

Data Mining:

Concepts and Techniques

— Chapter 8 —

8.4. Mining sequence patterns in biological data

Jiawei Han and Micheline Kamber

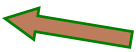
Department of Computer Science

University of Illinois at Urbana-Champaign

www.cs.uiuc.edu/~hanj

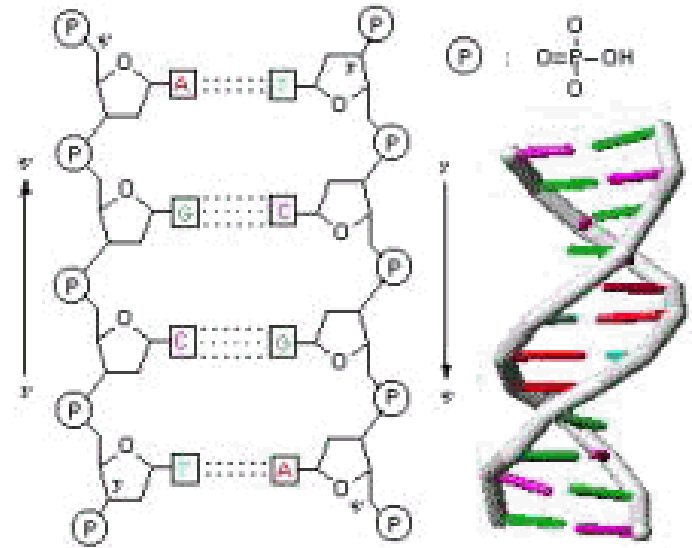
©2006 Jiawei Han and Micheline Kamber. All rights reserved.

Mining Sequence Patterns in Biological Data

- A brief introduction to biology and bioinformatics 
- Alignment of biological sequences
- Hidden Markov model for biological sequence analysis
- Summary

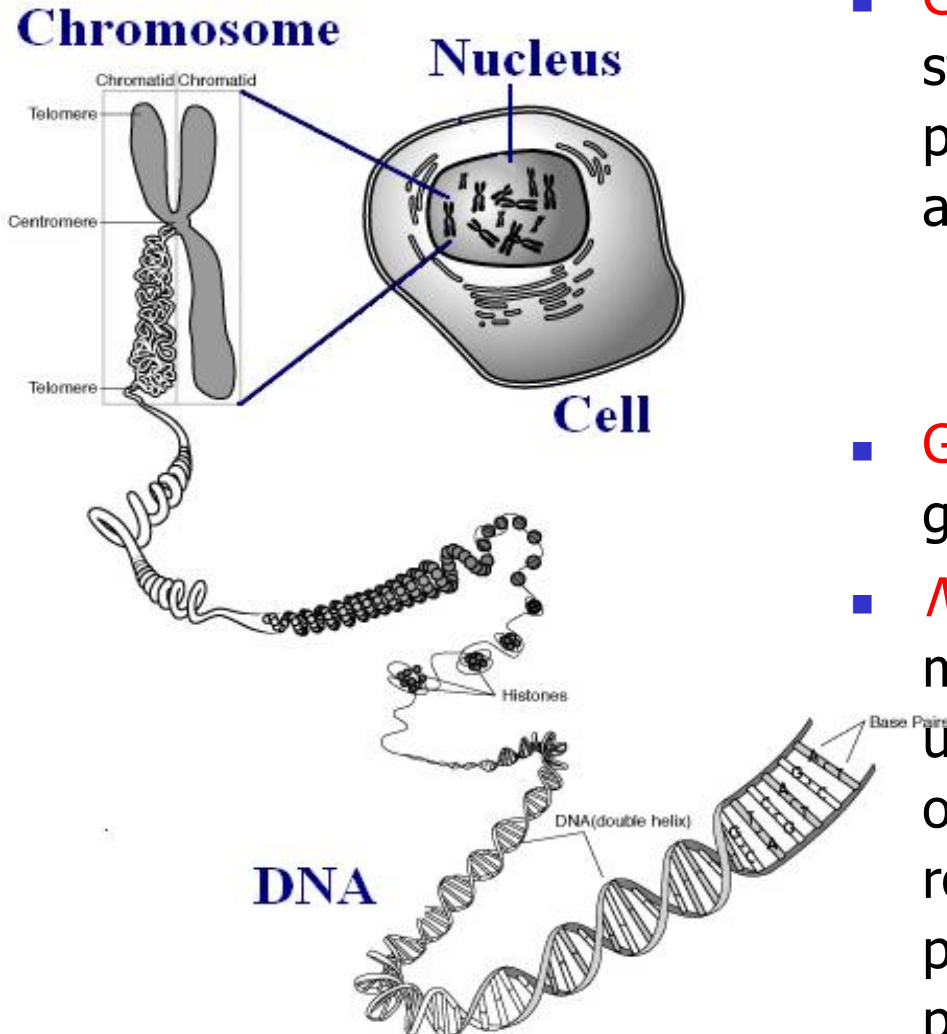
Biology Fundamentals (1): DNA Structure

- DNA: helix-shaped molecule whose constituents are two parallel strands of nucleotides
- DNA is usually represented by sequences of these four nucleotides
- This assumes only one strand is considered; the second strand is always derivable from the first by pairing A's with T's and C's with G's and vice-versa



- Nucleotides (bases)
 - Adenine (A)
 - Cytosine (C)
 - Guanine (G)
 - Thymine (T)

Biology Fundamentals (2): Genes



- **Gene:** Contiguous subparts of single strand DNA that are templates for producing *proteins*. Genes can appear in either of the DNA strand.
 - **Chromosomes:** compact chains of coiled DNA
- **Genome:** The *set of all genes* in a given organism.
- **Noncoding part:** The function of DNA material between genes is largely unknown. Certain intergenic regions of DNA are known to play a major role in *cell regulation* (controls the production of proteins and their possible interactions with DNA).

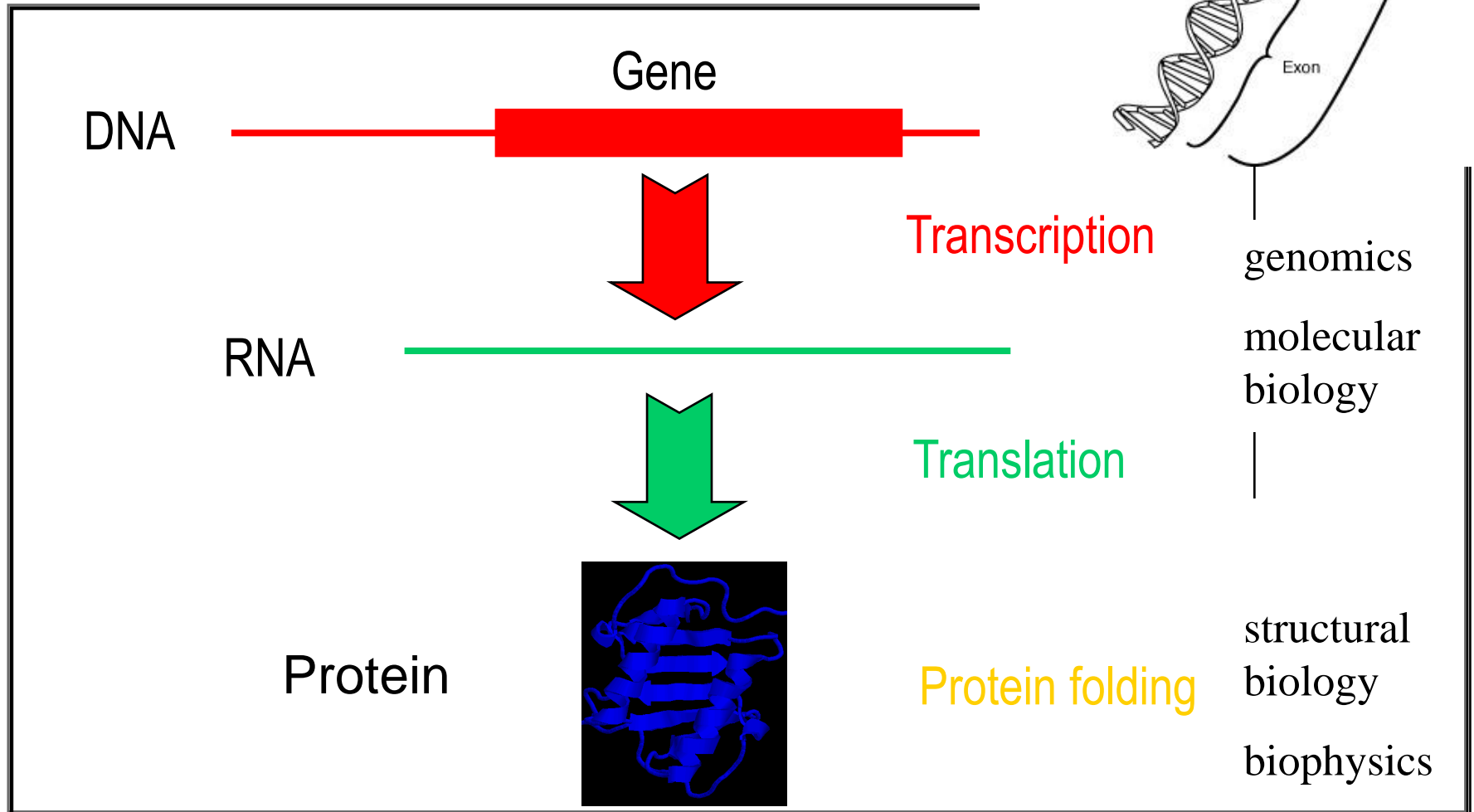
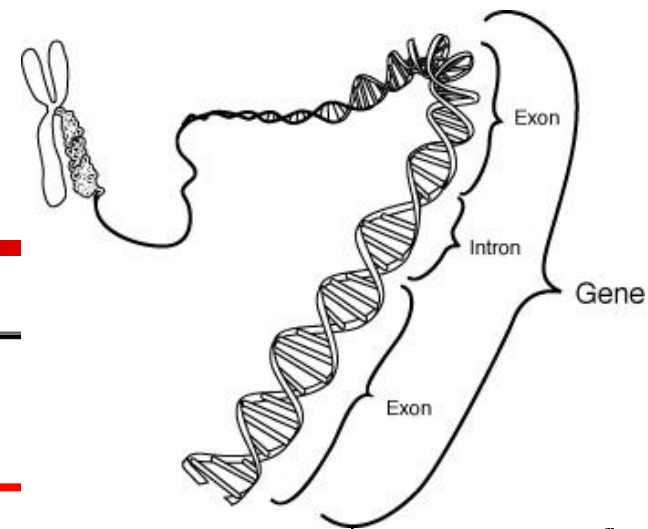
Biology Fundamentals (3): Transcription

- **Proteins:** Produced from DNA using 3 operations or transformations: *transcription*, *splicing* and *translation*
 - In *eukaryotes* (cells with nucleus): genes are only a minute part of the total DNA
 - In *prokaryotes* (cells without nucleus): the phase of splicing does not occur (no pre-RNA generated)
- DNA is capable of replicating itself (*DNA-polymerase*)
- Genes are *transcribed* into pre-RNA by a complex ensemble of molecules (*RNA-polymerase*). During transcription T is substituted by the letter U (for *uracil*).
- Pre-RNA can be represented by alternations of sequence segments called *exons* and *introns*. The exons represents the parts of pre-RNA that will be *expressed*, i.e., translated into proteins.

