**Next Generation Bioinformatics Tools**

Fall 2012
Day 2 – Sequence Analysis and Phylogeny

Hesham H. Ali
UNO Bioinformatics Research Group
College of Information Science and Technology

---

**Sequence Comparison**

- Biology has a long tradition of comparative analysis leading to discovery.
- The number of sequences available for comparison has been growing explosively.
- Efficient algorithms already exist for solving many sequence comparison related problems.
- Sequences may be incomplete and/or having different sizes

---

**...Its Not Straightforward**

- Compare the two DNA sequences: GACGATTAG and GATCGAATAG

<table>
<thead>
<tr>
<th>GACGATTAG</th>
<th>GATCGAATAG</th>
</tr>
</thead>
<tbody>
<tr>
<td>GA</td>
<td>GA</td>
</tr>
<tr>
<td>CG</td>
<td>TC</td>
</tr>
<tr>
<td>ATTAG</td>
<td>AATAG</td>
</tr>
</tbody>
</table>

- 20% similar... but add room for gaps...

<table>
<thead>
<tr>
<th>GA- CGATTAG</th>
<th>GATCGAATAG</th>
</tr>
</thead>
<tbody>
<tr>
<td>GA</td>
<td>GA</td>
</tr>
<tr>
<td>CG</td>
<td>TG</td>
</tr>
<tr>
<td>ATTAG</td>
<td>AATAG</td>
</tr>
</tbody>
</table>

- 80% similar
Why Compare Sequences?

- Identification of organisms using obtained sequences
- Classification of organisms - evolutionary analysis
- Function prediction
- Structure prediction
- Motif Finding
- Such a tool would have to utilize biological knowledge and databases to identify sequences
- Clustering and Phlogenetic Analysis

Sequence Comparison Solutions

- Sequence alignment
- Sequence comparison using data compression
- Sequence comparison using landmarks/motifs
- Sequence comparison using Longest Common Subsequence
- Sequence Comparison using composition based methods and various statistical properties

Each method works well for different types of Bioinformatics problems

String Matching

- Exact Matching: What is the problem?
- Importance of the exact matching problem
- The Naïve string matching method
- String matching with finite automaton
- Classical Comparison-Based Methods
- Boyer Moore Algorithm
- Knuth-Morris-Pratt Algorithm (KMP)
- Applications of exact matching
Exact Matching: What’s the problem?

• Given a string P called a pattern and a longer string T called the text, the exact matching problem is to find all occurrences, if any, of pattern P in text T.
  – If P = "aba" and T = "bbabaxababay" then P occurs in T starting at locations 3, 7, and 9. Note that two occurrences of P may overlap (as at locations 7 and 9).

Importance of Exact Matching

• Obvious applications such as Word-processing in utilities such as grep on UNIX.
• In textual information retrieval systems such as Medline, Lexis, etc.
• Library catalog searching programs.
• Directories, Dictionary, etc.
• In Molecular Biology there are several hundred specialized databases holding raw DNA, RNA and Amino acid strings, or processed patterns (called Motifs) derived from raw string data.

Classical Comparison Based Methods

• Naïve string matching
• Finite Automaton Algorithms
• KMP Algorithm
• Boyer Moore Algorithm
The Naïve Method

• Finds all valid shifts using a loop that checks the condition
  
P[1...m] = T[s+1...s+m]
  
  for each of the \( (n - m + 1) \) possible values of \( s \).

The Naïve Method

• Can be interpreted graphically as sliding a "template" containing the pattern over the text.
• The For loop considers each possible shift explicitly.
• The Test condition determines whether the current shift is valid or not.
• The worst-case running time is \( O((n - m + 1)m) \)

String matching with Finite State Automata

• Many string matching algorithms build a finite automaton that scans the text string \( T \) for all occurrences of the pattern \( P \).
• These string-matching automata are very efficient
  
  – They examine each text character exactly once, taking constant time per text character.
  – The time taken after the automaton is built – \( O(n) \).
String matching with Finite Automata

- Finite automaton: A finite automaton $M$ is a 5-tuple $(Q, q_0, A, S, d)$, where
  - $Q$ is a finite set of states.
  - $q_0$ belongs to set $Q$ is the start state.
  - $A$ subset of $Q$ is a distinguished set of accepting states or final states.
  - $S$ is a finite input alphabet.
  - $d$ is a function from $Q \times S$ into $Q$, called the transition function of $M$.

Boyer-Moore Algorithm

- If the pattern $P$ is relatively long and the alphabet $\Sigma$ is reasonably large, then the Boyer-Moore algorithm is likely to be the most efficient string matching algorithm.
- Contains 3 clever ideas
  - The right-to-left scan.
  - The bad character shift rule
  - And the good suffix shift rule

Knuth-Morris-Pratt Algorithm (KMP)

- It is a linear time string matching algorithm.
- Achieves a $O(n+m)$ running time by avoiding the computation of the transition function $d$ altogether.
- It does the pattern matching using an auxiliary function $\Phi[1..m]$, which is pre-computed from the pattern in time $O(m)$. 
But we are more interested in Non-Exact matching

- Sequencing errors
- Repeats, translocation, insertions/deletions (indels)
- Evolutionary aspects of biological sequences
- Evolutionary aspects of systems and organisms

Sequence Alignment

- Goal: To enable researchers to determine whether two sequences display sufficient similarity to justify the inference of homology.
- Definition: Given two sequences of sizes m and n, an alignment is the insertion of spaces in arbitrary locations along the sequences so that they end up with the same size. Possible restriction: No space in one sequence is aligned with a space in the other.
Sequence Alignment

• Types:
  – Pairwise vs Multiple
    • Pairwise: Comparing 2 sequences at a time
    • Multiple: Comparing 2+ sequences (pairwise can be multiple)
  – Local vs. Global
    • Local: Comparing short sequences/regions of sequence (e.g. CDS regions)
    • Global: Comparing entire sequences/regions (e.g. genomes, chromosomes)

Global Alignment

• Alignment is defined as the insertion of spaces in arbitrary locations along the sequences so that they end up with the same size.
• No space in one sequence be aligned with a space in the other.

Example:

• Compare the two DNA sequences:
  GACGATTAG and GATCGAATAG

GACGATTAG
GATCGAATAG
• 20% similar…but add room for gaps …

GA-CGATTAG
GATCGAATAG
• 80% similar
Score of Alignment

- **GA-CGATTAG**
- **GATCGAATTAG**
- 80% similarity
- Scoring:
  - Match = +1
  - Mismatch = -1
  - Space = -2
- \[
\begin{align*}
&\text{GA - CGATTAG} \\
&\text{GATCGAATTAG}
\end{align*}
\]

\[
\begin{align*}
&1 & 1 & -2 & 1 & 1 & 1 & -1 & 1 & 1 & 1 \\
\text{Sum} &= 5
\end{align*}
\]

The Main Equation

\[
\begin{align*}
\text{Sim}(S_1[1...i], S_2[1...j]) &= \text{max} \{ \\
&\text{Sim}(S_1[1...i-1], S_2[1...j-1]) + \text{p}(i, j) \\
&\text{Sim}(S_1[1...i], S_2[1...j-1]) \delta(-, j) \\
&\text{Sim}(S_1[1...i-1], S_2[1...j]) \delta(i, -) \}
\end{align*}
\]

Where \( p(i, j) \) = either +1 (match) or -1 (mismatch) and \( \delta(-, j) = \delta(i, -) = -2 \)

Global Alignment

- Consider:
  - **GA-CGATTAG**
  - **GATCGAATTAG**
- Scoring an alignment
- Total score = (8*1) + (1*1) + (1*2) = 5
- Observations:
  - For \((i,j)\) we need only \((i-1,j),(i-1,j-1),(i-1,j-1))\)

\[
\begin{align*}
&A[i,j] = \text{max} \{ a[i-1, j-1] + p(i, j), \\
&a[i, j-1] - d(-, j), \\
&a[i-1, j] - d(i, -) \}
\end{align*}
\]
Complexity (Cost) of the Alignment Problem

- Generating all possible alignments (exponential)
- Smart algorithm using dynamic programming (efficient optimal algorithm)
  Modification of the DP solution to the longest common subsequence problem

Optimal Alignment: DP Approach

- Key observation: the optimal alignment for the two sequences S1[1…m] and S2[1…n] can be evaluated iteratively by finding the optimal alignment for S1[1…i] and S2[1…j] for all i<m and j<n.

