Instructions on using the `IntronAlignment` package

Miklós Csűrös
e-mail:csuros AT iro.umontreal.ca

April 19, 2007

1 Introduction

The Java package `IntronAlignment.jar` contains a number of Java classes for computing a protein multiple alignment where sequences are annotated with the intron positions [Csűrös, Holey, Rogozin "In search of lost introns", *ISMB*, 2007]. The main program is intended to work as a realignment tool for multiple alignment computed by more sophisticated methods (e.g., MUSCLE).

The program takes a multiple alignment as input, and then realigns one sequence at a time to maximize the sum-of-pairs score. You can invoke the program as `java -cp IntronAlignment.jar ca.umontreal.iro.evolution.introns.AAIntronAlign` ⟨switches⟩. Intron positions are specified in the header line for each sequence in the input file. The program looks for a parenthesized list with the syntax `{t` $p_1, p_2, \ldots$ `t}`. Here p_1, p_2, etc. are intron positions with respect to the coding sequences (CDS): $p_i = 1$ is the position after the first position of the first codon. Another character can be specified instead of `t` using the switch `-intron-position-paren`. Instead of interpreting the positions with respect to the (ungapped) CDS, they can be specified relative to the input alignment by using the switch `-intron-position-relative aln`; default behavior is with switch `-intron-position-relative seq`. If positions are relative to the alignment, the sequence headers in the output are amended with sequence-specific intron positions in the format `{I ... I}`.

The recognized switches are the following.

in Specifies the initial alignment, in a multi-Fasta file. Syntax: `-in` ⟨file⟩.

out Specifies the output file name; results are written in Fasta format. Syntax: `-out` ⟨file⟩.

matrix Specifies the amino acid scoring matrix. Syntax: `-matrix` M. If M is one of `vtml240`, `blosum45`, `blosum62` or `blosum80`, then those scores are used (stored internally), otherwise M denotes a file from which scores are read (same format as with BLAST).

gapopen Specifies the score for opening a gap (usually a negative value). Syntax: `-gapopen` d. Half of this value is applied at the beginning of the gap and the other half at the end. Gaps that start at the very beginning of the alignmeent or end at the very end of it are penalized by 50% only.

gapextend Score for extending a gap (usually negative). Syntax: `-gapextend` d.

intronmatch Score of matching two introns of the same phase. Syntax: `-intronmatch` d.

intronmismatch Score of aligning no intron with an intron (aligning a phase-1 and phase-2 intron gets twice this score). syntax: `-intronmismatch` d.

intron-position-paren Specifies a delimiter for listing intron positions. The default delimiter is "t", but you can change it to something else. For instance if the delimiter is "i", then the intron positions are in a comma-separated list enclosed by `{i...i}`. Syntax: `-intron-position-paren` c.

intron-position-relative Specifies the interpretation of the listed intron positiones. Positions may be relative to the sequence, or to the input alignment. Syntax: `-intron-position-relative` X, where X is either `seq` or `aln`.

realign Gives the order of sequences for the realignment. Syntax: `-realign` ⟨list⟩. The ⟨list⟩ is a list of sequence indexes in the input file (0-based indexing). Example: `-realign 0,2,0,3`.

rep Number of realignment cycles (in each cycle, the sequences are realigned in the order specified by `realign`. Syntax: `-rep` n.