Multiple sequence alignment with user-defined constraints

Burkhard Morgenstern[1], Sonja J. Prohaska[2], Nadine Werner[1], Jan Weyer-Menkhoff[1], Isabelle Schneider[1], Amarendran R. Subramanian[3], Peter F. Stadler[2]

[1]Universität Göttingen, Institut für Mikrobiologie und Genetik, Abteilung für Bioinformatik, Goldschmidtstr. 1, D-37077 Göttingen, Germany. [2]Universität Leipzig, Institut für Informatik und Interdisziplinäres Zentrum für Bioinformatik, Kreuzstrasse 7b, D-04103 Leipzig, Germany. [3]Universität Tübingen, Wilhelm-Schickard-Institut für Informatik, Sand 13, D-72076 Tübingen, Germany.

Abstract

In many situations, *automated* multi-alignment programs are not able to correctly align families of nucleic acid or protein sequences. Difficult cases comprise not only distantly related sequences but also tandem duplications independent of their evolutionary age. Frequently, additional biological information is available that establishes homologies at least in parts of the sequences based on structural or functional consideration. In the present paper, we describe a *semi-automatic* approach to multiple sequence alignment in which the user can explicitly specify parts of the sequences that are biologically related to each other. Our software program uses these sites as *anchor points* and creates a multiple alignment that respects these user-defined constraints and hence should be biologically more plausible than alignments produced by fully automated procedures. We apply our approach to genomic sequences adjacent to the *Hox* genes. As a by-product, we obtain not only useful insights for the further development of alignment algorithms, but also an improved approach to phylogenetic footprinting.

Introduction

A large number of multi-alignment programs have been developed during the last twenty years, e.g. [31, 16, 23, 14, 10]; see [22] for an overview. The performance of these tools has been studied extensively [13, 24, 32]. Whatever their respective advantages and shortcomings are, it should be clear that no automatic alignment method can be expected to produce *biologically* meaningful alignments in all possible situations. At best, these methods can give us a good guess about *possible* homologies in a given set of sequences. Therefore, it is common practice to *manually* improve alignments produced by standard software tools and/or to compare the output from different software programs [19].

Practically all existing alignment methods are fully *automated*, i.e., they construct alignments following a fixed set of algorithmical rules. For most software tools, a number of program parameters such as gap penalties can be adjusted, but usually there is no way of *directly* influencing the alignment procedure. Such automatic alignment methods are clearly necessary and appropriate where large amounts of data are to be processed or in situations where no additional expert information about the sequence data is available. However, if a researcher is familiar with a specific sequence family under study, he or she may already know certain parts of the sequences that are functionally or phylogenetically related and should therefore be aligned to each other. In situations where automated programs *fail* to align these regions correctly, it is useful to have an alignment method that can incorporate user-defined homology information and would then align the remainder of the sequences automatically, respecting these user-specified *constraints*.

Multiple alignment under *constraints* has been proposed by Myers *et al.* [21] and Sammeth *et al.* [28]. The multi-alignment program `dialign` [18, 16] has a new option that can be used to calculate alignments under pre-defined constraints. Originally, this program feature has been introduced to reduce the alignment search space and program running time for large genomic sequences [4, 20]. Herein, we describe our constrained-alignment approach in detail using a previously introduced set-theoretical alignment concept. We apply our method to genomic sequences of the *Hox* gene clusters. For these sequences, the default version of `dialign` produces serious mis-alignments where entire genes are incorrectly aligned but meaningful alignments can be obtained if the known gene boundaries are used as anchor points. Interestingly, our anchoring procedure not only improves the *biological* quality of the output alignments but can also lead to alignments with significantly better *numerical* scores. This demonstrates that the heuristic optimization procedure used in `dialign` may produce alignments that are far below the optimal alignment for given data set. The latter result has important implications for the further development of alignment algorithms.

Alignment of tandem duplications

There are typical situations where automated alignment procedures tend to produce biologically incorrect aligments. The most obvious case is alignment of *distantly* related sequences where similarity at the primary sequence level is low and homologies can be obscured by spurious similarities. Moreover, most existing alignment programs can be confused by *duplications* within the input sequences. Here, *tandem duplications* are particularly hard to align, see e.g. [3]. Specialized software tools have been developed to cope with the problems caused by sequence duplications [11]. For the segment-based alignment program `dialign`, the situation is as follows. As described in previous publications, the program constructs pairwise and multiple alignments from pairwise local sequence similarities, so-called *fragment alignments* or *fragments* [16, 18]. A fragment is defined as an un-gapped pair of equal-length segments from two of the input sequences. Based on statistical considerations, the program assigns a *weight score* to each possible fragment and tries to find a consistent collection of fragments with maximum total score. For pairwise alignment, a *chain* of fragments with maximum score can be identified [17]. For multiple sequence sets, all possible pairwise alignments are performed and fragments contained in these pairwise alignments are integrated *greedily* into a resulting multiple alignment.