- To find the alignment for S1[1…i] and S2[1…j], we have three choices only:
  - Align S1[1…i-1] with S2[1…j-1] and match S1[i] with S2[j], or
  - Align S1[1…i] with S2[1…j-1] and match a space with S2[j], or
  - Align S1[1…i-1] with S2[1…j] and match S1[i] with a space.

Global Comparison: The Basic Algorithm (Needleman-Wunsch)

Algorithm Similarity
Input: sequences S1 and S2
Output: similarity Score between S1 and S2
m = |S1|
n = |S2|
for i=0 to m do d[i,0] = i-2
for j=0 to n do d[0,j] = j-2
for i=1 to m do
  for j=1 to n do
    d[i,j] = max(d[i-1,j-1]+p(i,j), d[i-1,j]-2, d[i,j-1]-2)
return d[m,n]
Gapped Alignments

- Biological sequences
  - Different lengths
  - Regions of insertions and deletions
- Notion of gaps (denoted by ‘-’)

Possible Residue Alignments

- Match
- Mismatch (substitution or mutation)
- Insertion/Deletion (INDELS – gaps)

Edit Graph
Different Alignments

- Which alignment is best?

\[
\begin{align*}
A & - C & - G & - G & - A & - C & T \\
| & | & | & | & | & | & | \\
A & T & C & G & G & A & T & _ & C & T
\end{align*}
\]

\[
\begin{align*}
A & - C & G & G & - A & C & T \\
| & | & | & | & | & | & | \\
A & - C & G & G & - A & C & T
\end{align*}
\]

Alignment Scoring Scheme

- Possible scoring scheme:
  - match: +2
  - mismatch: -1
  - indel: -2

- Alignment 1: 5 * 2 - 1(1) - 4(2) = 10 - 1 - 8 = 1
- Alignment 2: 6 * 2 - 1(1) - 2(2) = 12 - 1 - 4 = 7

Dynamic Programming

- Solve optimization problems by dividing the problem into subproblems
- Overlapping subproblems
- Optimal substructure
- Useful tabulation
- Sequence alignment has optimal substructure property
- Subproblem: alignment of prefixes of two sequences
- Each subproblem is computed once and stored in a matrix
DP Example

Sequence #1: GAATTCAGTTA; M = 11
Sequence #2: GGATCGA; N = 7

- \( s(a, b) = +5 \) if \( a = b \) (match score)
- \( s(a, b) = -3 \) if \( a \neq b \) (mismatch score)
- \( w = -4 \) (gap penalty)

View of the DP Matrix

- \( M+1 \) rows, \( N+1 \) columns

Global Alignment (Needleman-Wunsch)

- Attempts to align all residues of two sequences
- \( \text{INITIALIZATION: First row and first column set} \)
- \( S_{i,0} = w \times i \)
- \( S_{0,j} = w \times j \)
Initially, the matrix is initialized with:

\[
\begin{array}{cccccccc}
  & G & A & A & T & T & C & A & G & T & T & A \\
- & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\
G & 1 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\
A & 2 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 \\
A & 3 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \\
T & 4 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 \\
T & 5 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 \\
T & 6 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \\
T & 7 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 \\
T & 8 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 \\
T & 9 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 \\
T & 10 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 & 19 \\
T & 11 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 & 19 & 20 \\
T & 12 & 12 & 13 & 14 & 15 & 16 & 17 & 18 & 19 & 20 & 21 \\
\end{array}
\]

The matrix fill for global alignment is defined as:

\[
S_{i,j} = \text{MAXIMUM}\{ \\
S_{i-1,j-1} + s(a_i, b_j) \text{ (match/mismatch in the diagonal)}, \\
S_{i-1,j} + w(\text{gap in sequence } w_1), \\
S_{i,j-1} + w(\text{gap in sequence } w_2) \\
\}
\]

For the initialization:

\[
S_{1,1} = \text{MAXIMUM}\{ \\
S_{0,0} + 5, S_{1,0} - 4, S_{0,1} - 4 \}
\]

\[
S_{1,1} = \text{MAXIMUM}\{ \\
5, -8, -8 \}
\]

\[
S_{1,1} = 5
\]
Matrix Fill (Global Alignment)

\[ S_{1,1} = \text{MAX}[S_{0,1} - 3, S_{1,1} - 4, S_{2,1} - 4] = \text{MAX}[-4 - 3, 5 - 4, 8 - 4] = \text{MAX}[-7, 1, -12] \]

[Diagram of matrix fill process]

Filled Matrix (Global Alignment)

[Diagram of filled matrix]

9/10/12
Trace Back (Global Alignment)

- Maximum global alignment score = 11 (value in the lower right hand cell).
- Traceback begins in position $S_{M,N}$, i.e. the position where both sequences are globally aligned.
- At each cell, we look to see where we move next according to the pointers.
Checking Alignment Score

<table>
<thead>
<tr>
<th>G A A T C A G T T A</th>
<th>G G A - T C - G - - A</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ - + - + - + - + +</td>
<td></td>
</tr>
<tr>
<td>5 3 5 4 5 5 4 5 4 5</td>
<td></td>
</tr>
</tbody>
</table>

5 - 3 + 5 - 4 + 5 - 4 + 4 + 5 = 11

Variations of the Alignment Problem

- **Local Comparison:** Alignment between sub-strings of two sequences.
- **Semi-global comparison:** Ignoring some of the end spaces in the sequences
  
  **CAGC**-CTGGATTCTCGG
  
  **CAGC**-CTGGATTCTCGG
  
  -19 or 3
  
  **CAGC**-CTGGATTCTCGG
  
  -12

- Comparing multiple sequences:
  
  MQPILLL
  
  MRL-LL
  
  MR-ILL
  
  MPPVLIL

Local Alignment

- Given two Strings S1 and S2 find sub-strings A and B of S1 and S2, respectively. Whose similarity (optimal global alignment value) is maximum over all pairs of sub-strings from S1 and S2.
- Why? – Global alignment of protein is often meaningful when the 2 strings are members of the same protein family.
- Local alignment is critical because proteins from very different families are often made up of the same structural and functional subunits.
Local Alignment Algorithm (Smith-Waterman)

Algorithm Similarity
Input: sequences S1 and S2
Output: similarity Score between S1 and S2
m = |S1|
n = |S2|
for i=0 to m do d[i,0] = 7
for j=0 to n do d[0,j] = 7
for i=1 to m do
  for j=1 to n do
    d[i,j] = max(d[i-1,j-1]+p(i,j), d[i-1,j]-2, d[i,j-1]-2, 7)
return d[m,n]

Local Alignment

• Smith-Waterman: obtain highest scoring local match between two sequences
• Requires 2 modifications:
  – Initial scores are zeros
  – When a value in the score matrix becomes negative, reset it to zero (begin of new alignment)
Make sure mismatches have negative scores

Local Alignment Initialization

Values in row 0 and column 0 set to 0.
Matrix Fill (Local Alignment)

\[ S_{i,j} = \text{MAXIMUM} \{
\begin{align*}
S_{i-1,j-1} + s(a_i, b_j) & \quad \text{(match/mismatch in the diagonal)} \\
S_{i,j-1} + w & \quad \text{(gap in sequence #1)} \\
S_{i-1,j} + w & \quad \text{(gap in sequence #2)} \\
0 &
\end{align*}
\} \]