Biology Fundamentals (4): Proteins

- *Splicing* (by spliceosome—an ensemble of proteins): concatenates the exons and excises introns to form mRNA (or simply RNA)
- *Translation* (by ribosomes—an ensemble of RNA and proteins)
 - Repeatedly considers a *triplet* of consecutive nucleotides (called *codon*) in RNA and produces one corresponding amino acid
 - In RNA, there is one special codon called *start codon* and a few others called *stop codons*
- An **Open Reading Frame (ORF)**: a sequence of codons starting with a start codon and ending with an end codon. The ORF is thus a sequence of nucleotides that is used by the ribosome to produce the sequence of amino acid that makes up a protein.
- There are basically **20 amino acids** (A, L, V, S, ...) but in certain rare situations, others can be added to that list.

Biological Information: From Genes to Proteins



Biology Fundamentals (5): 3D Structure

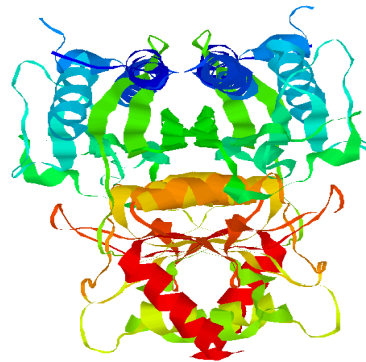
- Since there are 64 different codons and 20 amino acids, the “table look-up” for translating each codon into an amino acid is redundant: multiple codons can produce the same amino acid
- The table used by nature to perform translation is called the **genetic code**
- Due to the **redundancy** of the genetic code, certain nucleotide changes in DNA may not alter the resulting protein
- Once a protein is produced, it folds into a unique structure in 3D space, with 3 types of components: *α -helices*, *β -sheets* and *coils*.
- The **secondary structure** of a protein is its sequence of amino acids, annotated to distinguish the boundary of each component
- The **tertiary structure** is its **3D** representation

From Amino Acids to Proteins Functions

CGCCAGCTGGACGGGCACACC
 ATGAGGCTGCTGACCCTCCTG
 GGCCTTCTG...

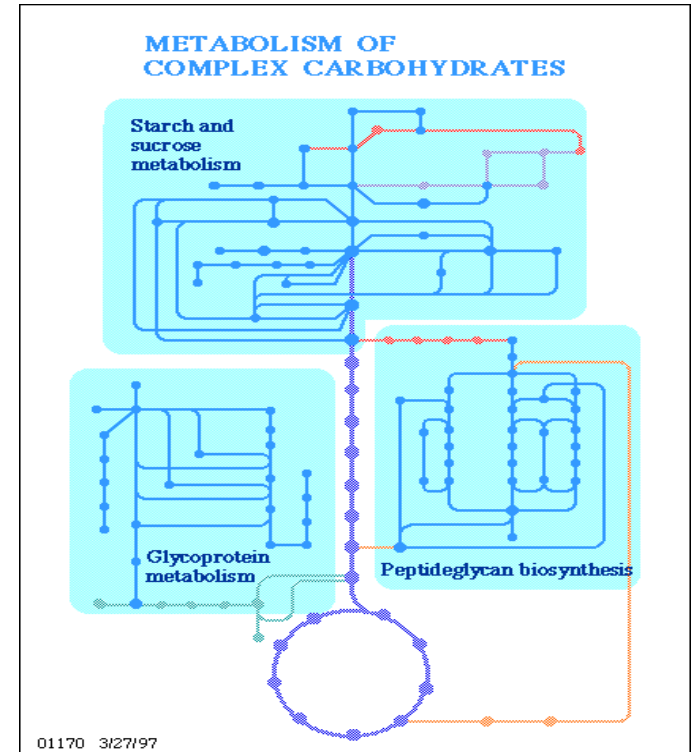


TDQAAFDTNIVTLTRFVMEQG
 RKARGTGEMTQLLNSLCTAVK
 AISTAVRKAGIAHLYGIAGST
 NVTGDQVKKLDVLSNDLVINV
 LKSSFATCVLVTEEDKNAIIV
 EPEKRGKYVVCFDPLDGSSNI
 DCLVSI GTIFGIYRKNSTDEP
 SEKDALQPGRNLVAAGYALYG
 SATML



DNA / amino acid
 sequence

3D structure



protein functions

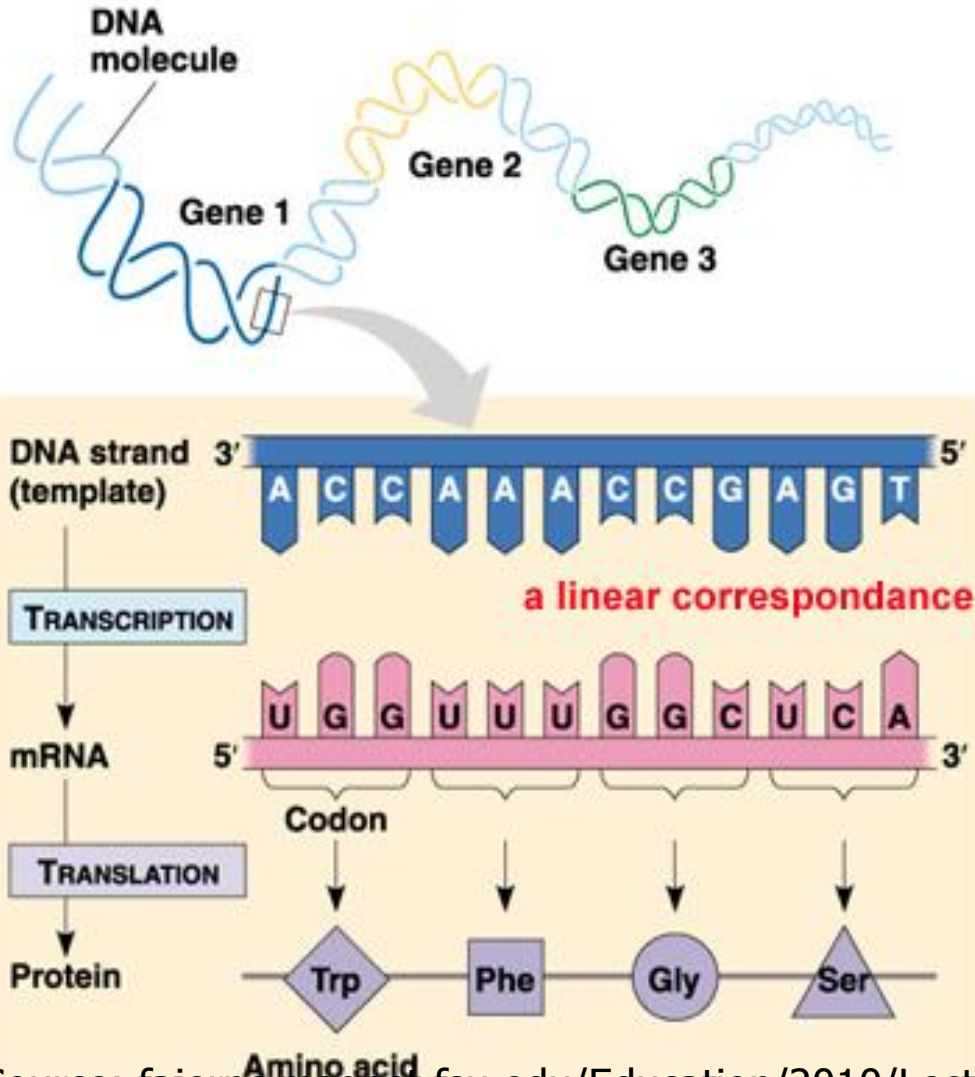
DNA (gene) → → → pre-RNA → → → RNA → → → Protein

RNA-polymerase

Spliceosome

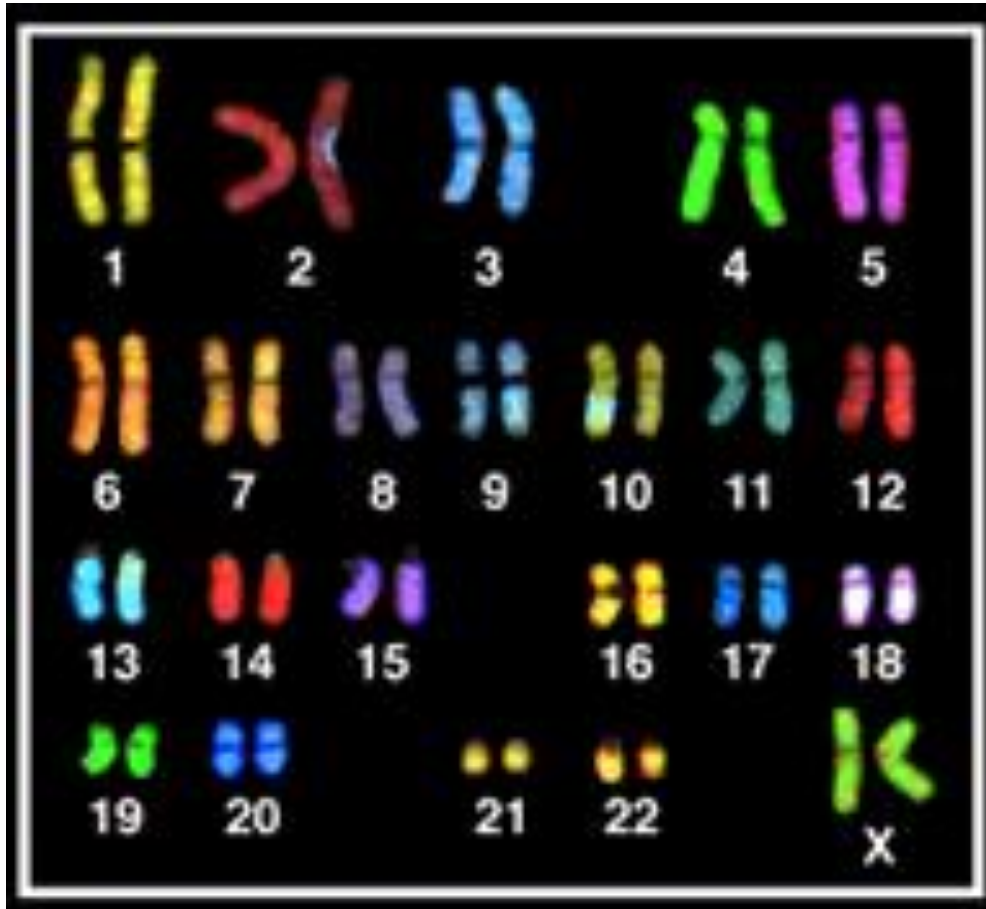
Ribosome

Biology Fundamentals (6): Functional Genomics



- The *function* of a protein is the way it participates with other proteins and molecules in keeping the cell alive and interacting with its environment
- Function is closely related to tertiary structure
- *Functional genomics*: studies the function of all the proteins of a genome