As indicated in Figure 1, tandem duplications can create various problems for the above outlined alignment approach. In the following, we consider a motif that is duplicated in one or several of the input sequences $S_1, \ldots, S_k$. For simplicity, let us assume that our sequences do not share any significant similarity outside the motif. Moreover, we assume that the degree of similarity among all instances of the motif is roughly comparable. There are no difficulties if the motif duplicated in a *pair* of input sequences, i.e if one has instances $M_1^{(1)}$ and $M_1^{(2)}$ of the motif in sequence S_1 and instances $M_2^{(1)}$ and $M_2^{(2)}$ of the same motif in a sequence S_2. In such a situation, the segment approach will correctly align $M_1^{(1)}$ to $M_2^{(1)}$ and $M_1^{(2)}$ to $M_2^{(2)}$. A correct alignment will be produced even if $M_1^{(1)}$ exhibits stronger similarity to $M_2^{(2)}$ than to $M_2^{(1)}$ since, for pairwise alignment, the program identifies a chain with maximum *total* score and a greedy heuristics is applied only for multiple alignment where an exact solution is not feasible. The trouble starts if a tandem duplication $M_1^{(1)}, M_1^{(2)}$ occurs in S_1 but only one instance of the motif M_2 is present in S_2. Here, it can happen that the beginning of M_2 is aligned to the beginning of $M_1^{(1)}$ and the end of M_2 is aligned to the end of $M_1^{(2)}$ as in Figure 1 (B).

The situation is even worse for multiple alignment. Consider, for example, three sequences S_1, S_2, S_3, where two instances $M_1^{(1)}, M_1^{(2)}$ of our motif occur in S_1 while S_2 and S_3 each contain only one instance of the motif M_2 and M_3, respectively. Under the above assumptions, a *biologically* meaningful alignment of these sequences would certainly align S_2 to S_3, and both motifs would be aligned either to $M_1^{(1)}$ or to $M_1^{(2)}$ – depending on the degree of similarity of S_2 and S_3 to $M_1^{(1)}$ and $M_1^{(2)}$, respectively. Note that such an alignment would also receive a high *numerical* score since it would involve three pairwise alignments of motifs. However, since the pairwise alignments are carried out independently for each sequence pair, it may happen that the first instance of the motif in sequence S_1, $M_1^{(1)}$ is aligned to M_2 but the second instance $M_1^{(2)}$ is aligned to M_3 in the respective pairwise alignments of S_1 with S_2 and S_3 as in Figure 1 (C). Thus, the correct alignment of M_2 and M_3 will be *inconsistent* to the first two pairwise alignments. Depending on the degree of similarity among the motifs, alignment of $M_1^{(2)}$ and M_3 may be rejected in the greedy algorithm, so these motifs may not be aligned in the resulting multiple alignment. It is easy to see that the resulting multiple alignment would not only be biologically questionable, but would also obtain a numerically lower score as it would involve only *two* pairwise alignments of the motif.

Multiple alignment with user-defined anchor points

To overcome the above mentioned difficulties, and to deal with other situations that cause problems to alignment programs, we use a semi-automatic *anchored* alignment procedure where the user can specify an arbitrary number of *anchoring points* in order to guide the subsequent alignment procedure. Each anchor point consists of a pair of equal-length segments of two of the input sequences. An anchor point is therefore characterized by five coordinates: the two *sequences* involved, the *starting positions* in the sequences and the *length* of the anchored segments. As a sixth parameter, our method requires a *score* that determines the *priority* of