Matrix Fill (Local Alignment)

\[ S_{1,1} = \text{MAX} \{ S_{0,0} + 5, S_{1,0} - 4, S_{0,1} - 4, 0 \} = \text{MAX} \{ 5, -4, -4, 0 \} = 5 \]

\[ S_{1,2} = \text{MAX} \{ S_{0,1} - 3, S_{1,1} - 4, S_{0,2} - 4, 0 \} = \text{MAX} \{ 0 - 3, 5 - 4, 0 - 4, 0 \} = \text{MAX} \{ -3, 1, -4, 0 \} = 1 \]
Matrix Fill (Local Alignment)

\[ S_{1,3} = \max \{ S_{0,2}, S_{1,2} - 4, S_{0,3} - 4, 0 \} = \max \{ -3, -4, -4, 0 \} = 0 \]

Filled Matrix (Local Alignment)

Maximum Local Alignment

\[
\begin{align*}
\text{G A A T T C - A} & \quad \text{G A A T T C - A} \\
\text{G G A T - C G A} & \quad \text{G G A - T C G A} \\
+ & \quad + \\
5 3 5 5 4 5 4 5 & \quad 5 3 5 4 5 5 4 5
\end{align*}
\]
Multiple Sequence Alignment

- Similar regions conserved across organisms
  - Same or similar function
  - Same or similar structure
- Simultaneous alignment of similar regions yields:
  - regions subject to mutation
  - regions of conservation
  - mutations or rearrangements causing change in conformation or function

Multiple Sequence Alignment

- New sequence can be aligned with known sequences
  - Yields insight into structure and function
- Multiple alignment can detect important features or motifs
- GOAL: Take 3 or more sequences, align so greatest number of characters are in the same column
- Difficulty: introduction of multiple sequences increases combination of matches, mismatches, gaps

Example Multiple Alignment

- Example alignment of 8 Ig sequences.
Approaches to Multiple Alignment

- Dynamic Programming
- Progressive Alignment
- Iterative Alignment
- Statistical Modeling

CLUSTALW

- Perform pairwise alignments of all sequences
- Use alignment scores to produce phylogenetic tree
- Align sequences sequentially, guided by the tree
  - Enhanced Dynamic Programming used to align sequences
  - Genetic distance determined by number of mismatches divided by number of matches
  - Gaps are added to an existing profile in progressive methods
  - CLUSTALW incorporates a statistical model in order to place gaps where they are most likely to occur

BLAST

- Basic Local Alignment Search Tool
- Identifies similar sequences based on a query
BLAST

- Basic Local Alignment Search Tool
  - Altschul et. al., 1990 J. Mol. Biology
  - One of the most widely used tool
    - 41812 citations until August 2012
  - Much faster than Smith-Waterman algorithm because it uses heuristics
    - may not guarantee optimal alignments as Smith-Waterman does

For fast results,
- Minimizes the search space
- May result in loss in sensitivity
- 3 heuristic layers are used
  - Seeding
  - Extension
  - Evaluation

<table>
<thead>
<tr>
<th>Program</th>
<th>Database</th>
<th>Query</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLASTN</td>
<td>Nucleotide</td>
<td>Nucleotide</td>
</tr>
<tr>
<td>BLASTP</td>
<td>Protein</td>
<td>Protein</td>
</tr>
<tr>
<td>BLASTX</td>
<td>Protein</td>
<td>Nucleotide translated into protein</td>
</tr>
<tr>
<td>TBLASTN</td>
<td>Nucleotide translated into protein</td>
<td>Protein</td>
</tr>
<tr>
<td>TBLASTY</td>
<td>Nucleotide translated into protein</td>
<td>Nucleotide translated into protein</td>
</tr>
</tbody>
</table>

Types of BLAST algorithms:
- Megablast:
  - Contiguous: Nearly identical sequences
  - Discontiguous: Cross-species comparison

- Position specific:
  - PSI-BLAST: Automatically generates a position specific score matrix (PSSM)
  - RPS-BLAST: Searches a database of PSI-BLAST PSSMs
Specialized BLAST options

BLAST Algorithm

1. For the query find the list of high scoring words of length w.

BLAST Algorithm

2. Compare the word list to the database and identify exact matches.
BLAST Algorithm

(3) For each word match, extend alignment in both directions to find alignments that score greater than score threshold S.

Word Segment Path (WSP)

BLAST

• Seeding
  – BLAST assumes significant alignments have common words
  – BLAST first finds common words also called word hits
  – In NCBI BLAST, blastn has minimum word size = 7

BLAST

• Extension
  – The word hits are seeds and alignments can be generated from the individual seeds
  – From seeds, the alignment is extended on either sides until the score drops to a threshold value
BLAST

• Evaluation
  – After the seeds are extended to create alignments
  – The alignments are evaluated to determine their statistical significance
    • Significant alignments are called HSPs (High Scoring Pairs)
    • An alignment threshold effectively removes random, low scoring alignments
    • Normalized score for all HSPs is calculated
    • E-Value
      – How many alignment with a given score are expected by chance

BLAST

• A = Low alignment thresholds
  – Shows all low scoring alignments
• B = Medium alignment thresholds
• C = High alignment thresholds

BLAST output report

• 4 major parts
  – Header
  – One line summaries
  – Alignments
  – Footer
BLAST output 1 (Header)

• It contains name of the program (blastn, blastp, blastx, tblastn, tblastx)
• Reference to scientific literature
• Query Sequence
• Database Information

BLAST output 2 (One-line summary)

• Each line has:
  – name of the sequence
  – highest scoring alignment and
  – E-value for the HSP
  – Usually limited to 500 hits

BLAST output 3 (Alignments)

• Major Bulk of the report
• Shows the match, gap, mismatch information for each HSP
  – Score, E-value also reported
BLAST output 4 (Footer)

- It reports various parameters
  - Scoring scheme (Matrix)
  - Neighborhood word threshold score (T)
  - Word size (A for Amino acid)

BLAST Example

- BLAST Example

- BLAST Example
Further Reading

- Korf, et al., BLAST, O'Reilly Publication, 2003

Non-Alignment Approaches

One Method is not Enough

- Would different objectives of sequence comparison demand different comparison approaches
- Recently, alignment free methods have been approached:
  - Data Compression based approaches
  - Motifs based approaches
  - Composition based approaches and other Statistics based approaches
Problems with Alignment

- Optimal alignment is expensive, $O(n^2)$
- Alignment based approach use a very fine grain perspective which may not be suitable for all applications
- Fails to compare long sequences, easy to fool by repetitions and translocations
- It is independent of the input domain
- Inaccurate with incomplete genomes
  
  Alignment-free methods?

Problems with Multiple Sequence Alignment?

- Multiple sequence alignment of the whole genomes is not feasible
  - Genomic rearrangements, insertion and deletions
  - High resource requirements
  - Long running time
- To compare genomes, specific genes are selected manually and compared across organisms
  - But the process could be manual

Next Generation Tools

- Dynamic: Custom built and domain dependent
- Collaborative: Incorporate biological knowledge and expertise
- Intelligent: based on a learning model that gets better with additional data/information

Intelligent Collaborative Dynamic (ICD) Tools
Proposed Vision
(Integrated Advanced Identification System)

- GenBank Sequence Alignment
- Proposed Method
- Other Methods (e.g., suffix/sub-sequence)

Feature Frequency Profile\(^1\)

- The frequency of all possible words (l-mers)
- A sliding window of length ‘l’ is run
  - All ‘l-mers’ are counted and tabulated in a vector
    - Feature vector \( C_l = < c_{1,1} \ldots c_{1,k} > \) where \( k \) = number of possible features
    - For DNA, alphabet size is 4
      - number of possible features of length \( l = 4^l \)

\(^1\)Sims, G.E. et al., Alignment-free genome comparison with frequency feature profiles (FFP) and optimal resolutions, PNAS February 2009.
**FFP contd..**

- Feature Frequency Profile is the Probability distribution vector of feature vector
  \[ FFP_{genome1} = \frac{C_i}{\sum C_i} \]
- FFPs from two genomes are compared using Jensen-Shannon Divergence
  - this metric is used as distance between the genomes

**Exercise**

- Later in the exercise, we have 6 different bacterial genomes to compare
  - 2 Streptococcus genomes
  - 3 Staphylococcus genomes
  - 1 Lactobacillus genome
- We use FFP to find distance between each of these genomes and draw a phylogeny tree

