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 it 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 for alignment programs, we implemented a semi-automatic *anchored* alignment approach. Here, the user can specify an arbitrary number of *anchoring points* in order to guide the alignment procedure. Each anchor point consists of a pair of equal-length segments of two of the input sequences. An anchor point is therefore characterised by five coordinates: the two *sequences* involved, the *starting positions* in these sequences and the *length* of the anchored segments. As a sixth parameter, our method requires a *score* that determines the *priority* of an anchor point. The latter parameter is necessary, since it is in general not meaningful to use *all* anchors proposed by the user. It is possible that the selected anchor points are *inconsistent* with each other in the sense that they cannot be included in one single multiple output alignment, see [16] for our concept of consistency. Thus, it may be necessary for the algorithm to select a suitable *subset* of the proposed anchor points.

Our software provides two slightly different options for using anchor points. There is a *strong* anchoring option, where the specified anchor positions are necessarily aligned to each other, consistency provided. The remainder of the sequences is then aligned based on the consistency constraints given by these pre-aligned positions. This option can be used to enforce correct alignment of those parts of the sequences for which additional expert information is available. For example, we are planning to align RNA sequences by using both primary and secondary structure information. Here, locally conserved secondary structures could be used as 'strong' anchor points to make sure that these structures are properly aligned, even if they share no similarity at the primary-structure level.

In addition, we have a *weak* anchoring option, where consistent anchor points are only used to constraint the output alignment, but are not necessarily aligned to each other. More precisely, 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 and y will actually appear in the same column of the output alignment depends on the degree of local 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. The 'weak' option may be useful if anchor points are used to reduce the program running time.

Algorithmically, strong or weak 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* $\leq_{Anc}$ 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 $\leq_{Ali}$ on X as described in [16,25]. 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 $\leq$ on X that is given by the 'natural' ordering of positions within the sequences. In order-theoretical terms, $\leq$ is the *direct sum* of the *linear* order relations defined on the individual sequences. The partial order relation $\leq_{Anc}$ is then defined as the *transitive closure* of the union $\leq \cup Anc$. In other words, we have $x \leq_{Anc} y$ if and only if there is a chain $x_0, \ldots, x_k$ of positions with $x_0 = x$ and $x_k = y$ such that for every $i \in \{1,\ldots, k\}$, position x_{i-1} is either anchored with x_i or x_{i-1} and x_i belong to the same sequence, and x_{i-1} is on the left-hand side of x_i in that sequence.

In our set-theoretical setting, a relation R on X is called consistent if all restrictions of the tansitive closure of the union $\leq \cup R$ to the idividual sequences *coincides* with their respective 'natural' linear orderings. With the *weak* version of our anchored-alignment approach, we are looking for an alignment *Ali* wich maximum score such that the union $Ali \cup Anc$ is consistent. With the *strong* option, we are looking for a maximum-scoring alignment *Ali* that is a superset of *Anc*. With both program options, our optimisation problem is to find an alignment *Ali* with maximum score – under the additional constraint that the set-theoretical union $Ali \cup Anc$ is consistent. In the weak anchoring approach, the output alignment is *Ali* while with the strong option, the program returns the transitive closure of the union $Ali \cup Anc$.