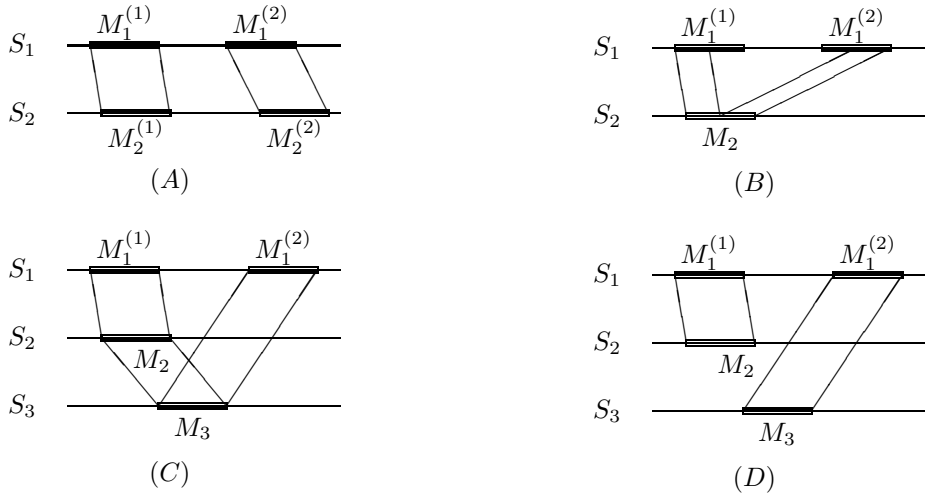

Figure 1: Possible mis-alignments caused by tandem duplications in the segment-based alignment approach (`dialign`). We assume that various instances of a motif are contained in the input sequence set and that the degree of similarity among the different instances is approximately equal. For simplicity, we also assume that the sequences do not share any similarity outside the conserved motif. (A) If a tandem duplication occurs in two sequences, the correct alignment will be found since the algorithm identifies a *chain* of local alignments with maximum *total* score. (B) If a motif is duplicated in one sequence but only one instance M_2 is contained in the second sequence, it may happen that M_2 is split up and aligned to different instances of the motif in the first sequence. (C) If the motif is duplicated in one sequences and one instance is contained in sequences two and three, respectively, *consistency* conflicts can occur. In this case, local similarities identified in the respective pairwise alignments cannot be integrated into one single output alignment. Here, DIALIGN uses a *greedy* heuristics to select a consistent *subset* of these pair-wise similarities. Depending on the degree of similarity among the instances of the motif, the greedy approach can lead to serious mis-alignments (D).

the anchor point. The latter parameter is necessary, since it is in general not meaningful to use *all* anchors proposed by the user so the algorithm needs to select a suitable *subset* of the proposed anchor points.

The selected anchor points are used to constraint the subsequent alignment procedure in the following way. If a position x in sequence S_i is *anchored* with a position y in sequence S_j through one of the anchor points this means that y is the *only* position from S_j that can be aligned to x. Whether or not x is will actually be aligned to y depends on the degree of local sequence similarity among the sequences around positions x and y. If no statistically significant similarity can be detected, x and y may remain un-aligned. Moreover, anchoring x and y means that positions strictly to the left (or strictly to the right) of x in S_i can be aligned only to positions strictly to the left (or strictly to the right) of y in S_j – and vice versa. Obviously, these relations are *transitive*, so if position x is anchored with position y_1, y_1 is to the left of another position y_2 in the same sequence, and y_2, in turn, is aligned to a position z, then positions to the left of x can be aligned only to positions to the left of z etc.

Algorithmically , anchor points are treated by `dialign` in the same way as *fragments* (= segment pairs) in the greedy procedure for multi-alignment. By transitivity, a set Anc of anchor points defines a *quasi partial order relation* $\preceq_{\mathsf{Anc}}$ on the set X of all positions of the input sequences - in exactly the same way as an alignment Ali induces a quasi partial order relation $\preceq_{\mathsf{Ali}}$ on X as described in [18, 1]. Formally, we consider an alignment Ali as well as a set of anchor points Anc as an *equivalence relation* defined on the set X of all positions of the input sequences. Next, we consider the partial order relation $\preceq$ on X that is given by the 'natural' ordering of positions within the sequences. In order-theoretical terms, $\preceq$ is the *direct sum* of the *linear* order relations defined on the individual sequences. The partial order relation $\preceq_{\mathsf{Anc}}$ is then defined as the *transitive closure* of the union $\preceq \cup \mathsf{Anc}$.

It makes sense to require a set of anchor points for a given data set to be *consistent*. Informally, this means, that it would be possible to *align* the anchored segment pairs to each other without leading to contradictions. In our set-theoretical setting, a relation R on X is called consistent if all restrictions of the tansitive closure of $\preceq \cup R$ to the idividual sequences *coincides* with their respective 'natural' linear orderings. In our anchored-alignment approach, we are looking for an alignment Ali such that the union $\mathsf{Ali} \cup \mathsf{Anc}$ is consistent. Thus, our optimization problem is to find an alignment Ali with maximum score – under the additional condition that the set-theoretical union $\mathsf{Ali} \cup \mathsf{Anc}$ is consistent. This makes sense only if the set Anc of anchor points is consistent itself. Since a user-defined set of anchor points cannot be expectd to be consistent, the first step

in our anchoring procedure is to select a consistent *subset* of the anchor points proposed by the user. To this end, the program uses the same greedy approach that it applies in the optimization procedure for multiple alignment. That is, each anchor point is associated with some user-defined score, and the program accepts input anchor points in order of decreasing scores – provided they are consistent with the previously accepted anchors.