**Compression Based Techniques**

- Each sequence is scanned and linearly independent strings are obtained and form a dictionary
- Differences among dictionaries reflects dissimilarity among input sequences
- Repetitions and translocation don’t impact the dictionaries as compared to alignment
- For any two sequences \( x \) and \( y \), we need
  - \( C(x), C(y), C(xy) \) and \( C(yx) \).
Example

S = AAGTTACCATTG  R = CTagggacttat
Q = AGGTCACCAA

H₁(S) = A/AC/G/T/ACC/AT/TG
H₁(R) = C/T/A/G/GGA/CTT/AT
H₁(Q) = A/G/G/CT/CA/CC/AA

• H₁(SQ) = A/AC/G/T/ACC/AT/TG/ACGG/TC/ACCAA
• H₁(RQ) = C/T/A/G/GGA/CTT/AT/ACG/GT/CA/CC/AA

c(SQ) - c(S) = 3  c(RQ) - c(R) = 5

Q is “closer” to S than R
Distance (S, Q) = c(SQ) - c(S) + c(QS) - c(Q)

Compression-Based Methods

• The choices
  – Kolomogrov complexity (3 distance measures)
  – Lempel-Ziv complexity (4 distance measures)
  – Clustering
    • UPGMA
    • NJ
  – Gold standard tree
  – Path-length difference

How to evaluate compression Methods?

• Find the distances between each pair of sequences
• Build the evolutionary tree (compression-based tree)
• Build the evolutionary tree of sequence alignment (alignment-based tree)
• Measure the distance between the two trees to the reference tree
• The closer tree to the reference tree; is a result for an algorithm with higher accuracy
Evaluation of Comparison Approaches

- **Gold standard tree**
- **MSA tree**
- **Algorithmic tree**

The distance between trees

- Problems with visual inspection?
- Computational ways?
  - Accurate
  - Fast

---

Evaluation, Evaluation

---

Path-Difference Length

---
Distance Between Trees

\[
\begin{align*}
& (AB - A'B')^2 + (AC - A'C')^2 + (AD - A'D')^2 + (BC - B'C')^2 + (BD - B'D')^2 + (CD - C'D')^2 \\
& = (2 - 3)^2 + (4 - 2)^2 + (4 - 4)^2 + (4 - 3)^2 + (4 - 3)^2 + (2 - 4)^2 = 11
\end{align*}
\]

the distance between the two trees would be \( \sqrt{11/20} \approx 0.1658 \) or 16.58%.

Conducted Experiments

- First Experiment:
  - Deal with biological sequences in general, DNA and protein sequences.
  - Measure the compression methods against alignment methods.
- Second Experiment:
  - Deal with DNA sequences that have specific properties.
  - Use sequences with high degree of repetitions.
- Third Experiment:
  - Incomplete sequences and several fragments from the sequences.
  - Several fragments from the sequences not in order.
Input Datasets

- The first datasets:
  - **CK-36-PDB**: Chew-Se德 dataset of 36 protein domains, amino acid sequences in FASTA format.
  - **AA-15-DNA**: Apostolico dataset of 15 species, mitochondrial DNA complete genomes.

- The second datasets (the mitochondrial DNA complete genomes), as the genomes fragments were extracted using this dataset.

---

**Results**

Comparisons of the compression algorithms and multiple sequence alignment for the protein dataset CK-36-PDB

<table>
<thead>
<tr>
<th>Test Algorithm</th>
<th>Variant</th>
<th>Neighbor-joining</th>
<th>UPGMA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kolmogorov using Huffman coding</td>
<td>CD</td>
<td>2.316244</td>
<td>2.064698</td>
</tr>
<tr>
<td></td>
<td>NCD</td>
<td>2.24362</td>
<td>2.24362</td>
</tr>
<tr>
<td></td>
<td>UCD</td>
<td>2.316244</td>
<td>2.064698</td>
</tr>
<tr>
<td>Kolmogorov using LZW compression</td>
<td>CD</td>
<td>2.189895</td>
<td>2.189895</td>
</tr>
<tr>
<td></td>
<td>NCD</td>
<td>2.117764</td>
<td>2.117764</td>
</tr>
<tr>
<td></td>
<td>UCD</td>
<td>2.189895</td>
<td>2.189895</td>
</tr>
<tr>
<td>Lempel-Ziv complexity</td>
<td>Distance 1</td>
<td>2.174544</td>
<td>2.06988</td>
</tr>
<tr>
<td></td>
<td>Distance 2</td>
<td>2.189895</td>
<td>2.189895</td>
</tr>
<tr>
<td></td>
<td>Distance 3</td>
<td>2.244911</td>
<td>2.244911</td>
</tr>
<tr>
<td></td>
<td>Distance 4</td>
<td>2.2291</td>
<td>2.2291</td>
</tr>
<tr>
<td>Multiple Sequence Alignment</td>
<td>2.117764</td>
<td>2.06988</td>
<td></td>
</tr>
</tbody>
</table>

Comparisons of the compression algorithms and multiple sequence alignment for the Mitochondrial genome dataset in experiment 1

<table>
<thead>
<tr>
<th>Test Algorithm</th>
<th>Variant</th>
<th>Neighbor-joining</th>
<th>UPGMA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kolmogorov using Huffman coding</td>
<td>CD</td>
<td>1.97454</td>
<td>2.12062</td>
</tr>
<tr>
<td></td>
<td>NCD</td>
<td>1.95963</td>
<td>2.12062</td>
</tr>
<tr>
<td></td>
<td>UCD</td>
<td>1.97454</td>
<td>2.12062</td>
</tr>
<tr>
<td>Kolmogorov using LZW compression</td>
<td>CD</td>
<td>1.914474</td>
<td>1.926759</td>
</tr>
<tr>
<td></td>
<td>NCD</td>
<td>1.90894</td>
<td>1.926759</td>
</tr>
<tr>
<td></td>
<td>UCD</td>
<td>1.914474</td>
<td>1.926759</td>
</tr>
<tr>
<td>Lempel-Ziv complexity</td>
<td>Distance 1</td>
<td>2.139806</td>
<td>2.139806</td>
</tr>
<tr>
<td></td>
<td>Distance 2</td>
<td>2.150504</td>
<td>2.150504</td>
</tr>
<tr>
<td></td>
<td>Distance 3</td>
<td>2.160706</td>
<td>2.160706</td>
</tr>
<tr>
<td></td>
<td>Distance 4</td>
<td>2.170909</td>
<td>2.170909</td>
</tr>
<tr>
<td>Multiple Sequence Alignment</td>
<td>1.5547053</td>
<td>1.878762</td>
<td></td>
</tr>
</tbody>
</table>
Comparisons of the compression algorithms and multiple sequence alignment for the Mitochondrial genomes

<table>
<thead>
<tr>
<th>Test Algorithm</th>
<th>NCO</th>
<th>UPGMA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kolmogorov using Huffman coding</td>
<td>12.3431994</td>
<td>17.8482317</td>
</tr>
<tr>
<td>UCD</td>
<td>16.1252195</td>
<td>16.3215466</td>
</tr>
<tr>
<td>NCD</td>
<td>9.4403764</td>
<td>10.5261928</td>
</tr>
</tbody>
</table>

Lempel-Ziv complexity

| Distance 1                  | 0.00000000 |
| Distance 2                  | 0.00000000 |
| Distance 3                  | 0.00000000 |
| Distance 4                  | 0.00000000 |
| Multiple Sequence Alignment | 0.4403764  | 0.00000000 |

Using Compression to Compare Incomplete Fragments of Genomes

Results of Third Experiment

fragments of genomes, not continuous, not ordered
Using Fragmented Genomes

Analysis of Results

• Compression techniques are more like to cluster genomes with errors, as compression look at these data in a linear fashion rather than in a parallel fashion
• Multiple sequence alignment does not consider the input domain in obtaining similarity measures which limits its use for a diverse input set
• Compression methods identify important signals/motifs in the input sequences and use them in the process

