

2.4 The global selection procedure

The resulting selection will give a list of the anchor points that maximize the global criterion :

$$GSc(k_1, ..., k_N) = \sum_{i=1}^{N} \sum_{j=1}^{i-1} GSc(k_i, k_j)$$

where N is the number of sequences, and $GSc(k_i, k_j)$ is the similarity score for the positions k_i and k_j of the sequences i and j.
The global selection procedure is a simplified version of the local one. In this case, the length of the anchor point has not to be optimized. Then, the score of $(k_1, ..., k_{m+1})$ can be directly calculated using the decomposition:

$$GSc(k_1, ..., k_{m+1}) = GSc(k_1, ..., k_m) + \sum_{i=1}^{m} GSc(k_i, k_{m+1})$$

2.5 The multiple alignment algorithm

The selected anchor points are given by the union of the lists based on local and global similarity criteria. These N-tuples will serve to limit the number of evaluated alignments. The alignment algorithm consists in selecting an ordered set of N-tuples and constructing the alignment between the successive anchor points.
Dynamic programming can be formally expressed: let E be a set of states partially ordered by a relation '$<$' describing the allowed transitions which are evaluated through a decomposable function F. Then dynamic programming builds the subset of E, completely ordered according to the relation '$<$' that maximizes the score corresponding to the sum of the transition costs given by F (figure 4). In the case of pairwise sequence alignment, the states are corresponding to every possible pairs of sequential positions, the acceptable transitions are described by mutations or insertions onto one of the sequence, and the evaluation function is defined by the substitution matrix and the insertion penalties.
In the case of multiple sequence alignment, the states are corresponding to the selected anchor points. A transition between two anchor points is acceptable if :

- The positions given by the anchor points are found in the same order for each sequences,
- The positions are equal or adjacent on at least one sequence.

	2 sequences	N sequences
States	$\vec{K} = (k_1, k_2)$	$\vec{K} = (k_1, \ldots, k_N)$
Allowed transitions $\vec{K} < \vec{L}$	$\vec{L} = (k_1 + 1, k_2 + 1)$ or $\vec{L} = (k_1 + g, k_2)$ or $\vec{L} = (k_1, k_2 + g)$	$\forall i, k_i \leq l_i$ and $\exists j, k_j \geq l_j - 1$
Evaluation function F	Mutation and insertion scores	Scores obtained with hierarchical dynamic programming

Figure 4: States, allowed transitions and evaluation functions of the pairwise and generalized versions of the dynamic programming algorithm.

The second requirement is optional but its use limits the combinatorial complexity of the algorithm. When the gaps between the positions of two successive anchor points are different, the hierarchical alignment algorithm is used to align the intermediate positions. A transition is evaluated as the sum of the pairwise alignment scores of the segments localized between the

two anchor points.
Using the parameters defined in the previous paragraph, the generalized formula of dynamic programming becomes:

$$Score(\vec{L}) = max_{\{\vec{K}<\vec{L}\}}[Score(\vec{K}) + F(\vec{K}, \vec{L})]$$

where $\vec{K}$ and $\vec{L}$ are two anchor points.

2.6 Computational complexity

Let N be the number of sequences, L be the average length of a sequence, M be the number of retained anchor points, then the complexity of the multiple sequence alignment algorithm based on an exhaustive search [9] is $O(L^N)$. It should be noted that this $O(L^N)$ complexity is broken down in the proposed method into successive phases which have the following polynomial complexities:

- Search of segments with maximal local similarity : $O(N^2L^3)$.
- Construction of sub-optimal alignment paths : $O(N^2L^2)$.
- Global selection procedure : $O(NML)$.
- Multiple sequence alignment by hierarchical clustering : $O(N^2L^2)$.
- Multiple sequence alignment by generalized dynamic programming: $O(Nl^2M^2/L)$, where l is the average length of a transition.

The reduction of the computational time permits the alignment of up to 20 sequences of typical length (around 200 amino acids) to be achieved in reasonable time.

