Multiple Whole Genome Alignment

BMI/CS 776
www.biostat.wisc.edu/bmi776/
Spring 2012
Colin Dewey
cdewey@biostat.wisc.edu
Goals for Lecture

the key concepts to understand are the following
• the large-scale multiple-alignment task
• progressive alignment
• breakpoint identification
• undirected graphical models
• minimal spanning trees/forests
Multiple Whole Genome Alignment: Task Definition

Given
– a set of $n > 2$ genomes (or other large-scale sequences)
– a method for scoring the similarity of a pair of characters

Do
– construct global alignment: identify matches between genomes as well as various non-match features
The MLAGAN Method
[Brudno et al., Genome Research, 2003]

Given: $k$ genomes $X^1, \ldots, X^k$, guide tree $T$
for each pair of genomes $X^i, X^j$

$\text{anchors}(i, j) = \text{find_anchors}(X^i, X^j)$

$\text{align} = \text{progressive_alignment}(T, \text{anchors})$
for each genome $X^i$

// iterative refinement
$\text{anchors} = \text{segments of } X^i \text{ with high scores in } \text{align}$
$\text{align} = \text{LAGAN}(\text{align} - X^i, X^i, \text{anchors})$  // realign $X^i$

\text{progressive_alignment}(T, \text{anchors})
if $T$ is not a leaf node

$\text{align_left} = \text{progressive_alignment}(T.\text{left}, \text{anchors})$
$\text{align_right} = \text{progressive_alignment}(T.\text{right}, \text{anchors})$
$\text{align} = \text{LAGAN}(\text{align_left}, \text{align_right}, \text{anchors})$
return $\text{align}$
Progressive Alignment

- given a *guide tree* relating *n* genomes
- construct multiple alignment by performing *n*-1 pairwise alignments
Progressive Alignment: MLAGAN Example

align pairs of sequences

align multi-sequences (alignments)

align multi-sequence with sequence
Progressive Alignment: MLAGAN Example

suppose we’re aligning the multi-sequence X/Y with Z

1. anchors from X-Z and Y-Z become anchors for X/Y-Z
2. overlapping anchors are reweighted
3. LIS algorithm is used to chain anchors

Figure from: Brudno et al. *Genome Research*, 2003
Reweighting Anchors in MLAGAN

\[
(s_1 + s_2) \times \frac{I}{U}
\]
Genome Rearrangements

- can occur within a chromosome or across chromosomes
- can have combinations of these events
Genome Rearrangement Example: Mouse vs. Human X Chromosome

- each colored block represents a syntenic region of the two chromosomes
- the two panels show the two most parsimonious sets of rearrangements to map one chromosome to the other

Figure from: Pevzner and Tesler. *PNAS*, 2003
The Mauve Method

[Darling et al., Genome Research, 2004]

Given: $k$ genomes $X^1, \ldots, X^k$

1. find multi-MUMs (MUMs present in 2 or more genomes)
2. calculate a guide tree based on multi-MUMs
3. find LCBs (sequences of multi-MUMs) to use as anchors
4. do recursive anchoring within and outside of LCBs
5. calculate a progressive alignment of each LCB using guide tree

* note: no LIS step!
2. Calculating the Guide Tree in Mauve

- unlike MLAGAN, Mauve calculates the guide tree instead of taking it as an input

1. find multi-MUMs in sequences
2. calculate pairwise distances
3. run neighbor-joining to get guide tree

- distance between two sequences is based on fraction of sequences shared in multi-MUMs
3. Selecting Anchors: Finding Local Collinear Blocks

repeat
  • partition set of multi-MUMs, $M$ into collinear blocks
  • find minimum-weight collinear block(s)
  • remove minimum weight block(s) if they’re sufficiently small until minimum-weight block is not small enough
4. and 5. Recursive Anchoring and Gapped Alignment

- recursive anchoring (finding finer multi-MUMs and LCBs) and standard alignment (CLUSTALW) are used to extend LCBs
Mauve Alignment of 9 Enterobacteria

(Shigella and E. coli)
Mauve vs. MLAGAN: Accuracy on Simulated Genome Data

substitution and indel rates observed in enterobacteria
Mauve vs. LAGAN: Accuracy on Simulated Genome Data with Inversions

Figure courtesy of Aaron Darling
Evolution with *Horizontal Transfer*
Mauve Accuracy on Simulated Enterobacteria-like Data

- data here include horizontal transfers
- small HT events have little effect compared to large HT events
- when scored on regions conserved in all 9 taxa, accuracy is always > 98%