Summary

• Next Generation Bioinformatics Tools need to be Intelligent, Collaborative, and Dynamic
• Biomedical scientists and Bioinformatics researchers need to work together to best utilize the combination of tools development and domain expertise
• The outcome of such collaboration has the potential of achieving explosive results with significant impact on human health and overall understanding of biological mysteries
Using Restriction Endonuclease Cut Order for Classification and Identification of Fungal Sequences

Sequence Identification/Classification

Current Approaches
1. Computational approach – Pairwise local and Multiple Sequence Alignment
2. Laboratory Method – RFLP, Southern Blotting

Existing Methods - Limitations

Pairwise or Multiple Alignment
1. Alignment is 'fine-grained' approach
2. More computation intensive and so NP hard for large dataset
3. Introduces gaps – gaps are interpreted as evolutionary events in molecular phylogeny, misaligned sequences have no useful biological information
4. Heuristics like BLAST is employed

Laboratory Methods (RFLP)
1. Only feasible for few sequences
2. Human and procedural error
3. In-silico RFLP methods (TRFLP program) requires Alignment as the second step for sequence identification

Utilize 'coarse-grain-features' of RFLP/Restriction Enzyme in-silico as opposed to the 'fine-grain-features' of alignment computationally.
Restriction Endonuclease

- Proteins that recognize a specific nucleotide subsequence
- Generally 4 to 8 bases long
- Cuts double stranded DNA molecule

Example:

Hae III recognizes GAG | CG site and cuts the product into two fragments as shown, adjacent G and C.

$\text{TTTTA0000000GT00000000000A00A \ldots GAG}$ (size = 600bp)

$\text{TTTTA0000000GT00000000000A00A}$

Fragment 1
Size = 100bp

$\text{CCA000000000000000000000A00A \ldots GAG}$

Fragment 2
Size = 500bp

RFLP

- RFLP = Restriction Fragment Length Polymorphism
  - Widely used laboratory method in molecular identification and Phylogenetic studies.
  - Requires the sequences to be cut into smaller fragments with the help of restriction endonucleases.
  - Variation in the position of these sites along the DNA leads to fragments of varying lengths.
  - High-resolution gel electrophoresis of the digested product, allows visualization of fragment-patterns, which is used to determine the similarity between the sequences.
Enzyme Cut Order (ECO)

- ECO of a DNA sequence S for a particular set of restriction enzymes \( \{ E_i \} \) is an ordered sequence of enzymes in the order each enzyme \( \text{cut} \) the sequence
  - ECO may also include position of nucleotide from the start of sequence where the cut occur.
- ECO is an array of pairs consisting of enzyme id and cut position
- Example:
  
<table>
<thead>
<tr>
<th>ECO ID</th>
<th>Enzyme Name</th>
<th>Cut Site</th>
<th>Cut POS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SmaI</td>
<td>3</td>
<td>(GCGCC)</td>
</tr>
<tr>
<td>2</td>
<td>ApaL</td>
<td>2</td>
<td>(CGC)</td>
</tr>
</tbody>
</table>
  
  (GCGCC is the best and end is shown by \( \ldots \))

The Enzyme Cut Order is \( \{ 2,1,2 \} \)
The Enzyme Cut Order with position is \( \{ (2,5),(0,1),(2,7) \} \)

Enzyme Cut Order

<table>
<thead>
<tr>
<th>Acc. Id</th>
<th>Organism</th>
<th>Enzyme Cut Order</th>
</tr>
</thead>
</table>

Observation:
Closely related organisms have similar Enzyme Cut Order

ECO Similarity Score

- The similarity score between two ECO should reflect:
  - Number of similar enzymes and
  - Order in which these enzyme cut the sequence
- Similarity score will be higher if we find larger number of similar enzymes appearing in the same order among two Enzyme Cut Orders.
- Similarity score can be obtained from the Longest Common Subsequence (LCS) among two strings, where the strings are the ECO
  - The length of Longest Common Subsequence (LCS) between two ECO (E1 and E2) of two corresponding sequences (S1 and S2) are considered as the Enzyme Cut Order Similarity Score between E1 and E2.
ECO Properties

- Enzyme Cut Order is a distinguishing characteristic of DNA sequences
- The similarity between two sequences can be defined by Enzyme Cut Order Similarity Score
- ECO-similarity score can be measured as the length of LCS among the corresponding Enzyme Cut Orders of the DNA sequences of the organisms

Main Hypothesis

- Organisms closer to each other in the Phylogenetic tree have highly similar Enzyme Cut Order
- Longest Common Subsequence (LCS) among the corresponding Enzyme Cut Orders is a quantitative measure to determine similarity
- Number of Enzymes used in the analysis determines the granularity of classification

ECO-Based Classification

- Step 1: Data Collection
- Step 2: Identify Enzyme Set
- Step 3: Obtain ECO
- Step 4: Calculated ECO Similarity Matrix
- Step 5: Clustering
- Step 6: Build Phylogenetic Tree
Step 1: Sequence Data Collection and Curation

- Created a local database of GenBank sequences obtained in FASTA or XML format.
- Reference these sequences against a taxon database.
- Create a curated taxonomy database for these sequences using user-defined taxonomic rules.
- Fungi ITS Sequences from GenBank
  - Organization description" of the genbank entries (or OrgName LINEAGE in XML format).
  - Classification categories included Kingdom, Division, Class, Order, Family, Genus, Species.
- Use a simple suffix rule and the position to decide.

<table>
<thead>
<tr>
<th>Suffix</th>
<th>Position</th>
<th>Tax Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ota</td>
<td>2nd</td>
<td>Fungi</td>
</tr>
<tr>
<td>&lt;ota</td>
<td>Between Division and Order</td>
<td>Class</td>
</tr>
<tr>
<td>&lt;ota</td>
<td>Between Class and Family</td>
<td>Order</td>
</tr>
<tr>
<td>&lt;ota</td>
<td>Below the Lower Class</td>
<td>Family</td>
</tr>
</tbody>
</table>

Step 2: Enzyme Data Collection

- Create a database of restriction enzymes obtained from REBASE.
- Add more relevant information about these restriction enzymes (Isoschizomers, Commercial availability, Reverse Cutsite) for later potential use.
- Appropriate recognition sequence containing bases other than A, T, G and C were interpreted as per IUB ambiguity code (Eur. J. Biochem. 156: 1-5, 1985).

<table>
<thead>
<tr>
<th>Enz ID</th>
<th>Enzyme Name</th>
<th>Recognition Sequence</th>
<th>Recognition Impurity</th>
<th>CoProduct</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A</td>
<td>GACTGATGTTG</td>
<td>CACCTCGATG</td>
<td>12</td>
</tr>
<tr>
<td>3</td>
<td>B</td>
<td>AGCTCTCGCA</td>
<td>AGCTCTCGCA</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>C</td>
<td>GTCTACCTG</td>
<td>GTCTACCTG</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>D</td>
<td>GCCTGCGG</td>
<td>GCCTGCGG</td>
<td>4</td>
</tr>
</tbody>
</table>
Step 3: Build Enzyme Cut Order DB

- Obtained Enzyme cut order using user defined set of restriction enzymes (Ez).
- The Enzyme cutorder is obtained for every test sequences and every enzyme in (Ez).
- Evaluate the effect of the size and type of restriction endonuclease. Different sets of (Ez) were chosen with the following properties:
  1. Enzymes that cut at least one of the sequences from the given sequence data.
  2. Enzymes that cut 50% of the sequences of the given sequence data.
  3. Enzymes that cut all the sequences at least once.
  4. Random enzyme set (consisting a mixture from the sets listed previously).
  5. Commonly used restriction enzymes in a biology laboratory working with the RFLP of fungi.

Step 4: Similarity Matrix Score

- Create a similarity matrix for a complete weighted graph for each Enzyme Set (Ez)
  - each node represents one of the input sequences and the weight between two nodes is similarity score SS using the LCS between two corresponding enzyme cut-order
  - Gx = (V,E) where each node v ∈ V represents a sequence and the weight on each edge e_{v1,v2} ∈ E is the similarity score between the two sequences represented by v1 and v2

Step 5: Clustering

The Similarity matrix is clustered and the cluster is analyzed for its phylogenetic accuracy.