Biology Fundamentals (7): Cell Biology



Human Genome—23 pairs of chromosomes

Source: www.mtsinai.on.ca/pdmg/images/pairscolor.jpg

- A cell is made up of molecular components that can be viewed as 3D-structures of various shapes
- In a living cell, the molecules interact with each other (w. shape and location). An important type of interaction involve catalysis (*enzyme*) that facilitate interaction.
- A *metabolic pathway* is a chain of molecular interactions involving enzymes
- *Signaling pathways* are molecular interactions that enable communication through the cell's membrane

Lab Tools for Determining Bio. Data (I)

- *Sequencer*: machines capable of reading off a sequence of nucleotides in a strand of DNA in biological samples
 - It can produce 300k base pairs per day at relatively low cost
 - A user can order from biotech companies vials containing short sequences of nucleotides specified by the user
- Since sequences gathered in a wet lab consist of short random segments, one has to use the *shotgun method* (a program) to reassemble them
 - Difficulty: redundancy of seq. and ambiguity of assembly.
- *Mass spectroscopy*: identifies proteins by cutting them into *short* sequences of amino acids (*peptides*) whose molecular weights can be determined by a mass spectrograph, and then computationally infer the constituents of peptides

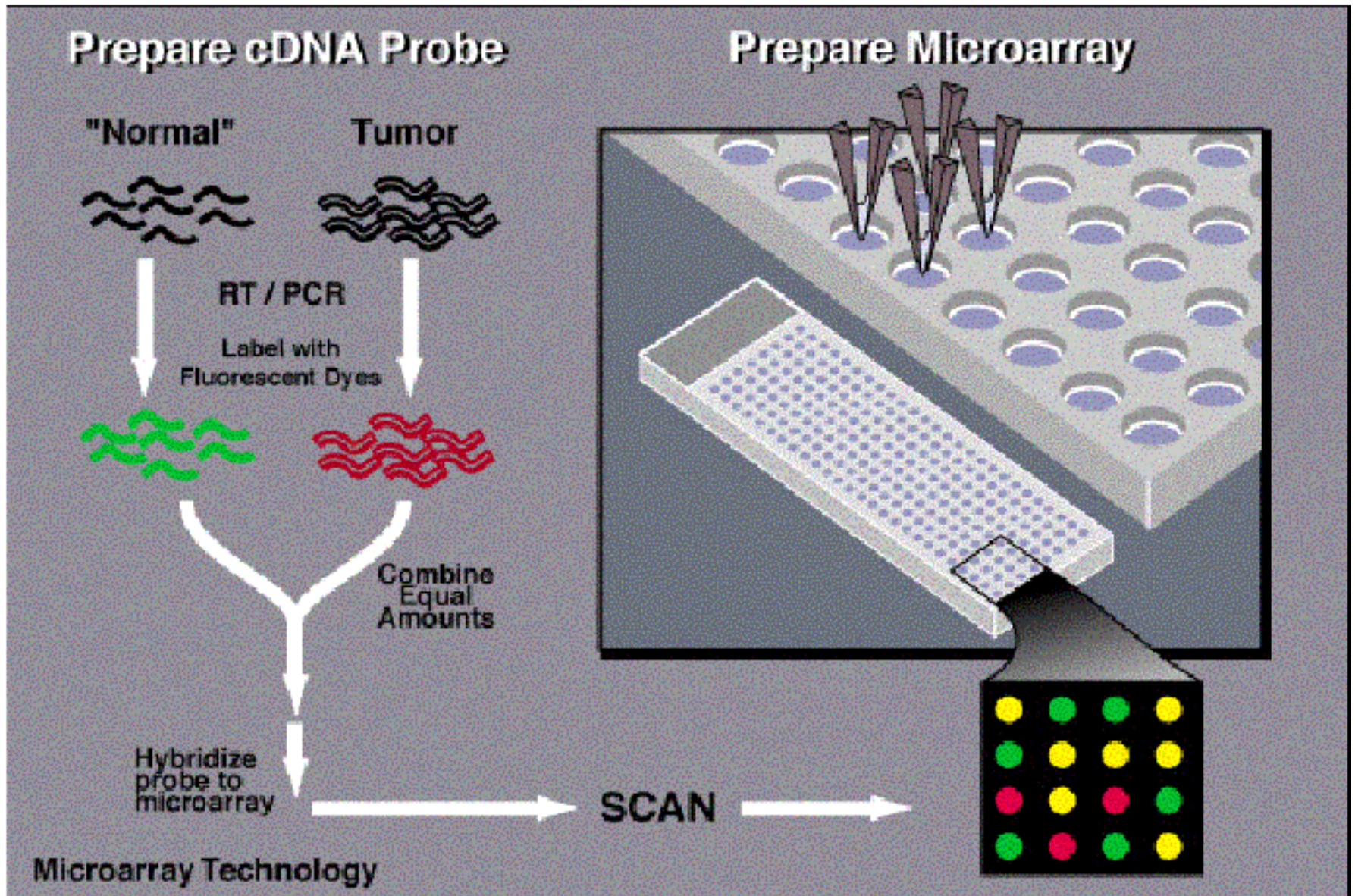
Lab Tools for Determining Bio. Data (II)

- The *3D-structure* of proteins is mainly determined (costly) by
 - *X-ray crystallography*: X-ray passing through a *crystallized* sample of that protein, and
 - *nuclear magnetic resonance (NMR)*: obtain a number of matrices that express that fact that two atoms are within a certain distance and then deduce a 3D shape
- *Expressed sequence tags (ESTs)*: RNA chunks that can be gathered from a cell in minute quantities (not containing the materials that would be present in introns), can be used to infer positions of introns
- *Libraries of variants of a given organism*:
 - Each variant may correspond to cells having a single one of its genes knocked out
 - Enable biologists to perform experiments and deduce information about cell behavior and fault tolerance
 - *RNA-i*: (the *i* denoting interference): chunks of the RNA of a given gene are inserted in the nucleus of a cell, that may prevent the production of that gene

Lab Tools for Determining Bio. Data (III)

- *Microarrays*: determine simultaneously the amount of mRNA production (gene expression) of thousands of genes. It has 3 phases:
 - Place thousands of different one-strand chunks of RNA in minuscule wells on the surface of a small glass chip
 - Spread genetic material obtained by a cell experiment one wishes to perform
 - Use a laser scanner and computer to measure the amount of combined material and determine the degree (a real number) of gene expression for each gene on the chip
- *Protein-arrays*: chips whose wells contain molecules that can be bound to particular proteins (for study of protein expression)
- Determining protein interaction by *two-hybrid* experiments:
 - Construct huge Boolean matrices, whose rows and columns represent the proteins of a genome
 - If a protein interacts with another, the corresp. position is set to true

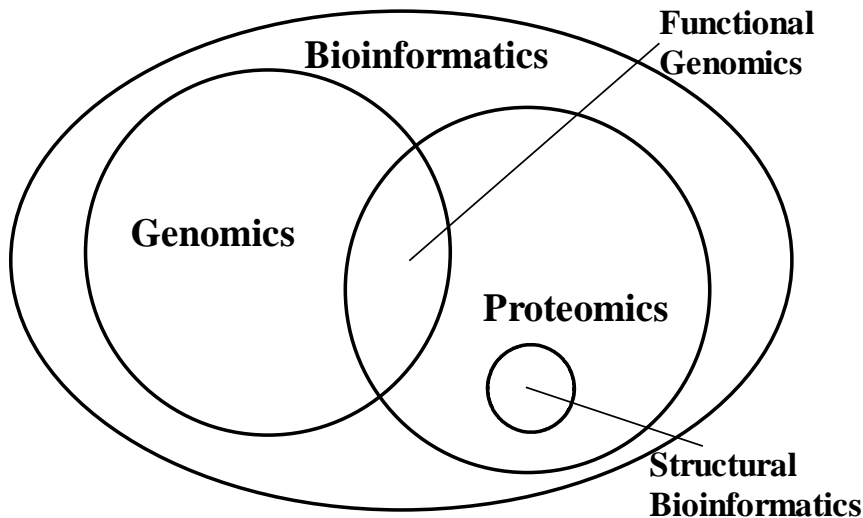
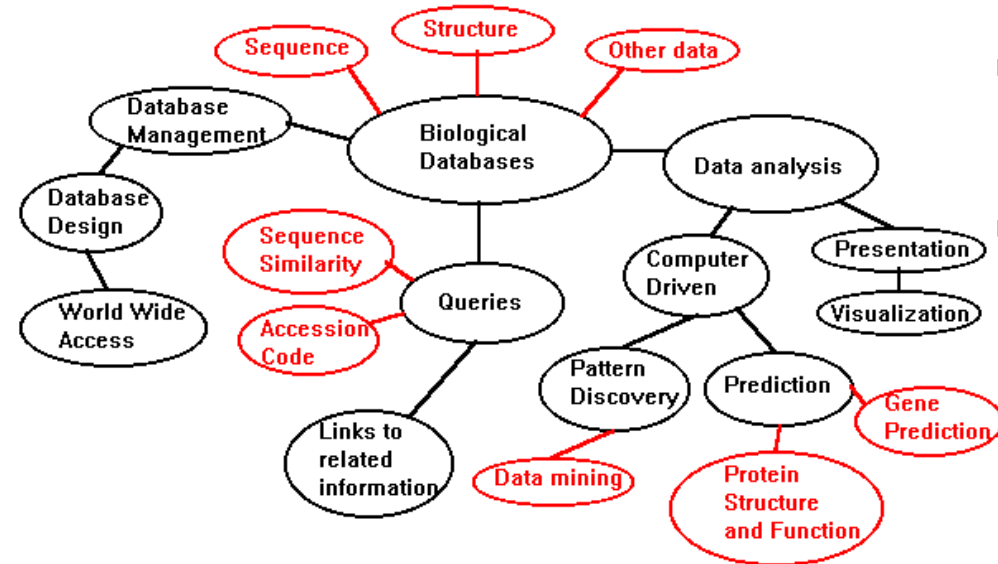
Gene Expression and Microarray



Biological Data Available

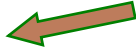
- Vast majority of data are *sequence of symbols* (*nucleotides*—*genomic data*, but also good amount on *amino acids*).
- Next in volume: *microarray* experiments and also *protein-array* data
- Comparably small: *3D structure of proteins* (PDB)
- *NCBI* (National Center for Biotechnology Information) server:
 - Total 26B bp: 3B bp human genome, then several bacteria (e.g., E. Coli), higher organisms: yeast, worm, fruitfly, mouse, and plants
 - The largest known genes has ~20million bp and the largest protein consists of ~34k amino acids
 - PDB has a catalogue of only 45k proteins, specified by their 3D structure (i.e, need to infer protein shape from sequence data)

Bioinformatics



- Computational management and analysis of biological information
- Interdisciplinary Field (Molecular Biology, Statistics, Computer Science, Genomics, Genetics, Databases, Chemistry, Radiology ...)
- Bioinformatics vs. *computational biology* (more on algorithm correctness, complexity and other themes central to theoretical CS)