The greedy selection of anchor points makes it possible for the user to *prioritize* potential anchor points according to arbitrary user-defined criteria. For example, one may use known gene boundaries in genomic sequences to define anchor points as we did in the *Hox* gene example described below. In addition, one may want to use *automatically* produced local alignments as anchor points to speed up the alignment procedure as outlined in [4]. While the set of gene boundaries should be consistent – as long as the relative ordering among the genes is conserved – the automatically created anchor points may well be *inconsistent* with those 'biologically defined' anchors (or inconsistent with each other). In this situation, it would make sense to first accept the known gene boundaries as anchors and then to use the automatically created local alignments, under the condition that they are consistent with the known gene boundaries. So in this case, one could use local alignment scores to score the *automatically* created anchor points, while one would assign (arbitrary) higher scores to the *biologically* verified gene boundaries.

Applications to *Hox* gene clusters

As explained above, tandem duplications pose a hard problem for automatic alignment algorithms. Clusters of such paraloguous genes are therefore particularly hard to align. As a real-life example we consider here the *Hox* gene clusters of vertebrates. *Hox* genes code for homeodomain transcription factors that regulate the anterior/posterior patterning in most bilaterian animals [7, 15]. This group of genes, together with the so-called *ParaHox* genes, arose early in metazoan history from a single ancestral "*UrHox* gene" [8]. Their early evolution was dominated by a series of tandem duplications. As a consequence, most bilaterians share at least eight distinct types (in arthropods, and 13 or 14 in chordates), usually referred to as paralogy classes. These *Hox* genes are usually organized in tightly linked clusters such that the genes at the 5'end (paralogy groups 9-13) determine features at the posterior part of the animal while the genes 3'end (paralogy groups 1-3) determine the anterior patterns.

In contrast to all known invertebrates, all vertebrate lineages investigated so far exhibit multiple copies of *Hox* clusters that presumably arose through genome duplications in early vertebrate evolution and later in the actinopterygian (ray finned fish) lineage [12, 9, 2]. These duplication events were followed by massive loss of the duplicated genes in different lineages, see e.g. [27] for a recent review on the situation in teleost fishes.

The individual *Hox* clusters of gnathostomes have a length of some 100,000nt and share besides a set of homologous genes also a substantial amount of conserved non-coding DNA [5] that predominantly consists of transcription factor binding sites. Most recently, however, some of these "phylogenetic footprints" were identified as microRNAs [33].

Fig. 2 shows four of the seven *Hox* clusters of the pufferfish *Takifugu rubripes*. Despite the fact that the *Hox* genes within a paralogy group are significantly more similar to each other than to members of other paralogy groups, there are several features that make this dataset particularly difficult and tend to mislead automatic alignment procedures: (1) Neither one of the 13 *Hox* paralogy groups nor the *Evx* gene is present in all four sequences. (2) Two genes, *HoxC8a* and *HoxA2a* are present in only a single sequence. (3) The clusters have different sizes and numbers of genes (33481nt to 125385nt, 4 to 10 genes).

We observe that without anchoring `dialign` mis-aligns many of of the *Hox* genes in this examples by matching blocks from one *Hox* gene with parts of a *Hox* gene from a different paralogy group. As a consequence, genes that should be aligned, such as *HoxA10a* and *HoxD10a*, are not aligned with each other. Anchoring the alignment, maybe surprisingly, increases the number of columns that contain aligned sequence positions from 3870 to 4960, i.e., by about 28%, see Tab. 1. At the same time, the CPU time is reduced by almost a factor of 3.

We investigated not only the *biological* quality of the anchored and non-anchored alignments but also looked at their *numerical* scores. Note that in `dialign`, the score of an alignment is defined as the sum of weight scores of the fragments it is composed of. For some sequence sets we found that the score of the anchored