Clustering algorithms employed:
- Maximum gap based exclusive clustering
- Hierarchical clustering
- Similarity Clustering
  - Newly developed clustering algorithm
  - Based on node merging and clique cover in the constructed graph
Experiments

- Collect Sequence Data
  - Target: Internal Transcribed Spacer (ITS) region of rRNA
  - Source: GenBank
- Extract taxonomic information from GenBank sequences
- Collect Restriction Endonuclease (RE) information
- Store Sequence and RE information in a local database
- Three sets of data
  1. AspCan: Sequences form the genus Aspergillus and Candida
  2. All9Genus (Randomly chosen 9 genera from AllFungi)
  3. AllFungi

Sequence Database

<table>
<thead>
<tr>
<th>Division</th>
<th>Class</th>
<th>Order</th>
<th>Family</th>
<th>Genus</th>
<th>Species</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basidiomycota</td>
<td>3</td>
<td>45</td>
<td>142</td>
<td>261</td>
<td></td>
</tr>
<tr>
<td>Ascomycota</td>
<td>11</td>
<td>20</td>
<td>49</td>
<td>137</td>
<td>443</td>
</tr>
<tr>
<td>Glomeromycota</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Zygomycota</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td>Chytridiomycota</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Impact of Using Different Enzyme Sets

- **Sequence (Set-1)**
  - Type = Internal Transcribed Spacer
  - Size (N) = 7
  - Taxonomy
    - Ascomycota = 5
      - Nectria sp. = 3
      - Leotia sp. = 2
    - Basidiomycota = 2
      - Olpiomyces sp. = 2
- **Enzyme (Set1)**
  - Example = TaqI, HaeIII, HinfI, Alul, RsaI, MspI
  - Size (N) = 6
  - Property = Frequent cutter
- **Enzyme (Set2)**
  - Size (N) = 57
Results using Enzyme Set 1

- All sequences are perfectly clustered
- Similarity Gap is close and reflected on highlighted samples

<table>
<thead>
<tr>
<th>Org.</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Results using Enzyme Set 2

1. Obtained better Similarity Matrix (Higher Similarity Gap)
2. Larger Enzyme set may have better clustering result
3. All Sequences are perfectly clustered

<table>
<thead>
<tr>
<th>Acc ID</th>
<th>Genus</th>
<th>Species</th>
<th>0</th>
<th>102</th>
<th>104</th>
<th>75</th>
<th>73</th>
<th>57</th>
<th>57</th>
</tr>
</thead>
<tbody>
<tr>
<td>AY10834</td>
<td>Nectria</td>
<td>haematococca</td>
<td>0</td>
<td>102</td>
<td>104</td>
<td>75</td>
<td>73</td>
<td>57</td>
<td>57</td>
</tr>
<tr>
<td>AY10835</td>
<td>Nectria</td>
<td>haematococca</td>
<td>103</td>
<td>0</td>
<td>136</td>
<td>72</td>
<td>70</td>
<td>62</td>
<td>62</td>
</tr>
<tr>
<td>AY10836</td>
<td>Nectria</td>
<td>haematococca</td>
<td>104</td>
<td>128</td>
<td>7</td>
<td>70</td>
<td>65</td>
<td>65</td>
<td></td>
</tr>
<tr>
<td>AF42441</td>
<td>Lankia</td>
<td>magnifica</td>
<td>75</td>
<td>72</td>
<td>72</td>
<td>0</td>
<td>123</td>
<td>64</td>
<td>64</td>
</tr>
<tr>
<td>AF42442</td>
<td>Lankia</td>
<td>magnifica</td>
<td>75</td>
<td>70</td>
<td>73</td>
<td>523</td>
<td>0</td>
<td>64</td>
<td>64</td>
</tr>
<tr>
<td>OP524289</td>
<td>Oligoporus</td>
<td>placenter</td>
<td>57</td>
<td>62</td>
<td>60</td>
<td>64</td>
<td>64</td>
<td>0</td>
<td>118</td>
</tr>
<tr>
<td>OP524289</td>
<td>Oligoporus</td>
<td>placenter</td>
<td>57</td>
<td>62</td>
<td>60</td>
<td>64</td>
<td>64</td>
<td>198</td>
<td>8</td>
</tr>
</tbody>
</table>

How to Find an Optimal Enzyme Set?

- Optimal enzyme set is defined as the minimal size enzyme set that shows highest phylogenetic resolution
- Find optimal enzyme set for a particular dataset using genetic algorithms
- The Fitness Function is based on the expected and actual count of an organism in the cluster. The score is quantitatively determined in terms of Sensitivity and Positive Predictive Value
- The Selection is either Roulette-wheel selection, tournament selection or random selection
- Uniform, Single-Point or Two-Point crossover is used along with a user specified crossover rate
GA Results

<table>
<thead>
<tr>
<th>DB</th>
<th>Species</th>
<th>Clustering Method</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>50</td>
<td>3.4, 4, 16, 39, 48, 72, 97, 10, 122, 122, 122, 122, 122, 122, 122, 122, 197, 298 (11)</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>3.4, 72, 152, 250 (6)</td>
</tr>
<tr>
<td></td>
<td>26</td>
<td>3.4, 3, 572 (6)</td>
</tr>
</tbody>
</table>

AHPGen 21

| 4, 11, 10, 22, 24, 28, 33, 36, 47, 50, 57, 64, 71, 73, 82, 93, 97, 98, 100, 109, 232, 192, 143, 14 |
| 7, 40, 96, 172, 172, 172, 172, 172, 209, 10 |
| 3, 5, 24, 33, 39, 47, 84, 197, 298, 10, 10 |
| 10, 10, 10, 10, 10, 10, 10, 10 |
| 15     | 4, 7, 3, 572 (6) |

Identification of Unknown Sequences

Clustering of Sequences from All9Genus DB

Division-level taxonomy
- Ascomycota
- Basidiomycota

Class-level taxonomy
- Dothideomycetes
- Chaetothynomycetes
- Sordaromycetes
Summary

• The inherent biological property of Restriction enzymes to recognize specific DNA a sequence is a valuable parameter in the analysis of DNA sequence.
• The order in which multiple RE cut multiple sequences can be modeled to classify set of DNA sequences at varying degree of detail.
• The length of LCS is a quantitative measure to determine similarity between sequences.
• The proposed alignment-free approach provides an alternative method for sequence identification and classification.

Identification of Microorganisms Using Curated Custom Databases

The Sequence Identification Problem

• Identification of organisms using obtained sequences is a very important problem.
• Relying on wet lab methods only is not enough.
• Employing identification algorithms using signature motifs to complement the experimental approaches.
• Currently, no robust software tool is available for aiding researchers and clinicians in the identification process.
• Such a tool would have to utilize biological knowledge and databases to identify sequences.
• Issues related to size of data and quality of data are suspect and would need to be dealt with.
Problem Definition and Motivation

- Identification of organisms using obtained sequences is a very important problem
- Relying on wet lab methods only is not enough
- Employing identification algorithms using signature motifs complement the experimental approaches
- Currently, no robust software tool is available for aiding researchers and clinicians in the identification process
- Such tool would have to utilize biological knowledge and databases to identify sequences
- Issues related to size of data and quality of data is suspect would need to be dealt with

The Computational Approach

- Sequence similarity and graph clustering are employed to identify unknown sequences
- Earlier results were not conclusive
- Local similarity in specific regions rather than global similarity is used, in particular, test validity of identifying *Mycobacterium* based on ITS region and 16S region
- Graph Clustering based on region similarity produced very good results, particularly when using ITS region
- Grammar based description of selected regions is used for identification

Custom Databases

- Allowing researchers to create custom sets of genetic data suited to their specific needs.
- Allowing researchers to control the quality of genetic data in their custom data sets through fine-tuning parameters.
- Searching data using optimal alignment algorithms, rather than using heuristic methods.
- Giving researchers/clinicians the ability to formulate sequence identification concepts and test their ideas against a validated database
- Incorporating information from GenBank if needed
Creating a Custom Database