Mining Sequence Patterns in Biological Data

- A brief introduction to biology and bioinformatics
- Alignment of biological sequences 
- Hidden Markov model for biological sequence analysis
- Summary

Comparing Sequences

- All living organisms are related to evolution
- **Alignment**: Lining up sequences to achieve the maximal level of identity
- Two sequences are **homologous** if they share a common ancestor
- Sequences to be compared: either nucleotides (DNA/RNA) or amino acids (proteins)
 - Nucleotides: identical
 - Amino acids: identical, or if one can be derived from the other by substitutions that are likely to occur in nature
- **Local vs. global alignments**: Local—only portions of the sequences are aligned. Global—align over the entire length of the sequences
 - Use gap “-” to indicate preferable not to align two symbols
- **Percent identity**: ratio between the number of columns containing identical symbols vs. the number of symbols in the longest sequence
- **Score** of alignment: summing up the matches and counting gaps as negative

Sequence Alignment: Problem Definition

- Goal:
 - Given two or more input sequences
 - Identify similar sequences with long conserved subsequences
- Method:
 - Use substitution matrices (probabilities of substitutions of nucleotides or amino-acids and probabilities of insertions and deletions)
 - *Optimal* alignment problem: NP-hard
 - Heuristic method to find *good* alignments

Pair-Wise Sequence Alignment

- Example

HEAGAWGHEE
PAWHEAE

```
HEAGAWGHE-E
 |   |  ||  |
P-A--W-HEAE
```

```
HEAGAWGHE-E
      ||  ||  |
--P-AW-HEAE
```

- Which one is better? → [Scoring alignments](#)
- To compare two sequence alignments, calculate a score
 - PAM (Percent Accepted Mutation) or BLOSUM (Blocks Substitution Matrix) (*substitution*) matrices: Calculate matches and mismatches, considering amino acid substitution
 - Gap penalty: Initiating a gap
 - Gap extension penalty: Extending a gap

Pair-Wise Sequence Alignment: Scoring Matrix

	A	E	G	H	W
A	5	-1	0	-2	-3
E	-1	6	-3	0	-3
H	-2	0	-2	10	-3
P	-1	-1	-2	-2	-4
W	-3	-3	-3	-3	15

- Gap penalty: -8
- Gap extension: -8

```

HEAGAWGHE-E
      ||  ||  |
--P-AW-HEAE
  
```

$$(-8) + (-8) + (-1) + 5 + 15 + (-8) + 10 + 6 + (-8) + 6 = 9$$

Exercise: Calculate for

```

HEAGAWGHE-E
      |  |  ||  |
P-A--W-HEAE
  
```

Formal Description

- *Problem:* **PairSeqAlign**

- *Input:* Two sequences x, y
Scoring matrix s
Gap penalty d
Gap extension penalty e

- *Output:* The optimal sequence alignment

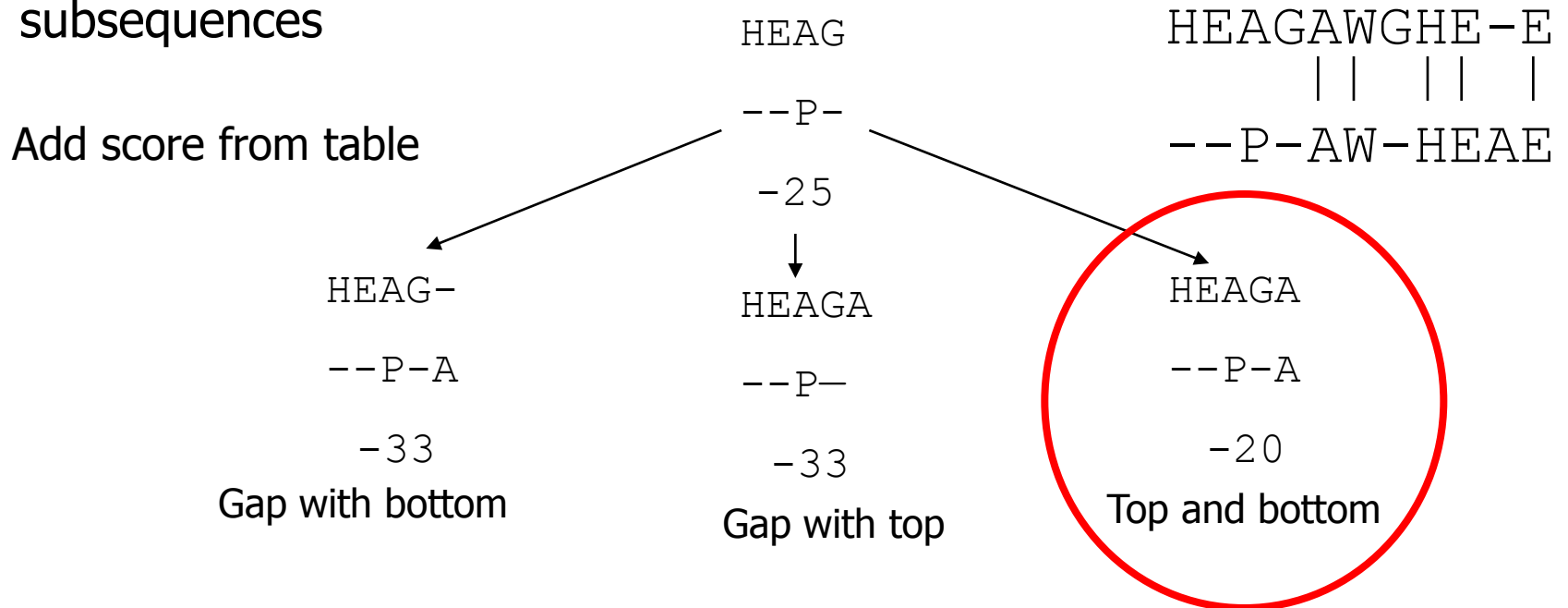
- *Difficulty:*

If x, y are of size n then
the number of possible
global alignments is

$$\rightarrow \binom{2n}{n} = \frac{(2n)!}{(n!)^2} \approx \frac{2^{2n}}{\sqrt{\pi n}}$$

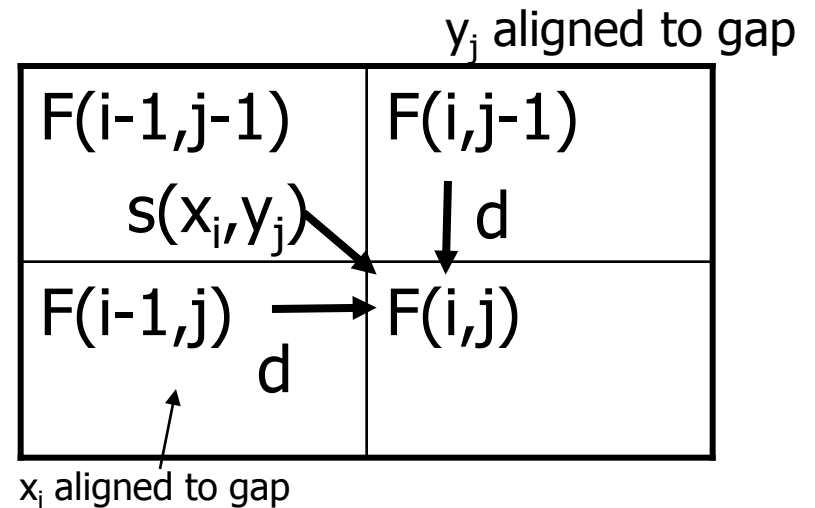
Global Alignment: Needleman-Wunsch

- Needleman-Wunsch Algorithm (1970)
 - Uses weights for the outmost edges that encourage the best overall (global) alignment
 - An alternative algorithm: Smith-Waterman (favors the contiguity of segments being aligned)
- Idea: Build up optimal alignment from optimal alignments of subsequences



Global Alignment

- Uses recursion to fill in intermediate results table
- Uses $O(nm)$ space and time
 - $O(n^2)$ algorithm
 - Feasible for moderate sized sequences, but not for aligning whole genomes.



While building the table, keep track of where optimal score came from, reverse arrows

Pair-Wise Sequence Alignment

Given $s(x_i, y_j)$, d

$$F(0,0) = 0$$

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

Alignment: $F(0,0) - F(n,m)$

Given $s(x_i, y_j)$, d

$$F(0,0) = 0$$

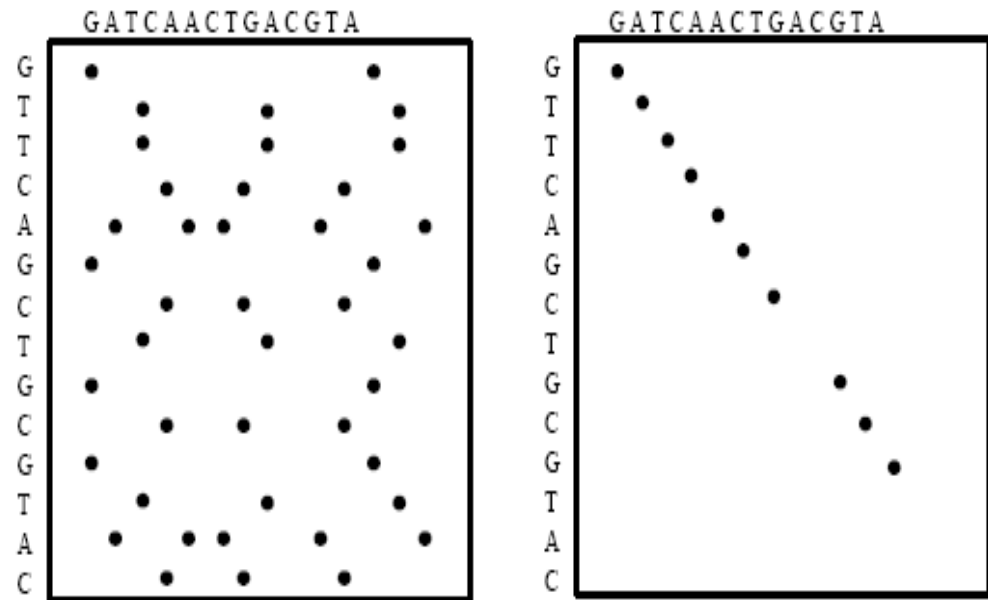
$$F(i, j) = \max \begin{cases} 0 \\ F(i-1, j-1) + s(x_i, y_j) \\ F(i-1, j) - d \\ F(i, j-1) - d \end{cases}$$

Alignment: $0 - F(i,j)$

We can vary both the model and the alignment strategies

Dot Matrix Alignment Method

- Dot Matrix Plot: Boolean matrices representing possible alignments that can be detected visually
 - Extremely simple but
 - $O(n^2)$ in time and space
 - Visual inspection



