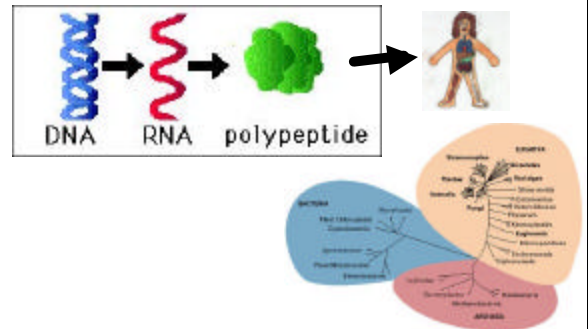


Computational Genomics



Lecture 1, Tuesday April 1, 2003

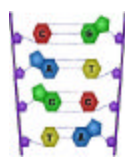
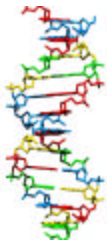
Biology in One Slide



Lecture 1, Tuesday April 1, 2003

High Throughput Biology

1. DNA Sequencing

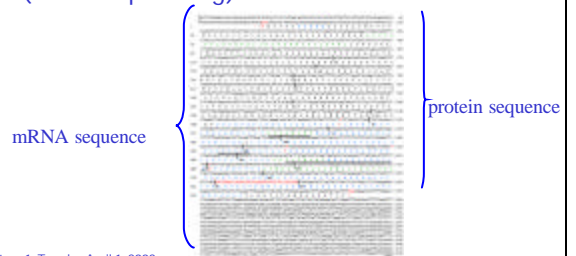


...ACGTGACTGAGGACCGTG
CGACTGAGACTGACTGGGT
CTAGCTAGACTACGTTTTA
TATATATATACGTCGCT
ACTGATGACTAGATTACAG
ACTGATTTAGATACCTGAC
TGATTTTAAAAAATATT...

Lecture 1, Tuesday April 1, 2003

High Throughput Biology

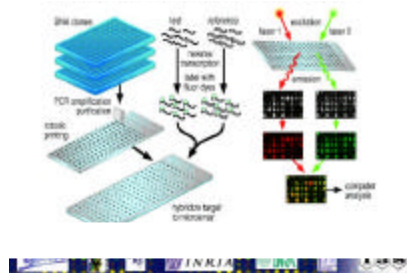
2. Sequencing of expressed genes (EST sequencing)



Lecture 1, Tuesday April 1, 2003

High Throughput Biology

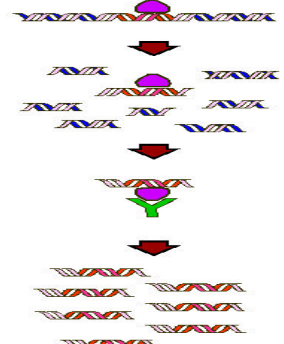
3. Gene Expression: Microarrays



Lecture 1, Tuesday April 1, 2003

High Throughput Biology

4. Gene Regulation: CH.I.P.



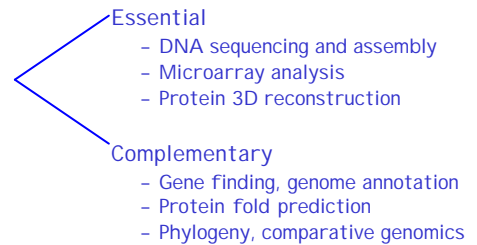
Lecture 1, Tuesday April 1, 2003

The goals of genomics

- Study organisms at the DNA level
 - Identify "parts" (genes, etc)
 - Figure out "connections" between "parts"
- Study evolution at the DNA level
 - Compare organisms
 - Uncover evolutionary history

Lecture 1, Tuesday April 1, 2003

The role of CS in Biology



Lecture 1, Tuesday April 1, 2003

Syllabus

- Tools
 - Alignment algorithms
 - Hidden Markov models
 - Statistical algorithms
- Applications
 - DNA sequencing and assembly
 - Sequence analysis (comparison, annotation)
 - Microarray analysis
 - Evolutionary analysis

Lecture 1, Tuesday April 1, 2003

Course responsibilities

- Homeworks [80%]
 - 4 challenging problem sets, 4 -5 problems/ pset
 - Collaboration allowed
 - 5 late days total
 - Televised students required to do 75%
- Final [20%]
 - Takehome, 1 day
 - Collaboration not allowed
 - Easy!
- Scribing
 - "Mandatory"
 - Grade replaces lowest 2 problems
 - Due one week after the lecture