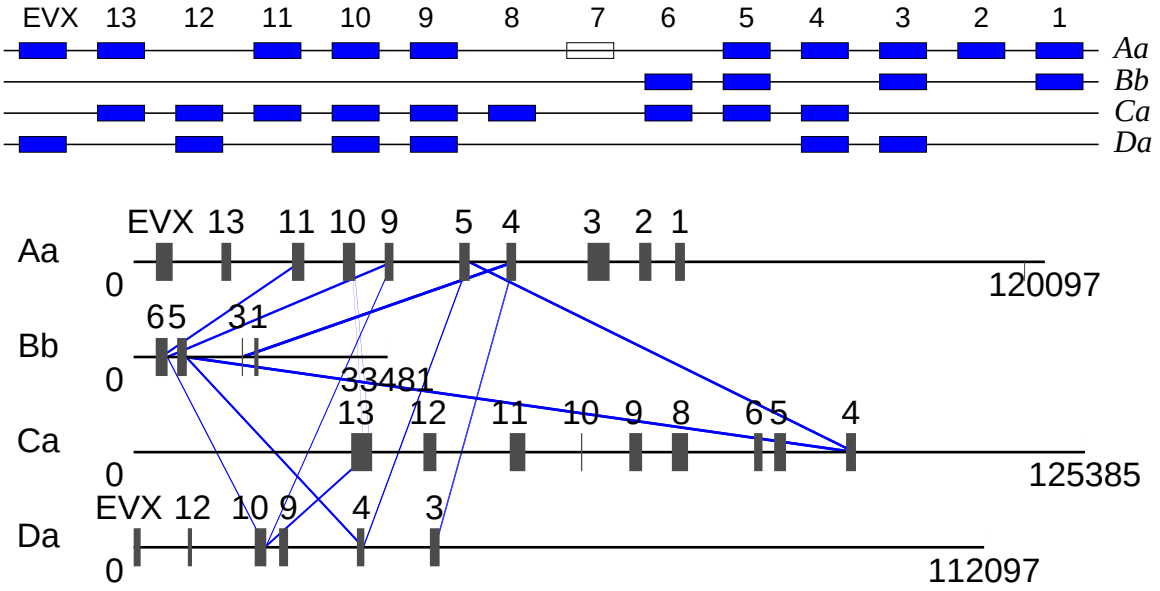

Figure 2: (top) The pufferfi sh *Takifugu rubripes* has 7 *Hox* clusters of which we use four in our computational example. The *Evx* gene, another homedomain transcription factor is usually liked with the *Hox* genes and can be considered as part of the *Hox* cluster. The paralogy groups are indicated. Filled boxes indicates intact *Hox* genes, the open box indicates a *HoxA7a* pseudogene [6].
(bottom) Result of a `dialign2` run on the nucleic acid sequences without anchoring. The diagram represents sequences and gene positions to scale. All incorrectly aligned segments (defi ned as parts of a gene that are aligned with parts of gene from a different paralogy group) are indicated by lines between the sequences.

alignment was above the non-anchored alignment while for other sequences, the non-anchored score exceeded the anchored one. For example, with the sequence set shown in Figure 2, the alignment score of the – biologically more meaningful – anchored alignment was $> 13\%$ *below* the non-anchored alignment (see Table 2). In contrast, another sequence set with fi ve *HoxA* cluster sequence (TrAa, TnAa, DrAb, TrAb, TnAb) from three teleost fi shes (*Takifugu rubripes*, Tr; *Tetraodon nigroviridis*, Tn; *Danio rerio*, Dr), yields an anchored alignment score that is some 15% *above* the non-anchored score.

Anchored Alignments for Phylogenetic Footprinting

Evolutionarily conserved non-coding genomic sequences represent a potentially rich source for the discovery of gene regulatory regions. Since these elements are subject to stabilizing selection they evolve much more slowly than adjacent non-functional DNA. These so-called phylogenetic footprints can be detected by

Table 1: Aligned sequence positions that result from fragment aligments in the Fugu *Hox* cluster example. We count here the numbers of columns containing uppercase letters in the `dialign` output. The number of columns in which two or three sequences are aligned increases when more anchors are used, while the number of columns in which all sequences are aligned decreases because the biologically correct alignment should not contain such columns. CPU times are measured on a PC with two Intel Xeon 2.4GHz processors and 1Gbyte of RAM.

anchor	alignment length	aligned sequences. 2	3	4	CPU time	score
none	281759	2958	668	244	4:22:07	1166
genes	252346	3674	1091	195	1:18:12	1007
genes + `blastz` hits	239326	4036	1139	33	0:19:32	742

Table 2: Aligned sequence positions *outside* the coding regions and introns (first column) in the Fugu example of Fig. 2. The second column gives the number of sequence positions to which `dialign` adds an additional sequence to a `tracker` footprint cluster. The third column lists new footprints, from which, as in `tracker`, low complexity regions were removed.

anchor	aligned positions		
	noncoding	addt'l seq.	new footprint
none	1546	0	618
genes	1686	39	694
genes + `blastz` hits	2433	39	841

comparison of the sequences surrounding orthologous genes in different species [30]. Alignment algorithms, including `dialign`, were advocated for this task. As the example in the previous section shows, however, anchoring the alignments becomes a necessity in applications to clusters of paralogous genes.

