The above optimisation problem makes sense only if the set *Anc* of anchor points is itself consistent. 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 optimisation 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 *prioritise* 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 [18]. Note that the set of gene boundaries will be necessarily consistent as long as the relative ordering among the genes is conserved. However, the automatically created anchor points may well be *inconsistent* with those 'biologically defined' anchors or inconsistent with each other. Since anchor points derived from expert knowledge should be more reliable than anchor points identified by some software program, 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 as scores for the *automatically* created anchor points, while one would assign arbitrarily defined 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 paralogous 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 [26,27]. This group of genes, together with the so-called *ParaHox* genes, arose early in metazoan history from a single ancestral "*UrHox* gene" [28]. 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 organised 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 at the 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 [29-33]. These duplication events were followed by massive loss of the duplicated genes in different lineages, see e.g. [34] 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 [35] that predominantly consists of transcription factor binding sites. Most recently, however, some of these "phylogenetic footprints" were identified as microRNAs [36].

Figure 2 and 3 show 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 (33481 nt to 125385 nt, 4 to 10 genes).

We observe that without anchoring DIALIGN mis-aligns many of of the *Hox* genes in this example 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 *HoxA1Oa* and *HoxDIOa*, 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 Table 2. 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 [17]. For some sequence sets we found that the score of the anchored 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 1). In contrast,