Heuristic Alignment Algorithms

- Motivation: Complexity of alignment algorithms: $O(nm)$
 - Current protein DB: 100 million base pairs
 - Matching each sequence with a 1,000 base pair query takes about 3 hours!
- Heuristic algorithms aim at speeding up at the price of possibly missing the best scoring alignment
- Two well known programs
 - BLAST: Basic Local Alignment Search Tool
 - FASTA: Fast Alignment Tool
 - Both find high scoring local alignments between a query sequence and a target database
 - Basic idea: first locate high-scoring short stretches and then extend them

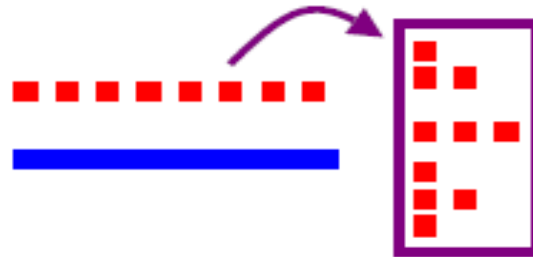
FASTA (Fast Alignment)

- **Approach** [Pearson & Lipman 1988]
 - Derived from the logic of the dot matrix method
 - View sequences as sequences of short words (k-tuple)
 - DNA: 6 bases, protein: 1 or 2 amino acids
 - Start from nearby sequences of exact matching words
- **Motivation**
 - Good alignments should contain many exact matches
 - Hashing can find exact matches in $O(n)$ time
 - Diagonals can be formed from exact matches quickly
 - Sort matches by position ($i - j$)
- Look only at matches near the longest diagonals
- Apply more precise alignment to small search space at the end

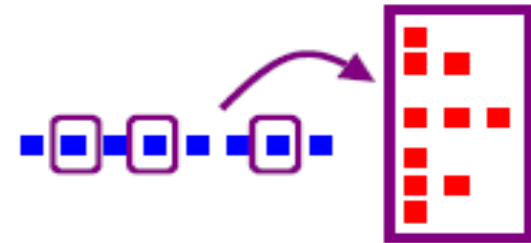
FASTA (Fast Alignment)



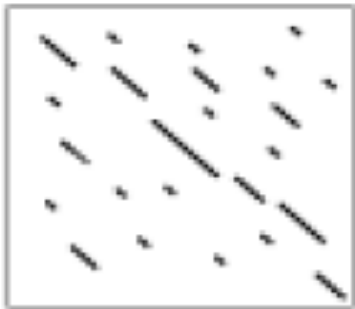
Input: two sequences



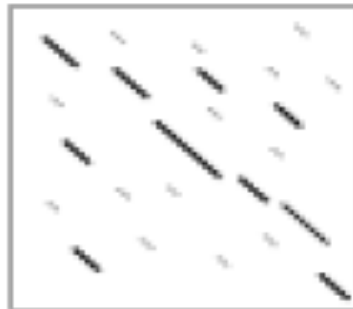
1) Convert 1st sequence into hash table of words



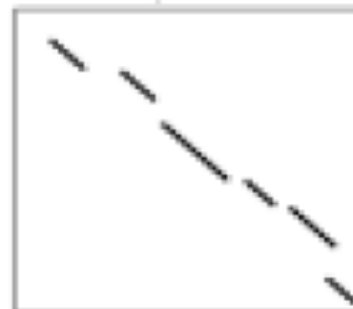
2) Scan 2nd sequence, find matching words in hash table, store locations



3) Convert matches to diagonals, discard rest



4) Re-score using scoring matrix



5) Try to join best (highest scoring) diagonals by adding gaps

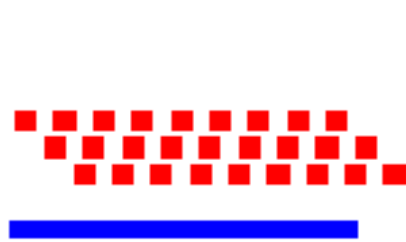


6) Use dynamic programming to align best diagonals

BLAST (Basic Local Alignment Search Tool)

- **Approach** (BLAST) (Altschul et al. 1990, developed by NCBI)
 - View sequences as sequences of short words (k -tuple)
 - DNA: 11 bases, protein: 3 amino acids
 - Create hash table of neighborhood (closely-matching) words
 - Use statistics to set threshold for “closeness”
 - Start from exact matches to neighborhood words
- **Motivation**
 - Good alignments should contain many close matches
 - Statistics can determine which matches are significant
 - Much more sensitive than % identity
 - Hashing can find matches in $O(n)$ time
 - Extending matches in both directions finds alignment
 - Yields high-scoring/maximum segment pairs (HSP/MSP)

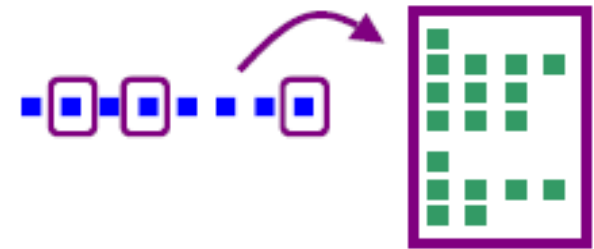
BLAST (Basic Local Alignment Search Tool)



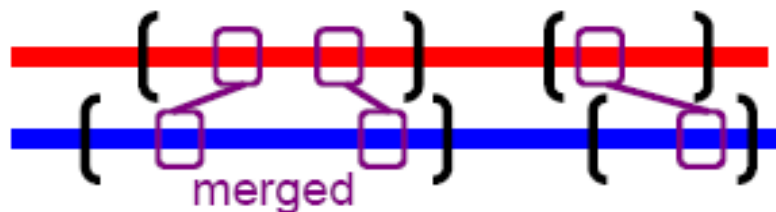
1) Convert 1st sequence into words (using all frames for given word size)



2) Calculate for each word list of “neighborhood” words (scoring threshold T) and enter in dictionary



3) Scan 2nd sequence, find matching words in dictionary, store locations



4) For each match, extend alignment in both directions while score above threshold S , merge segments



5) Align best segments using dynamic programming, report statistically significant matches

Multiple Sequence Alignment

```

**      :   *** **   :.:. : ** :**:* ** :* ** * * . : ... :*****: : *****:
gi|19923711|ref|NP_203523.2| MERL---ESELIRQSWRAVSRSPLEHGTVLFSRLFALEPSELLPLFQYNGRQFSSPEDCLSSPEFLDHIRKVMLVIDAAVT 77
gi|12584951|gb|AAG59898.1| MERP---ESELIRQSWRAVSRSPLEHGTVLFSRLFALEPSELLPLFQYNGRQFSSPEDCLSSPEFLDHIRKVMLVIDAAVT 77
gi|11967939|ref|NP_071859.1| MERP---ESELIRQSWRVVSRSPLEHGTVLFSRLFALEPSELLPLFQYNGRQFSSPEDCLSSPEFLDHIRKVMLVIDAAVT 77
gi|10864065|ref|NP_067080.1| MERP---EPELIRQSWRAVSRSPLEHGTVLFSRLFALEPDLPLFQYNGRQFSSPEDCLSSPEFLDHIRKVMLVIDAAVT 77
gi|15387696|emb|CAC59975.1| MEKLSKDKELIRGSNDSLGKKNKVPHGVI LFSRLFELDPELLNLFHYT-TNCGSTQDCLSSPEFLEHVTKVMLVIDAAVS 79
gi|15387694|emb|CAC59974.1| MEKLSKDKELIRGSNDSLGKKNKVPHGVI LFSRLFELDPELLNLFHYT-TNCGSTQDCLSSPEFLEHVTKVMLVIDAAVS 79
gi|18859087|ref|NP_571928.1| MEKLSKDKGLIRDSNESLGNKVPHGVI LFSRLFELDPELLNLFHYT-TNCGDAPECLSSPEFLEHVTKVMLVIDAAVS 79
ruler 1.....10.....20.....30.....40.....50.....60.....70.....80

```



```

.:** :**:* *****:****. ** :*****: ** :* : * ** :* ** :*:*** **
gi|19923711|ref|NP_203523.2| NVEDLSSLEEYLATLGRKHRAVGVR LSSFSTVGESLLYMLEKCLGPDFTPATRTANSQLYGAVVQAMSRGND--GE---- 151
gi|12584951|gb|AAG59898.1| NVEDLSSLEEYLATLGRKHRAVGVR LSSFSTVGESLLYMLEKCLGPDFTPATRTANSQLYGAVVQAMSRGND--GE---- 151
gi|11967939|ref|NP_071859.1| NVEDLSSLEEYLATLGRKHRAVGVR LSSFSTVGESLLYMLEKCLGPDFTPATRTANSQLYGAVVQAMSRGND--GE---- 151
gi|10864065|ref|NP_067080.1| NVEDLSSLEEYLATLGRKHRAVGVR LSSFSTVGESLLYMLEKCLGPDFTPATRTANSQLYGAVVQAMSRGND--GE---- 151
gi|15387696|emb|CAC59975.1| HLDDLHSLDFLLNLGRKHQAVGVK PQSFAVGESLLYMLQCSLQAYTASLRQANLNMYSVVVASM SRGNKNGEDKAD 159
gi|15387694|emb|CAC59974.1| HLDDLHSLDFLLNLGRKHQAVGVK PQSFAVGESLLYMLQCSLQAYTASLRQANLNMYSVVVASM SRGNKNGEDKAD 159
gi|18859087|ref|NP_571928.1| HLDDLHTLEDLLNLGRKHQAVGVN TQSFALVGESLLYMLQSSLGPAYTTS LRQANLTMYSIVVSAMTRGNKNGEHKSN 159
ruler .....90.....100.....110.....120.....130.....140.....150.....160

```



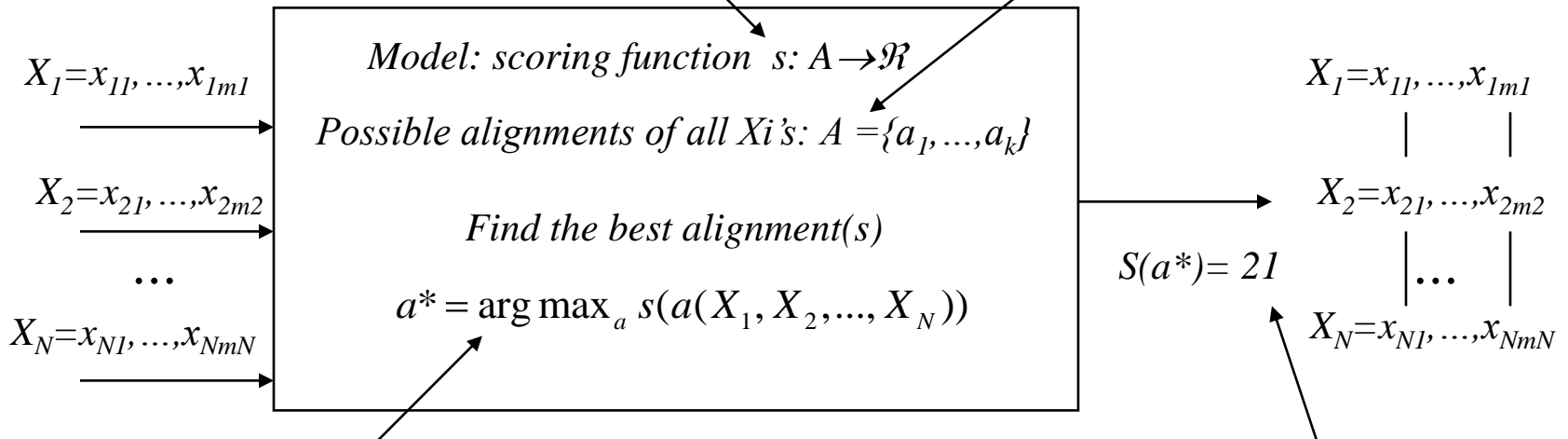
Multiple Sequence Alignment: Why?