Interspersed repeats pose an additional problem for unconstrained alignments. While these are normally removed ("masked") using e.g. `RepeatMasker`, they need to be taken into account in the context of phylogenetic footprinting: if a sequence motif is conserved hundreds of millions of years it may well have become a regulatory region even if it is (similar to) a repetitive sequence in some of the organisms under consideration.

The program `tracker` [26] was designed specifically to search for conserved non-coding sequences in large gene clusters. It is based on a similar philosophy as segment based alignment algorithms. The `tracker` program uses `blastz` [29] with non-stringent settings and restricted to homologous intergenenic regions and parallel strands to compute pairwise local alignments of all input sequences. These are post-processed to remove e.g. low-complexity regions. Effectively, `tracker` thus computes alignments anchored at the genes. The pairwise alignments are then combined into overlapping cluster based on their positions alone. Here the approach suffers from the same problem as `dialign`, which is, however, resolved in a different way: instead of producing a single locally optimal alignment, `tracker` lists all maximal compatible sets of pairwise alignments. For the case of Fig. 1(C), for instance, we obtain both $M_1^{(1)}M_2M_3$ and $M_1^{(2)}M_2M_3$. Since this step is performed based on the overlap of sequence intervals without explicitly considering the sequence information at all, `tracker` is very fast as long as the number of conflicting pairwise alignments remains small. In the final step `dialign` is used to re-calulate sequence alignments from the subsequences that belong to individual clusters.

The combination of `blastz` and an anchored version of `dialign` appears to be a very promising approach for phylogenetic footprinting. We have noticed in ref. [25] that `dialign` is more sensitive than `tracker` in general. A combination of anchoring at appropriate genes (with maximal weight) and `blastz` hits (with smaller weights proportional e.g. to $-\log E$ values) reduces the CPU requirements by more than an order of magnitude. While this is still much slower than `tracker` (20min vs. 40s) it increases the sensitivity of the approach by about $30-40\%$ in the Fugu example, Tab. 2. Work in progress aims at improving the significance measures for local multiple alignments. A more thorough discussion of anchored segment-bases alignments to phylogenetic footprinting will be published elsewhere.

Conclusions

Automated alignment procedures are based on simple algorithmical rules. For a given set of input sequences, they try to find an alignment with maximum score in the sense of some underlying objective function. The two basic questions in sequence alignment are therefore (a) to define an appropriate objective function and (b) to design an efficient optimization algorithm that finds optimal or at least near-optimal alignments with respect to the chosen objective function. Most multi-alignment programs are using *heuristic* optimization algorithms, i.e. they are, in general, not able to find the mathematically optimal alignment with respect to the objective function. An objective function for sequence alignment should assign *numerically* high scores to *biologically* meaningful alignments. However, it is clearly not possible to find a *universally* applicable objective function that would give highest numerical scores to the biologically correct alignments in all possible situations. This is the main reason why alignment programs may fail to produce biologically reasonable output alignments.

In fact, the impossibility to define a universal objective function constitutes a fundamental limitation for *all* automated alignment algorithms.

Often a user is already familiar with a sequence family that he or she wants to align, so some knowledge about existing sequence homologies may be available. Such expert knowledge can be used to direct an otherwise automated alignment procedure. To facilitate the use of expert knowledge for sequence alignment, we proposed an *anchored alignment* approach where known homologies can be used to restrict the alignment search space. This clearly improves the quality of the alignments and reduces the program running time, in particular in situations where automatic procedures are unlikely to produce meaningful alignments. For the *Hox* gene clusters that we analyzed, the non-anchored version of `dialign` produced serious mis-alignments. Using the anchored alignment approach with the known gene boundaries as anchor points guarantees a correct alignment of these genes and at the same time reduces the running time by almost a factor of 4.

There are two possible reasons why automated alignment procedures may fail to produce biologically correct alignments:

(a) The chosen objective function may not be in accordance with biology, i.e., it may assign mathematically high scores to biologically wrong alignments. In this case, even efficient optimization algorithms would lead to meaningless alignments.

(b) The mathematically optimal alignment is biologically meaningful, but the employed heuristic optimization procedure is not able to find the alignment with highest score.

