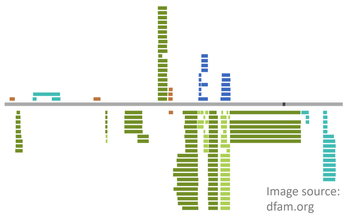
Adjudicating Between Competing Annotations

Kaitlin Carey, Travis Wheeler, Dept. of Computer Science, University of Montana, Missoula, Montana, USA

The Problem: Selecting From Competing Annotations

A common annotation process:

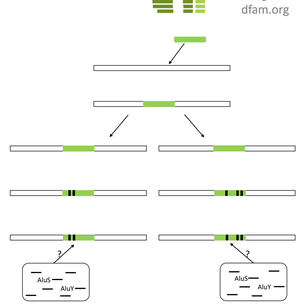
- Compare an un-annotated sequence (the *Target*) to a large collection of known sequences (the *Queries*)
- Sometimes, more than one of the Queries shows a significant match to the Target
- Pick 'true' matching query based on highest alignment score
- Falsely implies certainty of the true match
- When two or more sequences match with high alignment scores, this method is no longer reliable, because it is possible that either one is the true sequence.
- Relevant in databases with highly similar sequences



Pfam Dfam

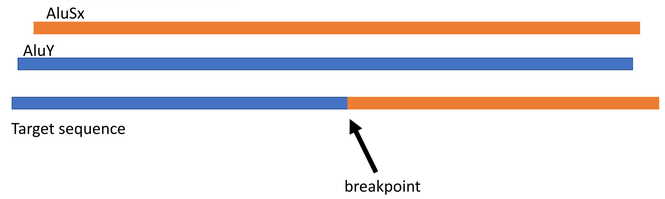
Reliability of Competing Annotations

Annotations of biological replicates should agree. Among transposable elements in humans, more than 10% of annotations disagree.

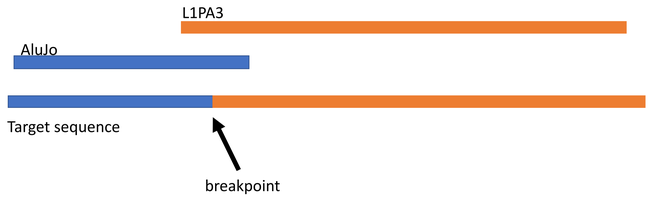


Confidence can also be used to:

1. Identify instances of gene conversion / homologous recombination



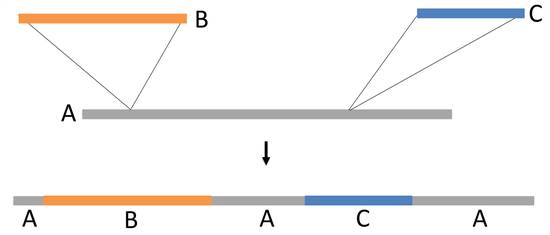
2. Detect boundaries of neighboring partial element matches



3. Identify inserted elements

Sequences can also insert themselves inside of other sequences.

- B and C have both inserted themselves inside of A
- Traditional annotation methods would find 3 occurrences of sequence A
- We want to find the 3 occurrences of A and recognize them as all belonging to the same original sequence



Confidence from Alignment Scores

$Q : q_1, q_2, q_3, \dots, q_n$ (where q can be a sequence or pHMM)

Define $\boldsymbol{P(q_i|t)}$: probability that the true label of t is q_i

Assuming alignments cover all possible explanations for t:

$$\boldsymbol{Conf(q_i|t) = \frac{P(q_i|t)}{\sum_j P(q_j|t)}}$$

We don't have $P(q_i|t)$, but Bayes' rule says:

$$\boldsymbol{P(q_i|t) = P(t|q_i) * \frac{P(q_i)}{P(t)}}$$

Assuming uniform distribution over $P(q_i)$, and with a fixed t, $\boldsymbol{P(q_i|t) \propto P(t|q_i)}$. So,

$$\boldsymbol{Conf(q_i|t) = \frac{P(t|q_i)}{\sum_j P(t|q_j)}}$$

Probabilities ⟷ Scores

Given scoring matrix, alignment scores are (scaled) log odds ratios:

$$\boldsymbol{Score(a,b) = int\left(\lambda * log_2 \frac{P(a,b|homology)}{P(a)P(b)}\right)}$$