Create Database Container

Create Database Region

Add Database Members

Add Database Sequence

Add Region Preference

Insert Region Preference

Add Another Region

How to Define Region Preferences

• Simple Definition
  – Letters (AGCT)
  – Wild Card (N)
  – Limits (wild cards, mismatches, Region Size)

• Grammar Based Definition
  – Employs regular expression for flexible region definitions
  – Powerful and Robust but a bit more complex
Regular Expressions

<table>
<thead>
<tr>
<th>Expression</th>
<th>Expression Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>.</td>
<td>any</td>
<td>matches any character</td>
</tr>
<tr>
<td>[ ]</td>
<td>character class</td>
<td>matches characters in brackets</td>
</tr>
<tr>
<td>*</td>
<td>repeat</td>
<td>matches zero or more times</td>
</tr>
<tr>
<td>+</td>
<td>repeat</td>
<td>matches one or more times</td>
</tr>
<tr>
<td>?</td>
<td>repeat</td>
<td>matches zero or one time</td>
</tr>
<tr>
<td>^</td>
<td>start</td>
<td>matches start of line</td>
</tr>
<tr>
<td>\d</td>
<td>digit</td>
<td>matches any digit</td>
</tr>
<tr>
<td>\w</td>
<td>word</td>
<td>matches any word character</td>
</tr>
<tr>
<td>\s</td>
<td>space</td>
<td>matches any space character</td>
</tr>
</tbody>
</table>

Table 1: Regular expressions

BioRegEx

<table>
<thead>
<tr>
<th>Expression</th>
<th>Expression Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>.</td>
<td>any</td>
<td>matches any character</td>
</tr>
<tr>
<td>[ ]</td>
<td>character class</td>
<td>matches characters in brackets</td>
</tr>
<tr>
<td>*</td>
<td>repeat</td>
<td>matches zero or more times</td>
</tr>
<tr>
<td>+</td>
<td>repeat</td>
<td>matches one or more times</td>
</tr>
<tr>
<td>?</td>
<td>repeat</td>
<td>matches zero or one time</td>
</tr>
<tr>
<td>^</td>
<td>start</td>
<td>matches start of line</td>
</tr>
<tr>
<td>\d</td>
<td>digit</td>
<td>matches any digit</td>
</tr>
<tr>
<td>\w</td>
<td>word</td>
<td>matches any word character</td>
</tr>
<tr>
<td>\s</td>
<td>space</td>
<td>matches any space character</td>
</tr>
</tbody>
</table>

Table 2: BioRegEx expressions

Sequence Validation

Sequence validation tool for sequence validation.
Searching in a Custom Database

Interpreting Search Results

The Mycobacterium Case Study

- 30 species associated with a variety of human and animal diseases such as tuberculosis
- Certain pathogenic species specific to humans. Some only affect animals
- Certain pathogenic species are drug-resistant
- Laboratory identification slow, tedious, and error-prone
- Sequencing provides an alternative to laboratory methods
- Researchers wanted to test validity of identifying Mycobacterium based on ITS region and 16S region
Case Study: Mycobacterium

- Enter Sequence
- Validate 16S Region
- Validate ITS Region
- Compute Similarity
- Output Results
- Validate Input Sequence

Results of Case Study

- Identified Mycobacterium to species & strain level
- 72 of 78 previously identified isolates identified
- 6 remaining may be new strains
- MAC-A correctly identified in BioDatabase.
- MAC-A mistakenly first identified as M.malmoense using NCBI BLAST against GenBank
- Highlights problems with GenBank data and BLAST heuristics

Nebraska gets its very own organism

- While trying to pinpoint the cause of a lung infection in local cancer patients, they discovered a previously unknown micro-organism. And they’ve named it "mycobacterium nebraskense," after the Cornhusker state.
- It was discovered few weeks ago using Mycoalign: A Bioinformatics program developed at PKU

Source: Omaha World Herald.
Clustering and Phylogeny

Clustering, Arrays and Trees

- Clustering is a key step in several Bioinformatics problems, primarily:
  - Analysis of Microarray data; and
  - Building evolutionary trees
- Many versions of the clustering problem are NP-hard, near-optimal solutions can be obtained using smart heuristics.
- A good example is Hierarchical clustering that assumes a constant molecular clock (rate of evolution) along all branches of the tree. Two closest sequences are clustered first, then next two closest, etc. A rooted tree is produced.

Data Analysis Cycle

- Data Generation and Collection
- Data Access, Storage and Retrieval
- Data Integration
- Data Visualization
- Analysis and Data Mining
- Decision Support
- Validation and Discovery
What Is Clustering?

A cluster is a collection of objects which are “similar” between them and are “dissimilar” to the objects belonging to other clusters.

It is about hiding information?

Clustering Problem Formulation

• Let \( O = \{O_1, O_2, \ldots, O_n\} \) be a set of \( n \) objects, and let \( \mathcal{C} = \{C_1, C_2, \ldots, C_k\} \) be a partition of \( O \) into \( k \) subsets such that \( \bigcup_{i=1}^{k} C_i = O \) and \( C_i \cap C_j = \emptyset \)

• \( C_i \) is called a cluster, and \( \mathcal{C} \) is a clustering solution.

• The goal of clustering is to determine the intrinsic grouping in a set of unlabeled data. The data should be homogeneous and well separated.

How to decide what constitutes a good clustering?

• No absolute “best” criterion which would be independent of the final aim of the clustering.

• Consequently, the user must supply this criterion, in such a way that the result of the clustering will suit their needs.
Applications of Clustering

• Viewing and analyzing vast amounts of biological data as a whole set can be perplexing
• It is easier to interpret the data if they are partitioned into clusters combining similar data points
• Importance of clustering is analyzing microarray data, regulation data, and network analysis data

Network Analysis in Aging

Validation
Inferring Gene Functionality

- Researchers want to know the functions of newly sequenced genes.
- Simply comparing the new gene sequences to known DNA sequences often does not give away the function of gene.
- For 40% of sequenced genes, functionality cannot be ascertained by only comparing to sequences of other known genes.
- Microarrays allow biologists to infer gene function even when sequence similarity alone is insufficient to infer function.
- Genome-wide variants are quickly emerging as a key alternative to obtain correlation relationships.

Design Microarray Experiments

- We can conclude the basic steps for designing a microarray experiment:
  1. Prepare DNA chip using your chosen target DNAs.
  2. Generate a hybridization solution containing a mixture of fluorescently labeled cDNAs.
  3. Incubate hybridization mixture containing fluorescently labeled cDNAs with your DNA chip.
  4. Detect bound cDNA using laser technology and store data in a computer.
  5. Analyze data using computational methods.
Using Microarrays

- **Green**: expressed only from control
- **Red**: expressed only from experimental cell
- **Yellow**: equally expressed in both samples
- **Black**: NOT expressed in either control or experimental cells

Microarray Data

- Microarray data are usually transformed into an intensity matrix
- The intensity matrix allows biologists to make correlations between different genes (even if they are dissimilar) and to understand how genes functions might be related

<table>
<thead>
<tr>
<th>Gene</th>
<th>Time X</th>
<th>Time Y</th>
<th>Time Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gene 1</td>
<td>10</td>
<td>8</td>
<td>10</td>
</tr>
<tr>
<td>Gene 2</td>
<td>10</td>
<td>0</td>
<td>9</td>
</tr>
<tr>
<td>Gene 3</td>
<td>4</td>
<td>8.6</td>
<td>3</td>
</tr>
<tr>
<td>Gene 4</td>
<td>7</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>Gene 5</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

Clustering of Microarray Data

- Plot each datum as a point in N-dimensional space
- Make a distance matrix for the distance between every two gene points in the N-dimensional space
- Genes with a small distance share the same expression characteristics and might be functionally related or similar.
- Clustering reveal groups of functionally related genes
Clustering of Microarray Data

Homogeneity and Separation Principles

- **Homogeneity**: Elements within a cluster are close to each other
- **Separation**: Elements in different clusters are further apart from each other
- ...clustering is not an easy task!