3 Results

The generalized dynamic programming algorithm (GDP) has been tested on the alignment of domains 1, 2, 3 and 4 of CD4 antigen with VL, CH3 and CH4 domains of immunoglobulins. These proteins belong to the immunoglobulin superfamily, but some sequences are very weakly related (less than 20% sequence identity). For this reason, classical sequence comparison methods lead to aligments which are wrongly shifted when compared with the correct structural alignments.
The multiple alignment of the 18 sequences of the test set was obtained on a DECstation 5000/33 by only using 266 seconds of CPU time. The result is compared on the figure 5 with the alignment obtained with the widely used Clustal method based on hierarchical clustering [5]. The correct alignment (according to the protein 3D superposition) of few key-residues is underlined. Incorrectly aligned residues are highlighted with a bold and italic font.
The number of incorrectly aligned key-residues in the Clustal and GDP alignments are respectively 27 and 14. Moreover, a visual inspection of the GDP alignment shows the wrongly shifted residues are mainly located on the C, C' and D strands. These beta strands correspond to a region with a high structural variability. Therefore, these particular errors can be explained by

the excessive structural divergence of the segments.
The only remaining error of the GDP alignment corresponds to the buried tryptophan of strand B. In fact, the residue has been conserved while the corresponding strand has been shifted inside the structural core. This particular case illustrates the frequently observed difference between the optimal sequence alignment and the real structural superposition. It should be noted that 17 errors are remaining on the strands B and C of the Clustal alignment.

4 Conclusion

The proposed method should be useful to align more than three sequences with less than 35% sequence identity on average. If the sequences show a high similarity, a simpler and faster method for the alignment will be sufficient. On the other hand, if only two or three sequences have to be aligned, the optimal solution given by the exhaustive dynamic programming method will be preferred. The described algorithms can be easily transposed to nucleic acid sequences. It should be noted however that lengthening the sequence by a factor three will have heavy combinatorial consequences.
The evolution process based on successive divergences could explain the heterogeneity of the similarity between sequences. An algorithm capable of managing partial anchor points is currently under development in order to increase the speed and the reliability of the process. Also, additional flexibilities to allow the user to introduce its own anchor points are under investigation.

5 Acknowledgement

This work has been supported by Framentec-Cognitech and the Centre National de la Recherche Scientifique.

References

[1] S.F. Altschul, "Amino acid substitution matrices from an information theoritic perspective," *J. Mol. Biol.*, 219, pp. 555-565, 1991.

[2] D.J. Bacon, W.F. Anderson, "Multiple sequence alignment," *J. Mol. Biol.*, 191, pp. 153-161, 1986.

[3] G. Barton, M.J.E. Sternberg, "Flexible protein sequence patterns," *J. Mol. Biol.*, 198, pp. 327-337, 1987.

[4] M.O. Dayhoff M.O., R.M. Schwartz, "Atlas of Protein Sequence and Structure," *Nat. Biomed. Res. Found.*, Washington, 1979.

[5] D.G. Higgins, A.J. Bleasby, R. Fuchs, "Clustal V: improved software for multiple sequence alignment," *CABIOS*, 8, pp. 189-191, 1992.

[6] S. Karlin, S.F. Altschul, "Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes," *P.N.A.S.*, 87, pp. 2264-2268, 1990.

[7] R.M. Karp, R.E. Miller, A.L. Rosenberg, "Rapid identification of repeated patterns in strings, trees and arrays," *Proc. 4th Annu. ACM Symp. Theory of Computing*, pp. 125-136, 1972.

[8] A. Landraud, J.F. Avril, P. Chrétienne, "An algorithm for finding a common structure shared by a family of strings," *IEEE Trans. on P.A.M.I.*, 2, 890-895, 1989.

[9] M. Murata, "Three-way Needleman-Wunsch algorithm," *Methods in enzymology*, 183, pp. 365-379, 1990.

[10] S.B. Needleman, C.D. Wunsch, "General method applicable for the search for similarities in the amino acid sequence of two proteins," *J. Mol. Biol.*, 48, pp. 443-453, 1970.

[11] J.C. Simon, "La reconnaissance des formes par algorithmes," *Masson Ed.*, 1984.

[12] R.F. Smith, T.F. Smith, "Automatic generation of primary patterns from sets of related sequences," *P.N.A.S.*, 87, pp. 118-122, 1990.

[13] S. Subbiah, S.C. Harrison, "A method for multiple sequence alignment with gaps," *J. Mol. Biol.*, 209, pp. 539-548, 1989.