- Identify highly conserved residues
 - Likely to be essential sites for structure/function
 - More precision from multiple sequences
 - Better structure/function prediction, pairwise alignments
- Building gene/protein families
 - Use conserved regions to guide search
- Basis for phylogenetic analysis
 - Infer evolutionary relationships between genes
- Develop primers & probes
 - Use conserved region to develop
 - Primers for PCR
 - Probes for DNA micro-arrays

Multiple Alignment Model

Q1: How should we define s ?

Q2: How should we define A ?



Q3: How can we find a^* quickly?

Q4: Is the alignment biologically Meaningful?

Minimum Entropy Scoring

- Intuition:
 - A perfectly aligned column has one single symbol (least uncertainty)
 - A poorly aligned column has many distinct symbols (high uncertainty)

$$S(m_i) = - \sum_a p_{ia} \log p_{ia}$$

$$p_{ia} = \frac{c_{ia}}{\sum_{a'} c_{ia'}}$$

Count of symbol a in column i

Multidimensional Dynamic Programming

Assumptions: (1) columns are independent (2) linear gap cost

$$S(m) = G + \sum_i s(m_i)$$

$$G = \gamma(g) = dg$$

$\alpha_{i_1, i_2, \dots, i_N}$

= Maximum score of an alignment up to the subsequences ending with

$x_{i_1}^1, x_{i_2}^2, \dots, x_{i_N}^N$

$\alpha_{0,0,\dots,0} = 0$

$$\alpha_{i_1, i_2, \dots, i_N} = \max \left\{ \begin{array}{l} \alpha_{i_1-1, i_2-1, \dots, i_N-1} + S(x_{i_1}^1, x_{i_2}^2, \dots, x_{i_N}^N) \\ \alpha_{i_1, i_2-1, \dots, i_N-1} + S(-, x_{i_2}^2, \dots, x_{i_N}^N) \\ \alpha_{i_1-1, i_2, \dots, i_N-1} + S(x_{i_1}^1, -, \dots, x_{i_N}^N) \\ \dots \\ \alpha_{i_1, i_2, \dots, i_N-1} + S(-, -, \dots, x_{i_N}^N) \\ \dots \\ \alpha_{i_1-1, i_2, \dots, i_N} + S(x_{i_1}^1, -, \dots, -) \end{array} \right.$$

Alignment: $0, 0, 0, \dots, 0 \text{---} |x^1|, \dots, |x^N|$

We can vary both the model and the alignment strategies

Complexity of Dynamic Programming

- Complexity: Space: $O(LN)$; Time: $O(2NLN)$
- One idea for improving the efficiency
 - Define the score as the sum of pairwise alignment scores

$$S(a) = \sum_{k < l} S(a^{kl})$$

Pairwise alignment between sequences k and l

- Derive a lower bound for $S(a^{kl})$, only consider a pairwise alignment scoring better than the bound

$$\sigma(a) \leq S(a^{kl}) - S(\hat{a}^{kl}) + \sum_{k' < l'} S(\hat{a}^{k'l'})$$

$$S(a^{kl}) \geq \beta^{kl}$$

$$\beta^{kl} = \sigma(a) + S(\hat{a}^{kl}) - \sum_{k' < l'} S(\hat{a}^{k'l'})$$

Approximate Algorithms for Multiple Alignment

- Two major methods (but it remains a worthy research topic)
 - Reduce a multiple alignment to a series of pairwise alignments and then combine the result (e.g., [Feng-Doolittle alignment](#))
 - Using HMMs ([Hidden Markov Models](#))
- [Feng-Doolittle alignment](#) (4 steps)
 - Compute all possible pairwise alignments
 - Convert alignment scores to distances
 - Construct a “guide tree” by clustering
 - Progressive alignment based on the guide tree (bottom up)
- Practical aspects of alignments
 - Visual inspection is crucial
 - Variety of input/output formats: need translation

More on Feng-Doolittle Alignment

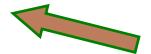
- Problems of Feng-Doolittle alignment
 - All alignments are completely determined by pairwise alignment (restricted search space)
 - No backtracking (subalignment is “frozen”)
 - No way to correct an early mistake
 - Non-optimality: Mismatches and gaps at highly conserved region should be penalized more, but we can’t tell where is a highly conserved region early in the process
- Iterative Refinement
 - Re-assigning a sequence to a different cluster/profile
 - Repeatedly do this for a fixed number of times or until the score converges
 - Essentially enlarge the search space

Clustal W: A Multiple Alignment Tool

- CLUSTAL and its variants are software packages often used to produce multiple alignments
- Essentially following Feng-Doolittle
 - Do pairwise alignment (dynamic programming)
 - Do score conversion/normalization (Kimura's model)
 - Construct a guide tree (neighbour-joining clustering)
 - Progressively align all sequences using profile alignment
- Offer capabilities of using substitution matrices like BLOSUM or PAM
- Many Heuristics

Mining Sequence Patterns in Biological Data

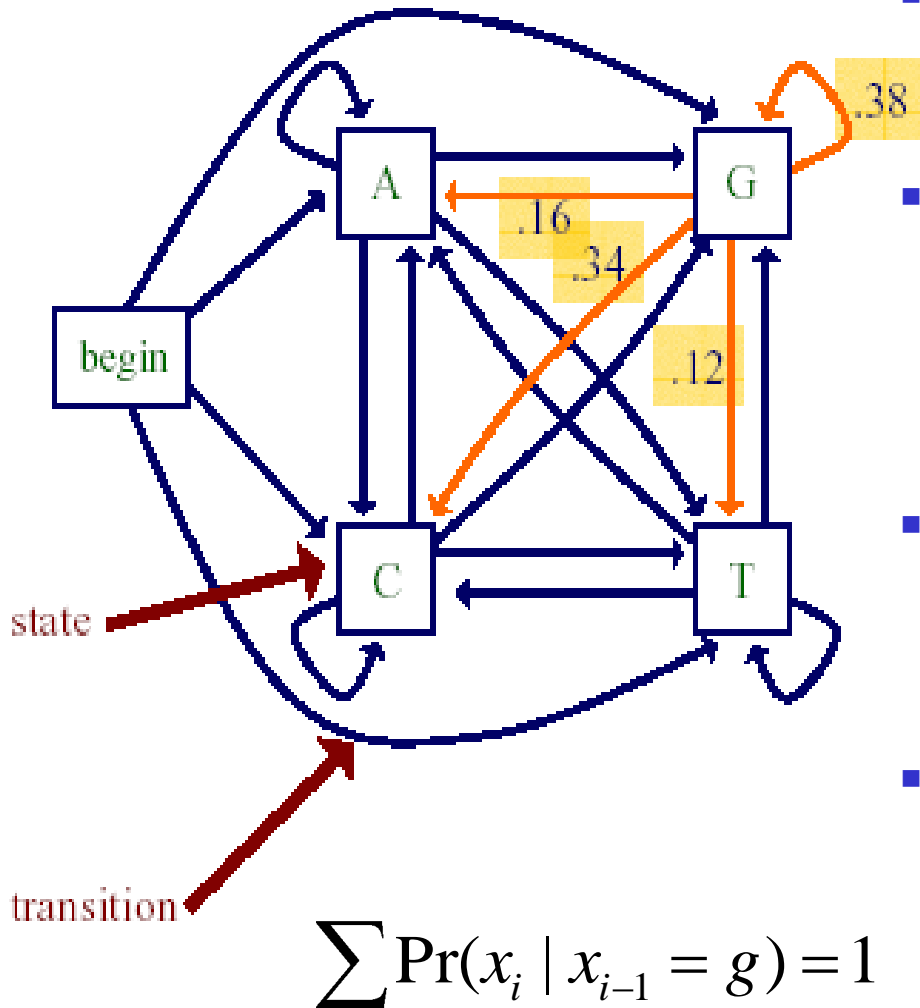
- A brief introduction to biology and bioinformatics
- Alignment of biological sequences
- Hidden Markov model for biological sequence analysis
- Summary



Markov Models in Computational Biology

- There are many cases in which we would like to represent the statistical regularities of some class of sequences
 - genes
 - various regulatory sites in DNA (e.g., where RNA polymerase and transcription factors bind)
 - proteins in a given family
- Markov models are well suited to this type of task

A Markov Chain Model



- **Markov property:** *Given the present state, future states are independent of the past states*
- At each step the system may change its state from the current state to another state, or remain in the same state, according to a certain probability distribution
- The changes of state are called ***transitions***, and the probabilities associated with various state-changes are called ***transition probabilities***
- Transition probabilities
 - $\Pr(x_i = a | x_{i-1} = g) = 0.16$
 - $\Pr(x_i = c | x_{i-1} = g) = 0.34$
 - $\Pr(x_i = g | x_{i-1} = g) = 0.38$
 - $\Pr(x_i = t | x_{i-1} = g) = 0.12$

Definition of Markov Chain Model

- A Markov chain model is defined by
 - A set of states
 - Some states emit symbols
 - Other states (e.g., the begin state) are silent
 - A set of transitions with associated probabilities
 - The transitions emanating from a given state define a distribution over the possible next states