Given these points a clustering algorithm might make two distinct clusters as follows:

Bad Clustering

This clustering violates both Homogeneity and Separation principles

Close distances from points in separate clusters

Far distances from points in the same cluster
**Good Clustering**

This clustering satisfies both Homogeneity and Separation principles.

**Hard and Fuzzy Clustering**

- **Hard clustering (exclusive)** -- each object is assigned to one and only one cluster.
  - The separation of points is achieved by a straight line.
  - Much more popular than fuzzy clustering.
- **Fuzzy clustering (overlapping)** -- each object can belong to more than one clusters with different degrees of membership.
  - Add membership function; data will be associated to an appropriate membership value.

**Distance (Similarity) Measure**

- Clustering algorithms are distance (similarity) based.
- Correlation coefficient: a similarity measure.

\[ S(X,Y) = \frac{1}{m} \sum \frac{(X_i - \bar{X})(Y_i - \bar{Y})}{\Phi_x \Phi_y} \]

\[ \Phi_x = \sqrt{\frac{\sum (X_i - \bar{X})^2}{m}} \]

\[ \bar{X} = \frac{\sum X_i}{m} \]

- Euclidean distance: a dissimilarity measure.

\[ D(X,Y) = \sqrt{\sum (X_i - Y_i)^2} \]
Similarity Criteria between Clusters

Single Linkage

Complete Linkage

Average Linkage

Centroid Linkage

Problems of Distance Measure

- Each distance measure has the advantage and the disadvantage.
- Different formulas lead to different clustering results.
- Domain knowledge must be used to guide the formulation of a suitable distance measure for each particular application.

Examples

Well separated but not particularly homogeneous clusters

Homogeneous but not well separated clusters

The ideal situation
Clustering Techniques

- Agglomerative: Start with every element in its own cluster, and iteratively join clusters together
- Divisive: Start with one cluster and iteratively divide it into smaller clusters
- Hierarchical: Organize elements into a tree, where the length of the paths between leaves represent the distances between genes. Similar genes lie within the same subtrees.

Clustering Methods

- Hierarchical methods (Hierarchical clustering, Super-paramagnetic clustering, Message Passing Clustering)
- Partition methods (K-means)
- Graph theoretic methods (CLICK, CAST)
- Neural network approaches (SOM)
- Various bi-clustering algorithms
- Other approaches:
  - Bayesian clustering, Matrix tree incision, Spectral clustering...
- Clustering results can be different for different methods and distance metrics

Agglomerative hierarchical clustering algorithm

1. Start by assigning each item to a cluster, so that if you have N items, you now have N clusters, each containing just one item.
2. Find the closest (most similar) pair of clusters and merge them into a single cluster, so that now you have one cluster less.
3. Compute distances (similarities) between the new cluster and each of the old clusters.
4. Repeat steps 2 and 3 until all items are clustered into a single cluster of size N.

Note: There is no point in having all N items grouped in a single cluster but, once you have got the complete hierarchical tree, if you want k clusters you just have to cut the k-1 longest links (branches).
* Different distance measures (single, complete, average or centroid linkage) may result in different clustering/dendrogram.
Hierarchical Clustering Example: Fibroblast clustering

- Cholesterol biosynthesis
- Cell cycle
- Immediate-early response
- Signaling and angiogenesis
- Wound healing and tissue remodeling


Hierarchical Clustering

- Merging (agglomerative): start with every instance as a separate cluster then combine until only one cluster remains.
- Splitting (Divisive): make one large cluster, then split up into smaller pieces until each single instance is in a cluster.

Hierarchical Clustering

- Hierarchical Clustering is often used to reveal evolutionary history
Hierarchical Clustering Algorithm

1. Hierarchical Clustering \((d, n)\)
2. Form \(n\) clusters each with one element
3. Construct a graph \(T\) by assigning one vertex to each cluster
4. while there is more than one cluster
5. Find the two closest clusters \(C_1\) and \(C_2\)
6. Merge \(C_1\) and \(C_2\) into new cluster \(C\) with \(|C_1| + |C_2|\) elements
7. Compute distance from \(C\) to all other clusters
8. Add a new vertex \(C\) to \(T\) and connect to vertices \(C_1\) and \(C_2\)
9. Remove rows and columns of \(d\) corresponding to \(C_1\) and \(C_2\)
10. Add a row and column to \(d\) corresponding to the new cluster \(C\)
11. return \(T\)

The algorithm takes a \(n \times n\) distance matrix \(d\) of pairwise distances between points as an input. Different ways to define distances between clusters may lead to different clustering.

K-Means Clustering

- **Input**: A set, \(V\), consisting of \(n\) points and a parameter \(k\)
- **Output**: A set \(X\) consisting of \(k\) points (cluster centers) that minimizes the squared error distortion \(d(V, X)\) over all possible choices of \(X\)

K-Means

- Iteratively solve for optimal cluster centers and partitions
- Need to know the number of clusters ahead of time
- Function optimization method that iterates so that objects within clusters are most similar
- Usually uses Euclidean distance
- Solution is not unique, clustering can depend on your starting point
K-Means Algorithm

1. Choose K initial group (cluster) centers at random.
2. Assign each object to the group that has the closest centroid.
3. When all objects have been assigned, recalculate the positions of the K centroids for K clusters.
4. Repeat Steps 2 and 3 until the centroids no longer move.
   ✷ This produces a separation of the objects into groups from which the metric to be minimized can be calculated.

Example (k=3) Initialization

Iteration 1
## Comparison of clustering algorithms

- **Hierarchical clustering**
  - Widely used.
  - Easy to understand.
  - Does not require the number of clusters *a priori*.
  - Difficult to implement well.
  - Requires post-processing.
  - Unstable.
  - Greediness can lock in early mistakes.
  - There is no reason to think that expression data is organized hierarchically.

- **k-means**
  - Less widely used.
  - Easy to understand.
  - Requires the number of clusters *a priori*.
  - Easy to implement.
  - Scales well.
  - Stable.
  - Creates unorganized cluster that are hard to interpret.

## Bi-Clustering

- The term “Biclustering” was first used by Cheng and Church in gene expression data analysis.
- Gene expression data or expression data
- Data Matrix
  - Each gene – One row
  - Each condition – One column
  - Each element – expression level of a gene under specific condition
**Biclustering**

- The term “Biclustering” was first used by Cheng and Church in gene expression data analysis [Year 2000]

- Clusters do not need to include all parameters (genes in Bioinformatics) for all conditions

- Data Matrix
  - Each gene – One row
  - Each condition – One column
  - Each element – expression level of a gene under specific condition

---

**Clustering Versus Biclustering**

- Clustering –
  - Applied to either rows or columns of the data matrix separately
  - Each gene is defined using all the conditions
  - Each condition is characterized by the activity of all the genes that belong to it

- Biclustering –
  - Performs clustering in these two dimensions simultaneously
  - Each gene is selected using only a subset of the conditions
  - Each condition is selected using only a subset of the genes

---

**Traditional Clustering**

- Applied to either rows or columns of the data matrix separately
- Each gene is defined using all the conditions
- Each condition is characterized by the activity of all the genes that belong to it
**Biclustering (Cont.)**

- Performs clustering in these two dimensions simultaneously
- Each gene is selected using only a subset of the conditions
- Each condition is selected using only a subset of the genes

---

**Phylogeny**

- Science of estimating the evolutionary past
- Phylogenetic analysis is the means used to estimate evolutionary relationships based on observable evidence
- Evidence can include morphology, physiology, and other properties of organisms. Paleontological and geological evidence is also used.
- Linnaeus’s system of grouping and naming organisms to reflect evolutionary relationship

---

**Phylogenetic Trees**
Phylogenetic Analysis

Relationship among species

Exercise Day 2

• Sequence Comparison using alignment and non-alignment approaches
• Use the comparison scores to cluster a set of organisms and build a phylogenetic Trees