For the further development of alignment algorithms, it is crucial to find out which one of these reasons is to blame for mis-alignments produced by existing software programs. If (a) is often observed for an alignment program, efforts should be made to improve its underlying objective function. If (b) is the case, the biological quality of the output alignments can be improved by using a more efficient optimization algorithm. For `dialign`, it is unknown how close the produced alignments come to the numerically optimal alignment – in fact, it is possible to construct example sequences where `dialign`'s greedy heuristics produces alignments with arbitrarily low scores compared with the possible optimal alignment.

In the Fugu example, Figure 2, the *numerical* alignment score of the (anchored) correct alignment was 13% below the score of the non-anchored alignment. All sequences in Figure 2 contain only subsets of the 13 *Hox* paralogy groups, and different sequences contain different genes. For such a data set, it is unlikely that any reasonable objective function would assign an optimal score to the biologically correct alignment. The only way of producing good alignments in such situations is to *force* a program to align certain known homologies to each other. With our anchoring approach we can do this, for example by using known gene boundaries as *anchor points*. In contrast, in the teleost *HoxA* cluster example the numerical score of the anchored alignment was around 15% *higher* than the score of the non-anchored alignment. This demonstrates that the greedy optimization algorithm used by `dialign` can lead to results with scores far below the optimal alignment for a given data set. In such situations, improved optimization algorithms may lead not only to mathematically higher-scoring alignments but also to alignments that are closer to the biologically correct alignment. This latter example suggests that much can be gained by developing more efficient optimization strategies for `dialign`, i.e. optimization algorithms that come closer to the mathematically optimal alignment. We will use our anchored-alignment approach systematically to study the efficiency of objective functions and optimization algorithms for the segment-based approach to multiple sequence alignment.

Acknowledgements

The work was supported by DFG grant MO 1048/1-1 to BM, IS and JWM and by DFG Bioinformatics Initiative BIZ-6/1-2 to SJP and PFS.

References

[1] S. Abdeddaïm and B. Morgenstern. Speeding up the DIALIGN multiple alignment program by using the 'greedy alignment of biological sequences library' (GABIOS-LIB). *Lecture Notes in Computer Science*, 2066:1–11, 2001.

[2] A. Amores, A. Force, Y. L. Yan, L. Joly, C. Amemiya, A. Fritz, R. K. Ho, J. Langeland, V. Prince, Y. L. Wang, M. Westerfield, M. Ekker, and J. H. Postlethwait. Zebrafish *Hox* clusters and vertebrate genome evolution. *Science*, 282:1711–1714, 1998.

[3] G. Benson. Sequence alignment with tandem duplication. *J. Comp. Biol.*, 4:351–367, 1997.

[4] M. Brudno, M. Chapman, B. Göttgens, S. Batzoglou, and B. Morgenstern. Fast and sensitive multiple alignment of large genomic sequences. *BMC Bioinformatics*, 4:66, 2003.
`http://www.biomedcentral.com/1471-2105/4/66`.

[5] C.-h. Chiu, C. Amemiya, K. Dewar, C.-B. Kim, F. H. Ruddle, and G. P. Wagner. Molecular evolution of the *HoxA* cluster in the three major gnathostome lineages. *Proc. Natl. Acad. Sci. USA*, 99:5492–5497, 2002.

[6] C.-H. Chiu, K. Dewar, G. P. Wagner, K. Takahashi, F. Ruddle, C. Ledje, P. Bartsch, J.-L. Scemama, E. Stellwag, C. Fried, S. J. Prohaska, P. F. Stadler, and C. T. Amemiya. Bichir *HoxA* cluster sequence reveals surprising trends in rayfinned fish genomic evolution. *Genome Res.*, 14:11–17, 2004.

[7] D. Duboule and P. Dollé. The structural and functional organization of the murine HOX gene family resembles that of *Drosophila* homeotic genes. *EMBO J.*, 8:1497–1505, 1989.

[8] D. E. K. Ferrier and P. W. H. Holland. Ancient origin of the *Hox* gene cluster. *Nat. Rev. Genet.*, 2:33–38, 2001.

[9] J. Garcia-Fernández and P. W. Holland. Archetypal organization of the amphioxus *Hox* gene cluster. *Nature*, 370:563–566, 1994.

[10] C. Grasso and C. Lee. Combining partial order alignment and progressive multiple sequence alignment increases alignment speed and scalability to very large alignment problems. *Bioinformatics*, in press.

[11] J. Heringa. Detection of internal repeats: how common are they? *Curr. Opin. Struc. Biol.*, 8:338–345, 1998.

[12] P. W. H. Holland, J. Garcia-Fernández, N. A. Williams, and A. Sidow. Gene duplication and the origins of vertebrate development. *Development*, (Suppl.):125–133, 1994.