Markov Chain Models: Properties

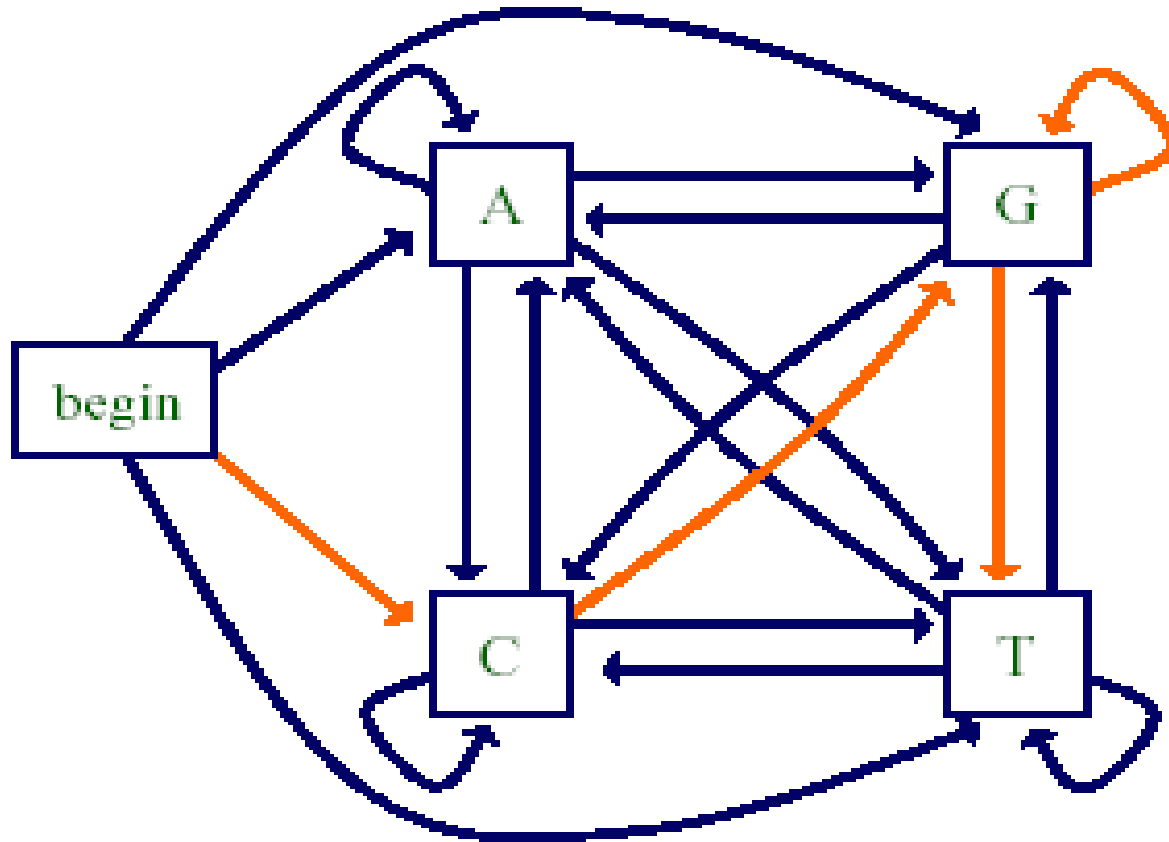
- Given some sequence x of length L , we can ask how probable the sequence is given our model
- For any probabilistic model of sequences, we can write this probability as

$$\begin{aligned}\Pr(x) &= \Pr(x_L, x_{L-1}, \dots, x_1) \\ &= \Pr(x_L / x_{L-1}, \dots, x_1) \Pr(x_{L-1} | x_{L-2}, \dots, x_1) \dots \Pr(x_1)\end{aligned}$$

- key property of a (1st order) Markov chain: the probability of each x_i depends only on the value of x_{i-1}

$$\begin{aligned}\Pr(x) &= \Pr(x_L / x_{L-1}) \Pr(x_{L-1} | x_{L-2}) \dots \Pr(x_2 | x_1) \Pr(x_1) \\ &= \Pr(x_1) \prod_{i=2}^L \Pr(x_i | x_{i-1})\end{aligned}$$

The Probability of a Sequence for a Markov Chain Model



$$\Pr(\text{cggt}) = \Pr(\text{c})\Pr(\text{g}|\text{c})\Pr(\text{g}|\text{g})\Pr(\text{t}|\text{g})$$

Example Application

- CpG islands
 - CG dinucleotides are rarer in eukaryotic genomes than expected given the marginal probabilities of C and G
 - but the regions upstream of genes are richer in CG dinucleotides than elsewhere – CpG islands
 - useful evidence for finding genes
- Application: Predict CpG islands with Markov chains
 - one to represent CpG islands
 - one to represent the rest of the genome

Markov Chains for Discrimination

- Suppose we want to distinguish CpG islands from other sequence regions
- Given sequences from CpG islands, and sequences from other regions, we can construct
 - a model to represent CpG islands
 - a null model to represent the other regions
- can then score a test sequence by:

$$score(x) = \log \frac{\Pr(x | CpGModel)}{\Pr(x | nullModel)}$$

Markov Chains for Discrimination

- Why use

$$score(x) = \log \frac{\Pr(x | CpGModel)}{\Pr(x | nullModel)}$$

- According to Bayes' rule

$$\Pr(CpG | x) = \frac{\Pr(x | CpG) \Pr(CpG)}{\Pr(x)}$$

$$\Pr(null | x) = \frac{\Pr(x | null) \Pr(null)}{\Pr(x)}$$

- If we are not taking into account of prior probabilities of two classes, we just need to compare $\Pr(x|CpG)$ and $\Pr(x|null)$

Higher Order Markov Chains

- The Markov property specifies that the probability of a state depends only on the probability of the previous state
- But we can build more “memory” into our states by using a higher order Markov model
- In an n-th order Markov model

$$\Pr(x_i \mid x_{i-1}, x_{i-2}, \dots, x_1) = \Pr(x_i \mid x_{i-1}, \dots, x_{i-n})$$

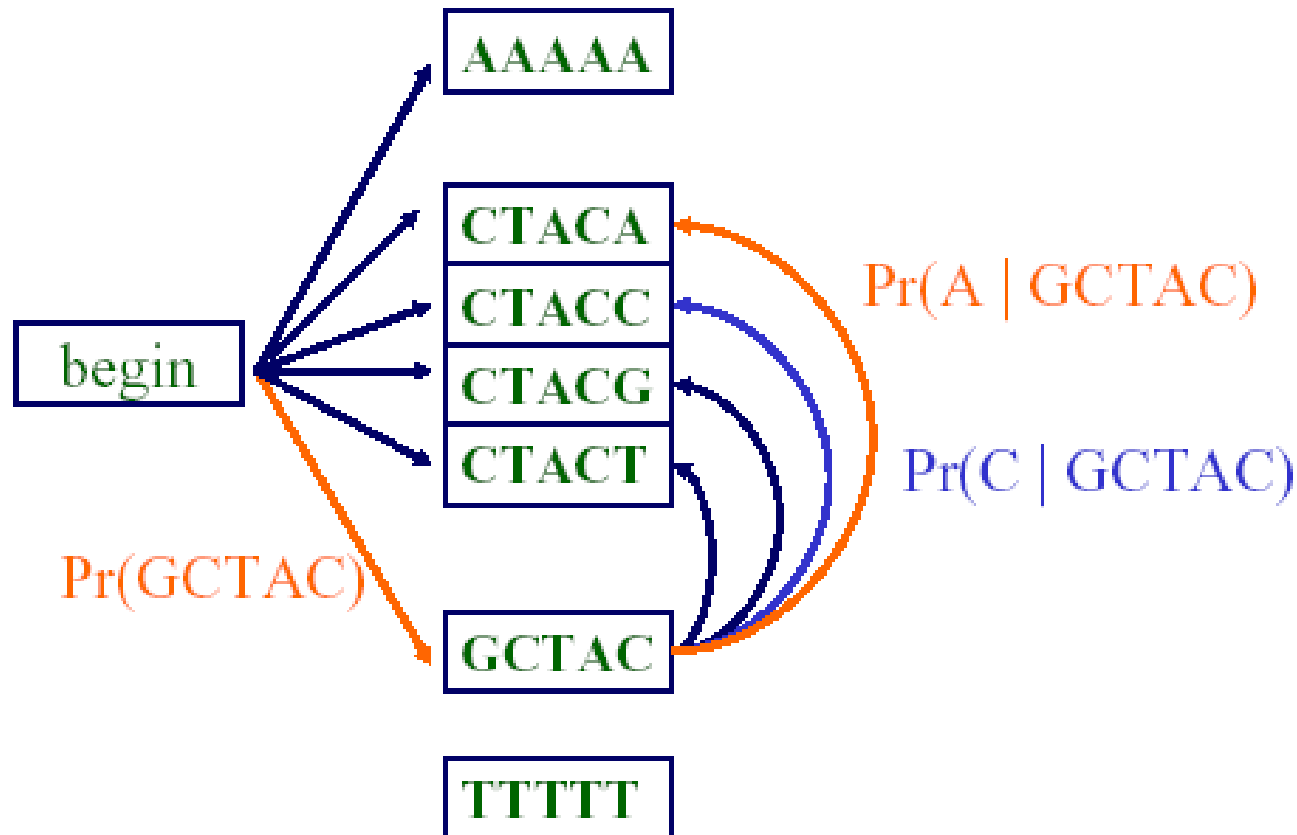
Selecting the Order of a Markov Chain Model

- The number of parameters we need to estimate grows exponentially with the order
 - for modeling DNA we need $O(4^{n+1})$ parameters for an n-th order model
- The higher the order, the less reliable we can expect our parameter estimates to be
 - estimating the parameters of a 2nd order Markov chain from the complete genome of E. Coli, we'd see each word $> 72,000$ times on average
 - estimating the parameters of an 8-th order chain, we'd see each word ~ 5 times on average

Higher Order Markov Chains

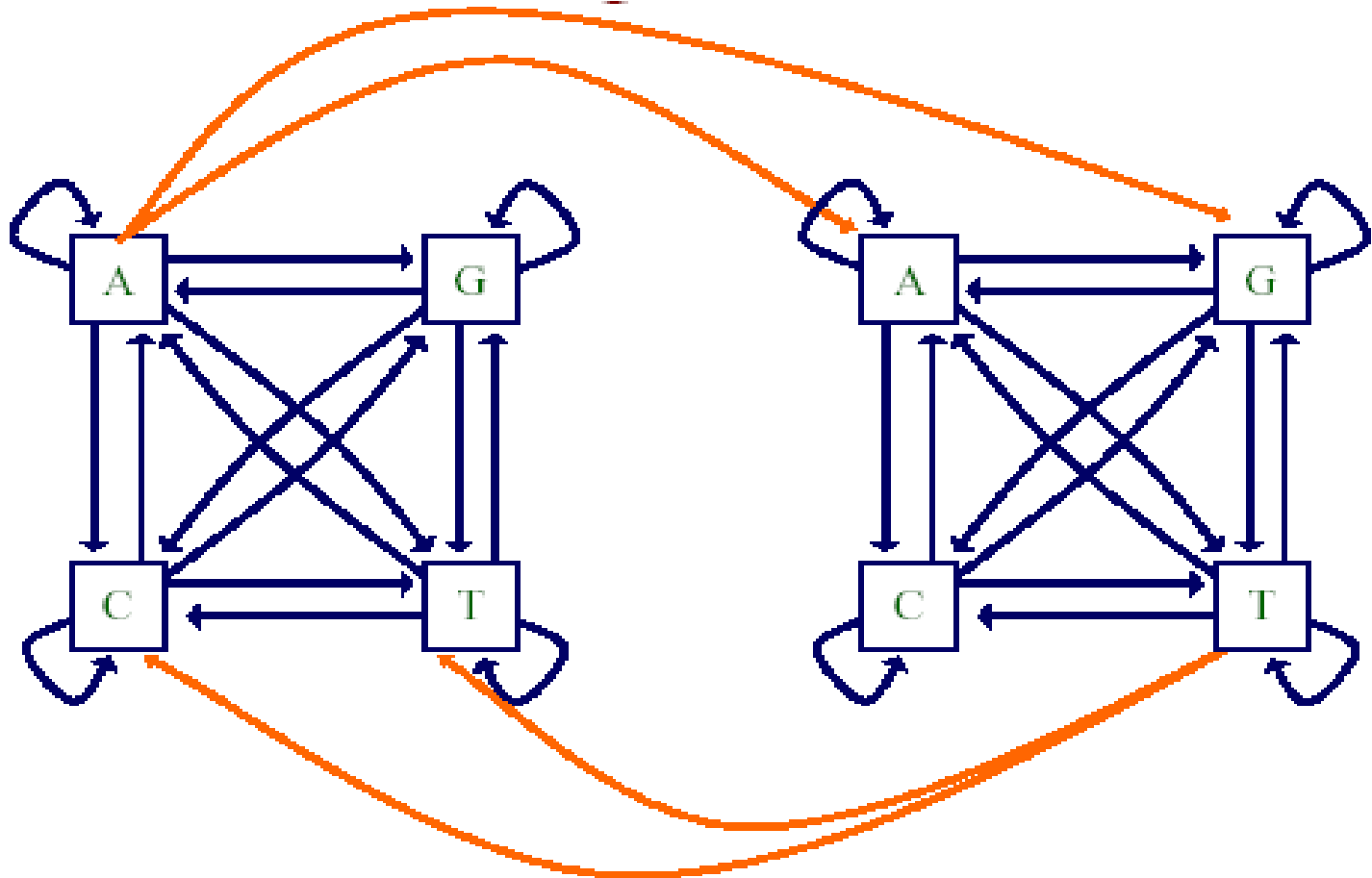
- An n-th order Markov chain over some alphabet A is equivalent to a first order Markov chain over the alphabet of n-tuples: A^n
- Example: A 2nd order Markov model for DNA can be treated as a 1st order Markov model over alphabet
AA, AC, AG, AT
CA, CC, CG, CT
GA, GC, GG, GT
TA, TC, TG, TT