Lecture 1, Tuesday April 1, 2003

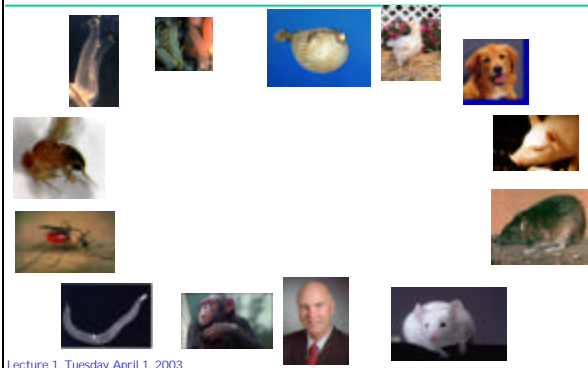
Reading material

- Books
 - "Biological sequence analysis" by Durbin, Eddy, Krogh, Mitchinson
 - Chapters 1-4, 6, (7-8), (9-10)
 - "Algorithms on strings, trees, and sequences" by Gusfield
 - Chapters (5-7), 11-12, (13), 14, (17)
- Papers
- Lecture notes

Lecture 1, Tuesday April 1, 2003

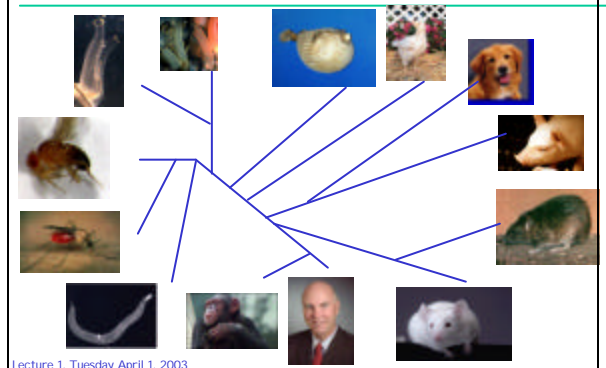
Topic 1. Sequence Alignment

Complete genomes



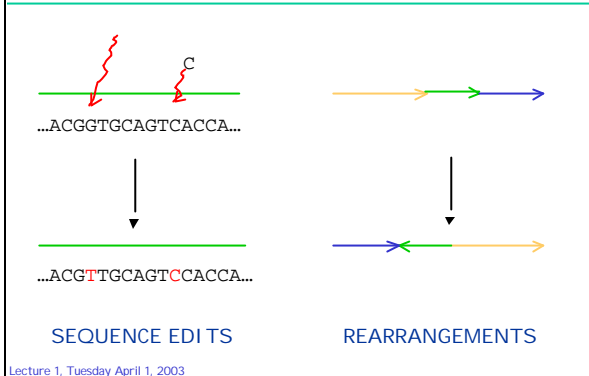
Lecture 1, Tuesday April 1, 2003

Evolution



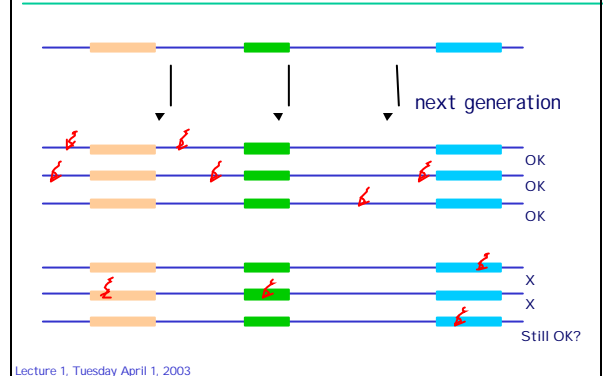
Lecture 1, Tuesday April 1, 2003

Evolution at the DNA level



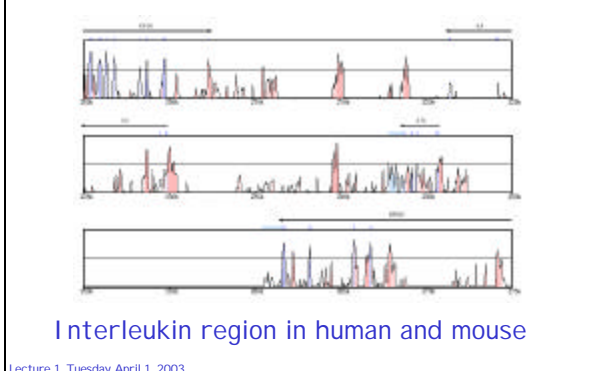
Lecture 1, Tuesday April 1, 2003

Evolutionary Rates



Lecture 1, Tuesday April 1, 2003

Sequence conservation implies function



Lecture 1, Tuesday April 1, 2003

Sequence Alignment

AGGCTATCACCTGACCTCCAGGCCGATGCC
TAGCTATCACGACCGCGGTCGATTGCCCCGAC

-AGGCTATCACCTGACCTCCAGGCCGA--TGCCC--
TAG-CTATCAC--GACCGC--GGTCGATTGCCCCGAC

Definition

Given two strings $x = x_1x_2...x_M$, $y = y_1y_2...y_N$

an alignment is an assignment of gaps to positions $0, ..., N$ in x , and $0, ..., N$ in y , so as to line up each letter in one sequence with either a letter, or a gap in the other sequence

Lecture 1, Tuesday April 1, 2003

What is a good alignment?

Alignment:

The "best" way to match the letters of one sequence with those of the other

How do we define "best"?

Alignment:

A hypothesis that the two sequences come from a common ancestor through sequence edits

Parsimonious explanation:

Find the minimum number of edits that transform one sequence into the other