[13] T. Lassmann and E. L. Sonnhammer. Quality assessment of multiple alignment programs. *FEBS Letters*, 529:126–130, 2002.

[14] C. Lee, C. Grasso, and M. F. Sharlow. Multiple sequence alignment using partial order graphs. *Bioinformatics*, 18(3):452–464, 2002.

[15] W. McGinnis and R. Krumlauf. Homeobox genes and axial patterning. *Cell*, 68:283–302, 1992.

[16] B. Morgenstern. DIALIGN 2: improvement of the segment-to-segment approach to multiple sequence alignment. *Bioinformatics*, 15:211–218, 1999.

[17] B. Morgenstern. A simple and space-efficient fragment-chaining algorithm for alignment of DNA and protein sequences. *Applied Mathematics Letters*, 15:11–16, 2002.

[18] B. Morgenstern, A. Dress, and T. Werner. Multiple DNA and protein sequence alignment based on segment-to-segment comparison. *Proc. Natl. Acad. Sci. USA*, 93:12098–12103, 1996.

[19] B. Morgenstern, S. Goel, A. Sczyrba, and A. Dress. AltAVisT: a WWW server for comparison of alternative multiple sequence alignments. *Bioinformatics*, 19:425–426, 2003.

[20] B. Morgenstern, O. Rinner, S. Abdeddaïm, D. Haase, K. Mayer, A. Dress, and H.-W. Mewes. Exon discovery by genomic sequence alignment. *Bioinformatics*, 18:777–787, 2002.

[21] G. Myers, S. Selznick, Z. Zhang, and W. Miller. Progressive multiple alignment with constraints. *J. Computational Biology*, 3, 1996.

[22] C. Notredame. Recent progress in multiple sequence alignment: a survey. *Pharmacogenomics*, 3:131–144, 2002.

[23] C. Notredame, D. Higgins, and J. Heringa. T-Coffee: a novel algorithm for multiple sequence alignment. *J. Mol. Biol.*, 302:205–217, 2000.

[24] D. A. Pollard, C. M. Bergman, J. Stoye, S. E. Celniker, and M. B. Eisen. Benchmarking tools for the alignment of functional noncoding DNA. *BMC Bioinformatics*, 5:6, 2004.
`http://www.biomedcentral.com/1471-2105/5/6`.

[25] S. J. Prohaska, C. Fried, C. Flamm, and P. F. Stadler. Phylogenetic footprint patterns in large gene clusters. Technical report, University of Leipzig, Bioinformatics Group, 2003. Extended Abstract: Proceedings of the German Conference on Bioinformatics. Volume II. H.-W. Mewes, V. Heun, D. Frishman, S. Kramer (Eds.), belleville Verlag Michael Farin, Mnchen, 2003, p.145-147. Full text:
`http://www.bioinf.uni-leipzig.de/Publications/POSTERS/P-005abs.pdf`.

[26] S. J. Prohaska, C. Fried, C. Flamm, G. Wagner, and P. F. Stadler. Surveying phylogenetic footprints in large gene clusters: Applications to *Hox* cluster duplications. *Mol. Phyl. Evol.*, 31:581–604, 2004.

[27] S. J. Prohaska and P. F. Stadler. The duplication of the *Hox* gene clusters in teleost fishes. *Th. Biosci.*, 2004. in press.

[28] M. Sammeth, B. Morgenstern, and J. Stoye. Divide-and-conquer alignment with segment-based constraints. *Bioinformatics, ECCB special issue*, 19:ii189–ii195, 2003.

[29] S. Schwartz, W. Kent, A. Smit, Z. Zhang, R. H. R Baertsch, D. Haussler, and W. Miller. Human-mouse alignments with BLASTZ. *Genome Research*, 13:103–107, 2003.

[30] D. A. Tagle, B. F. Koop, M. Goodman, J. L. Slightom, D. L. Hess, and R. T. Jones. Embryonic epsilon and gamma globin genes of a prosimian primate (galago crassicaudatus). Nucleotide and amino acid sequences, developmental regulation and phylogenetic footprints. *J. Mol. Biol.*, 203:439–455, 1988.

[31] J. D. Thompson, D. G. Higgins, and T. J. Gibson. CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Research*, 22:4673–4680, 1994.

[32] J. D. Thompson, F. Plewniak, and O. Poch. A comprehensive comparison of protein sequence alignment programs. *Nucleic Acids Research*, 27:2682–2690, 1999.

[33] S. Yekta, I.-h. Shih, and D. P. Bartel. MircoRNA-directed cleavage of *HoxB8* mRNA. *Science*, 304:594–596, 2004.