λ : scaling factor of scoring matrix

Over entire alignment:

$$\boldsymbol{score(t,q_i) = \lambda * log_2 \frac{P(t|q_i)}{P(t|R)}}$$

[model of homology] / [R=random model]

So:

$$\boldsymbol{P(t|q_i) = P(t|R) * 2^{score(t|q_i)/\lambda}}$$

$$\boldsymbol{Conf(q_i|t) = \frac{P(t|R) * 2^{score(t|q_i)/\lambda}}{\sum_j P(t|R) * 2^{score(t|q_j)/\lambda}}}$$

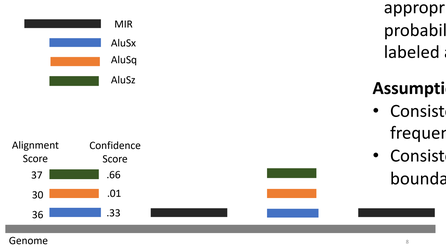
$$\boldsymbol{Conf(q_i|t) = \frac{2^{score(t|q_i)/\lambda}}{\sum_j 2^{score(t|q_j)/\lambda}}}$$

This allows us to:

- "Ambiguate" competing annotations (Shows a more accurate depiction of annotation certainty) or
- Produce a set of appropriate, probabilistically labeled annotations

Assumptions:

- Consistent background frequencies
- Consistent alignment boundaries



Jumping Profile Hidden Markov Model (ideal)

One approach for identifying breakpoints:

- Sub-model HMM for each subfamily in the alignment
- Allows transitions (jumps) between submodels
- Combinatoric explosion of transitions unless the number of possible jumps is constrained

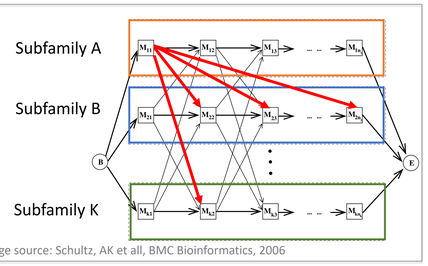
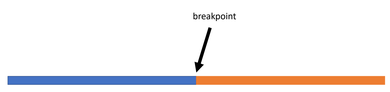


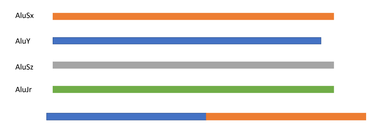
Image source: Schultz, AK et all, BMC Bioinformatics, 2006

Segmented Confidence Approximates Jumping HMM Approach

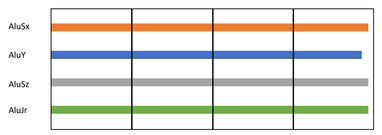
1. Target sequence



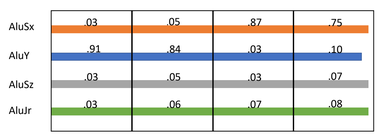
2. Alignment of query sequences



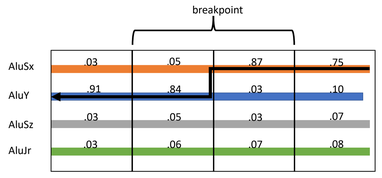
3. Break alignment into segments



4. Confidence for segments



5. Most probable path through the segments



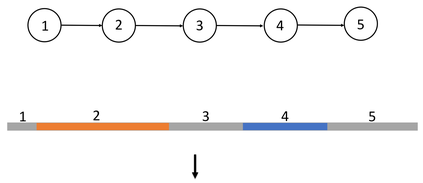
$$S_{i,j} = Conf_{i,j} * \max_{j'} \begin{cases} s_{i-1,j'} * t_m, & \text{If } j \neq j' \\ s_{i-1,j'} * t_s, & \text{If } j = j' \end{cases}$$