[14] M. Zuker, "Suboptimal sequence alignment in molecular biology. Alignment with error analysis," *J. Mol. Biol.*, 221, pp. 403-420, 1991.

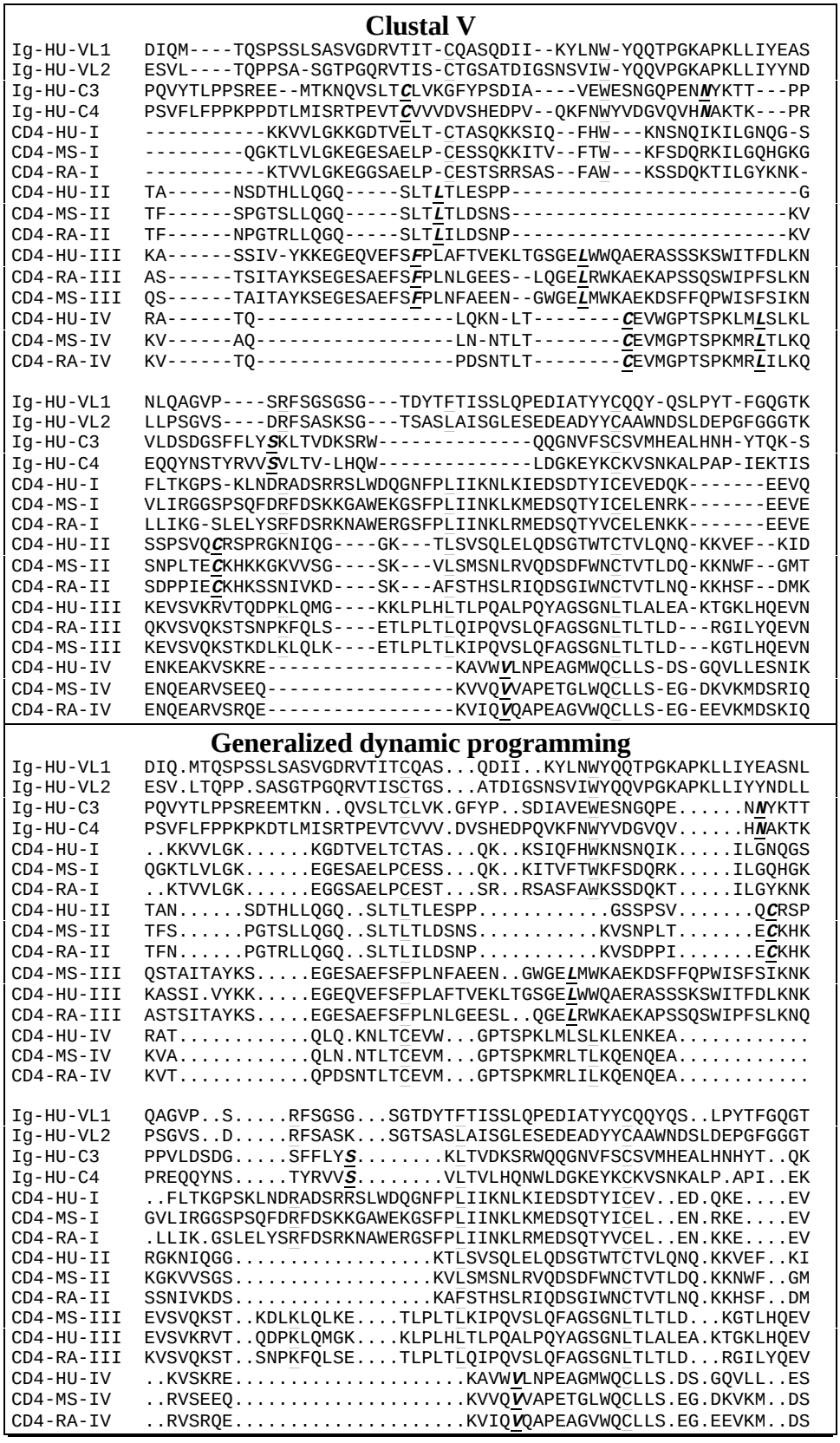

Figure 5: Comparison between Clustal V and the GDP algorithm (see text).