Lecture 1, Tuesday April 1, 2003

Scoring Function

Sequence edits:

- Mutations
- Insertions
- Deletions

AGGCCTC

AGGA^ACTC

AGG^GCCTC

AGG⁻CTC

Scoring Function:

Match: +m

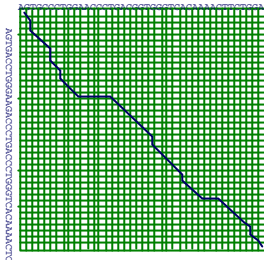
Mismatch: -s

Gap: -d

$$\text{Score } F = (\# \text{ matches}) \times m - (\# \text{ mismatches}) \times s - (\# \text{ gaps}) \times d$$

Lecture 1, Tuesday April 1, 2003

How do we compute the best alignment?



Too many possible alignments:

$$O(2^{M+N})$$

Lecture 1, Tuesday April 1, 2003

Alignment is additive

Observation:

The score of aligning

$x_1 \dots x_M$

$y_1 \dots y_N$

is additive

Say that
aligns to

$x_1 \dots x_i$
 $y_1 \dots y_j$

$x_{i+1} \dots x_M$
 $y_{j+1} \dots y_N$

The two scores add up:

$$F(x[1:M], y[1:N]) = F(x[1:i], y[1:j]) + F(x[i+1:M], y[j+1:N])$$

Lecture 1, Tuesday April 1, 2003

Dynamic Programming

- We will now describe a dynamic programming algorithm

Suppose we wish to align

$x_1 \dots x_M$

$y_1 \dots y_N$

Let

$F(i,j)$ = optimal score of aligning

$x_1 \dots x_i$

$y_1 \dots y_j$

Lecture 1, Tuesday April 1, 2003

Dynamic Programming (cont'd)

Notice three possible cases:

1. x_i aligns to y_j
 $x_1 \dots x_{i-1}$ x_i
 $y_1 \dots y_{j-1}$ y_j

$$F(i,j) = F(i-1, j-1) + \begin{cases} m, & \text{if } x_i = y_j \\ -s, & \text{if not} \end{cases}$$

2. x_i aligns to a gap
 $x_1 \dots x_{i-1}$ x_i
 $y_1 \dots y_j$ -

$$F(i,j) = F(i-1, j) - d$$

3. y_j aligns to a gap
 $x_1 \dots x_i$ -
 $y_1 \dots y_{j-1}$ y_j

$$F(i,j) = F(i, j-1) - d$$

Lecture 1, Tuesday April 1, 2003

Dynamic Programming (cont'd)

- How do we know which case is correct?