Overlapping Confidence Enables Exact Breakpoint Location

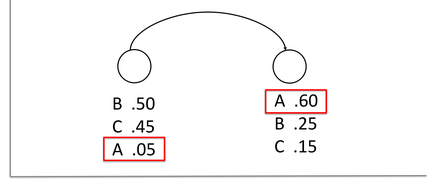
Using overlapping segments, starting at every nucleotide, we compute nucleotide specific support scores by averaging all the confidence values for each segment that overlaps the nucleotide position. Per position scores in the dynamic programming matrix, enables us to find the exact breakpoint location.

Graph Algorithm Identifies Inserted Elements

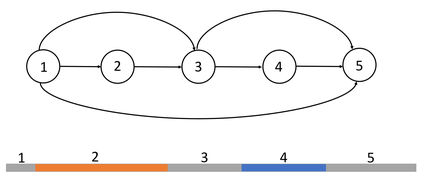
1. Make graph of all sequences labeled in dynamic programming. Each labeled segment becomes a node



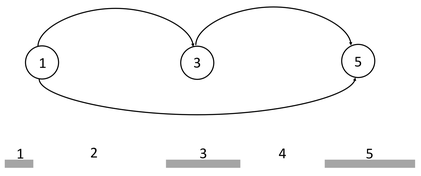
Edge Creation Rule:
Create edge if the destination's best label has a confidence > 0.01 in the source



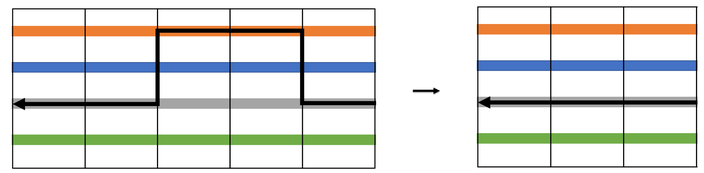
2. Find alternative paths through the graph based on confidence



3. Splice out all sequences with no alternate paths



4. Splice corresponding positions out of dynamic programming matrix, stitching the original sequence back together



Example: chr11:11,989,996-11,992,119

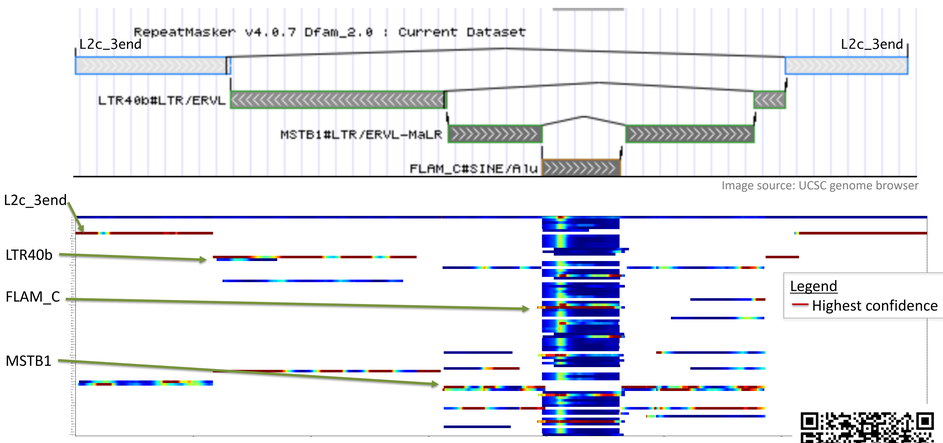


Image source: UCSC genome browser

Heatmap of support scores used in dynamic programming matrix - demonstrates how the inserted elements are identified and the original sequences are stitched back together.

Works Cited: [1] Bailey, JA, et all, Science, 2002 [2] hmmer.org v3.1 [3] Schultz, AK et all, BMC Bioinformatics, 2006 [4] Smit, A, Hubley, R, RepeatMasker, 2013 [5] Yu, YK et all, PNAS, 2003

Research funded by: NIH P20GM103546, 5U24HG010136