Figures courtesy of Aaron Darling
Mercator

- orthologous segment identification: graph-based method
- breakpoint identification: refine segment endpoints with a graphical model
Establishing Anchors Representing Orthologous Segments

- anchors can correspond to genes, exons or MUMS
- e.g., may do all-vs-all pairwise comparison of genes
- construct graph with anchors as vertices and high-similarity hits as edges (weighted by alignment score)
Rough Orthology Map

k-partite graph with edge weights

vertices = anchors, edges = sequence similarity
Greedy Segment Identification

- for $i = k$ to 2 do
  - identify repetitive anchors (depends on number of high-scoring edges incident to each anchor)
  - find “best-hit” anchor cliques of size $\geq i$
  - join colinear cliques into segments
  - filter edges not consistent with significant segments
Mercator Example

- Repetitive elements (black anchors) are identified; 3-cliques (red and blue anchors) are found.
- Segments are formed by red and blue anchors; inconsistent edges are filtered.
- 2-cliques are found and incorporated into segments.
Refining the Map: Finding Breakpoints

- **breakpoints**: the positions at which genomic rearrangements disrupt colinearity of segments

Mercator finds breakpoints by using inference in an undirected graphical model.
**Undirected Graphical Models**

- an undirected graphical model represents a probability distribution over a set of variables using a factored representation

\[
p(b) = \frac{1}{Z} \prod_{C \in \text{cliques}} \psi_C(b_C)
\]

- \(B_i\) random variable
- \(b\) assignment of values to all variables
- \(b_C\) assignment of values subset of variables in \(C\)
- \(\psi_C\) function (called a potential) representing the “compatibility” of a given set of values
- \(Z\) normalization term
Undirected Graphical Models

\[ p(b) = \frac{1}{Z} \prod_{C \in \text{cliques}} \psi_C(b_C) \]

for the given graph:

\[ p(b) = \frac{1}{Z} \psi_1(b_1, b_3, b_5) \psi_2(b_1, b_6, b_7) \psi_3(b_2, b_4, b_6) \]
The Breakpoint Graph

some prefix of region 2 and some prefix of region 11 should be aligned
Breakpoint Undirected Graphical Model

- Mercator frames the task of finding breakpoints as an inference task in an undirected graphical model.

\[
p(b) = \frac{1}{Z} \prod_{C \in \text{cliques}} \psi_C(b_C)
\]

- Configuration of breakpoints:

- Potential function representing score of multiple alignment of sequences in clique \( C \) for breakpoints in \( b \).
Breakpoint Undirected Graphical Model

- the possible values for a variable indicate the possible coordinates for a breakpoint
- the potential for a clique is a function of the alignment score for the breakpoint regions split at the breakpoints $b_C$
Breakpoint Undirected Graphical Model

\[ p(b) = \frac{1}{Z} \prod_{c \in \text{cliques}} \psi_c(b_c) \]

- \textit{inference task}: find most probable configuration \( b \) of breakpoints
- not tractable in this case
  - graph has a high degree of connectivity
  - multiple alignment is difficult
- so Mercator uses several heuristics
Making Inference Tractable in Breakpoint Undirected Graphical Model

\[ p(\mathbf{b}) = \frac{1}{Z} \prod_{C \in \text{cliques}} \psi_C(\mathbf{b}_C) \]

- assign potentials, based on pairwise alignments, to edges only

\[ p(\mathbf{b}) = \frac{1}{Z} \prod_{(i,j) \in \text{edges}} \psi_{i,j}(b_i, b_j) \]

- eliminate edges by finding a minimum spanning forest, where edges are weighted by phylogenetic distance
Minimal Spanning Forest

• *minimal spanning tree*: a minimal-weight tree that connects all vertices in a graph

• *minimal spanning forest*: a set of MSTs, one for each connected component
Breakpoint Finding Algorithm

1. construct breakpoint segment graph
2. weight edges with phylogenetic distances
3. find minimum spanning tree/forest
4. perform pairwise alignment for each edge in MST
5. use alignments to estimate $\psi_{i,j}(b_i, b_j)$
6. perform max-product inference (similar to Viterbi) to find maximizing $b_i$
Comments on Whole-Genome Alignment Methods

• employ common strategy
  – find seed matches
  – identify (sequences of) matches to anchor alignment
  – fill in the rest with standard methods (e.g. DP)

• vary in what they (implicitly) assume about
  – the distance of sequences being compared
  – the prevalence or rearrangements

• involve a lot of heuristics
  – for efficiency
  – because we don’t know enough to specify a precise objective function (e.g. how should costs should be assigned to various rearrangements)