Inductive assumption:

$F(i, j-1)$, $F(i-1, j)$, $F(i-1, j-1)$ are optimal

Then,

$$F(i, j) = \max \begin{cases} F(i-1, j-1) + s(x_i, y_j) \\ F(i-1, j) - d \\ F(i, j-1) - d \end{cases}$$

Where $s(x_i, y_j) = m$, if $x_i = y_j$; $-s$, if not

Lecture 1, Tuesday April 1, 2003

Example

$x = \text{AGTA}$
 $y = \text{ATA}$

$m = 1$
 $s = -1$
 $d = -1$

$F(i, j)$		$i = 0$	1	2	3	4	
				A	G	T	A
$j = 0$		0	-1	-2	-3	-4	
1	A	-1	1	0	-1	-2	
2	T	-2	0	0	1	0	
3	A	-3	-1	-1	0	2	

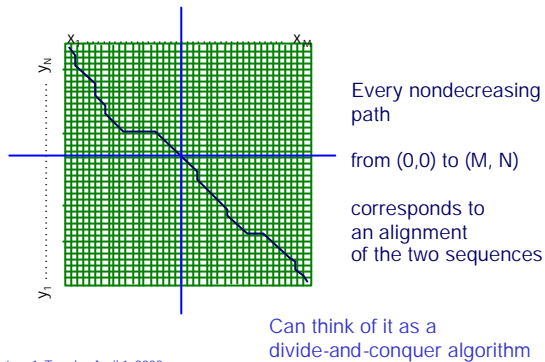
Optimal Alignment:

$F(4, 3) = 2$

AGTA
A - TA

Lecture 1, Tuesday April 1, 2003

The Needleman-Wunsch Matrix



Lecture 1, Tuesday April 1, 2003

The Needleman-Wunsch Algorithm

- Initialization
 - $F(0, 0) = 0$
 - $F(0, j) = -j \times d$
 - $F(i, 0) = -i \times d$
- Main Iteration, Filling-in partial alignments
 - For each $i = 1 \dots M$
For each $j = 1 \dots N$

$$F(i, j) = \max \begin{cases} F(i-1, j) - d & \text{[case 1]} \\ F(i, j-1) - d & \text{[case 2]} \\ F(i-1, j-1) + s(x_i, y_j) & \text{[case 3]} \end{cases}$$

$$\text{Ptr}(i, j) = \begin{cases} \text{UP} & \text{if [case 1]} \\ \text{LEFT} & \text{if [case 2]} \\ \text{DIAG} & \text{if [case 3]} \end{cases}$$
- Termination, $F(M, N)$ is the optimal score, and from $\text{Ptr}(M, N)$ can trace back optimal alignment

Lecture 1, Tuesday April 1, 2003

Performance

- Time: $O(NM)$
- Space: $O(NM)$
- Later we will cover more efficient methods

Lecture 1, Tuesday April 1, 2003

A variant of the basic algorithm:

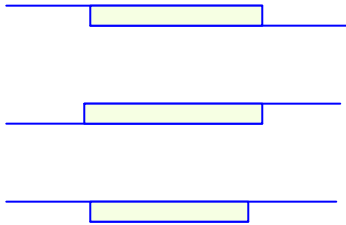
- Maybe it is OK to have an unlimited # of gaps in the beginning and end:

-----CTATCACCTGACCTCCAGGCCGATGCCCTTCCGGC
GCGAGTTCATCTATCAC--GACCGC--GGTCG-----

- Then, we don't want to penalize gaps in the ends

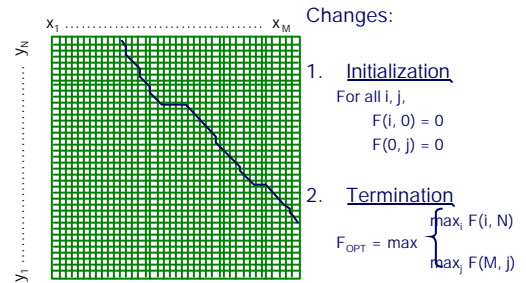
Lecture 1, Tuesday April 1, 2003

Different types of overlaps



Lecture 1, Tuesday April 1, 2003

The Overlap Detection variant



Lecture 1, Tuesday April 1, 2003

Next Lecture

- Local alignment
- More elaborate scoring function
- Memory-efficient algorithms

Reading:

Durbin, Chapter 2
Gusfield, Chapter 11

Lecture 1, Tuesday April 1, 2003