A Fifth Order Markov Chain



$$\Pr(\text{gctaca}) = \Pr(\text{gctac})\Pr(\text{a} | \text{gctac})$$

Hidden Markov Model: A Simple HMM



Given observed sequence AGGCT, which state emits every item?

Hidden Markov Model

- A **hidden Markov model (HMM)**: A statistical model in which the system being modeled is assumed to be a Markov process with *unknown parameters*
- The challenge is to determine the hidden parameters from the observable data. The extracted model parameters can then be used to perform further analysis
- An HMM can be considered as the simplest dynamic Bayesian network
- In a *hidden* Markov model, the state is not directly visible, but variables influenced by the state are visible
- Each state has a probability distribution over the possible output tokens. Therefore the sequence of tokens generated by an HMM gives some information about the sequence of states.

Learning and Prediction Tasks

- Learning
 - **Given** a model, a set of training sequences
 - Find model parameters that explain the training sequences with relatively high probability (goal is to find a model that *generalizes* well to sequences we haven't seen before)
- Classification
 - **Given** a set of models representing different sequence classes, a test sequence
 - Determine which model/class best explains the sequence
- Segmentation
 - **Given** a model representing different sequence classes, a test sequence
 - Segment the sequence into subsequences, predicting the class of each subsequence

Algorithms for Learning & Prediction

- Learning
 - **correct path known for each training sequence** → simple maximum likelihood or Bayesian estimation
 - **correct path not known** → Forward-Backward algorithm + ML or Bayesian estimation
- Classification
 - **simple Markov model** → calculate probability of sequence along single path for each model
 - **hidden Markov model** → Forward algorithm to calculate probability of sequence along all paths for each model
- Segmentation
 - **hidden Markov model** → Viterbi algorithm to find most probable path for sequence

The Parameters of an HMM

- Transition Probabilities

$$a_{kl} = \Pr(\pi_i = l \mid \pi_{i-1} = k)$$

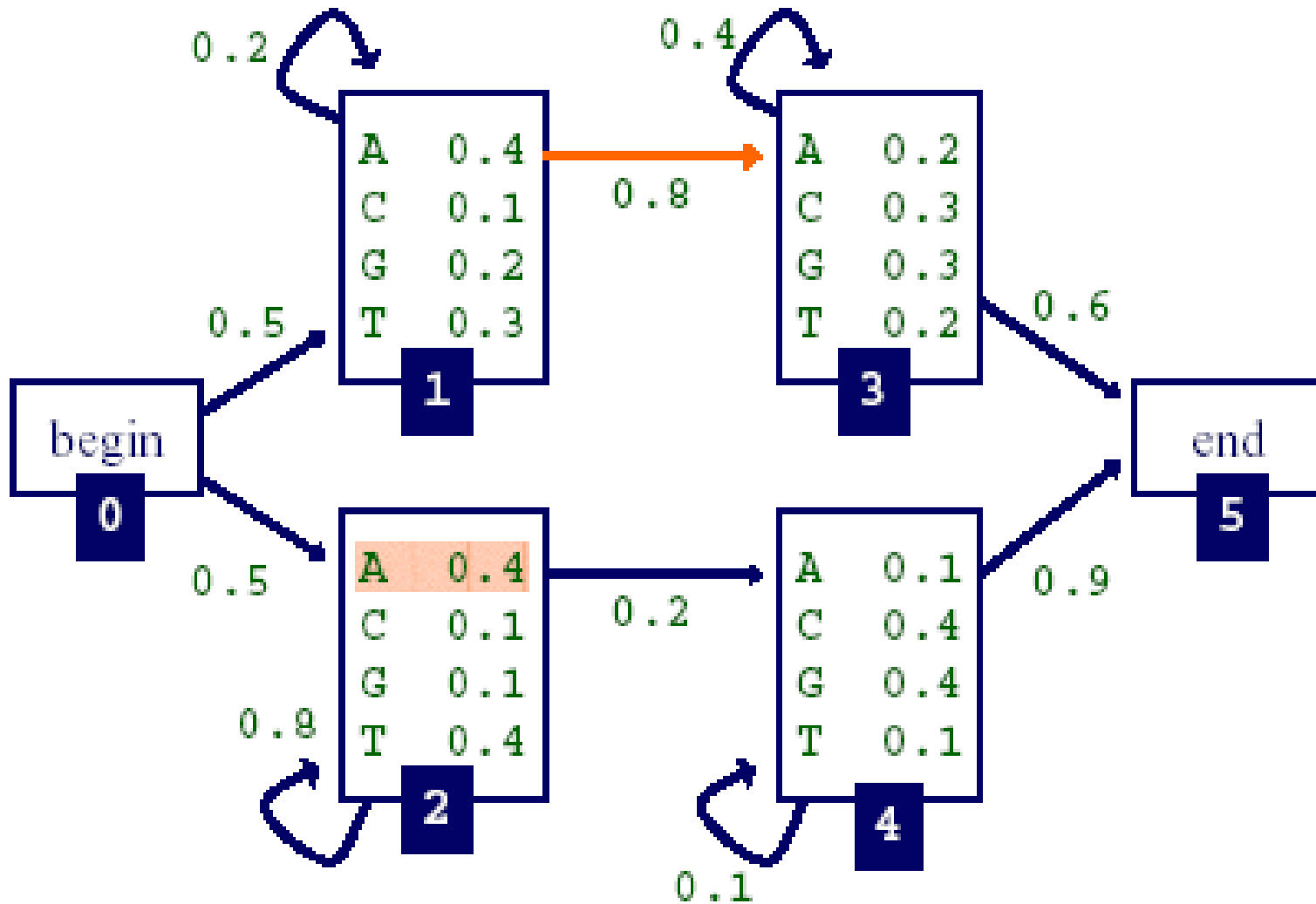
- Probability of transition from state k to state l

- Emission Probabilities

$$e_k(b) = \Pr(x_i = b \mid \pi_i = k)$$

- Probability of emitting character b in state k

An HMM Example



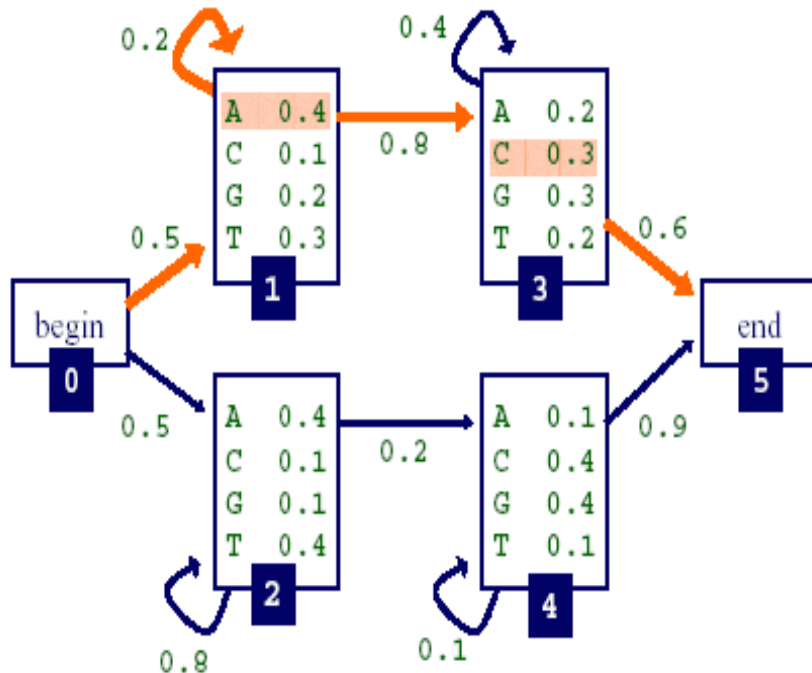
Three Important Questions

- How likely is a given sequence?
 - The Forward algorithm
- What is the most probable “path” for generating a given sequence?
 - The Viterbi algorithm
- How can we learn the HMM parameters given a set of sequences?
 - The Forward-Backward (Baum-Welch) algorithm

How Likely is a Given Sequence?

- The probability that the path is taken and the sequence is generated:

$$\Pr(x_1 \dots x_L, \pi_0 \dots \pi_N) = a_{0\pi_1} \prod_{i=1}^L e_{\pi_i}(x_i) a_{\pi_i \pi_{i+1}}$$



$$\begin{aligned} \Pr(AAC, \pi) &= a_{01} \times e_1(A) \times a_{11} \times e_1(A) \\ &\times a_{13} \times e_3(C) \times a_{35} \\ &= .5 \times .4 \times .2 \times .4 \times .8 \times .3 \times .6 \end{aligned}$$

How Likely is a Given Sequence?

- The probability over all paths is

$$\Pr(x_1 \dots x_L) = \sum_{\pi} \Pr(x_1 \dots x_L, \underbrace{\pi_0 \dots \pi_N}_{\pi})$$

- But the number of paths can be exponential in the length of the sequence...
- The Forward algorithm enables us to compute this efficiently
 - Define $f_k(i)$ to be the probability of being in state k having observed the first i characters of sequence x
 - To compute $f_N(L)$, the probability of being in the end state having observed all of sequence x
 - Can define this recursively
 - use dynamic programming

The Forward Algorithm

- Initialization

- $f_0(0) = 1$ for start state; $f_i(0) = 0$ for other state

- Recursion

- For emitting state ($i = 1, \dots, L$)

$$f_l(i) = e_l(i) \sum_k f_k(i-1) a_{kl}$$

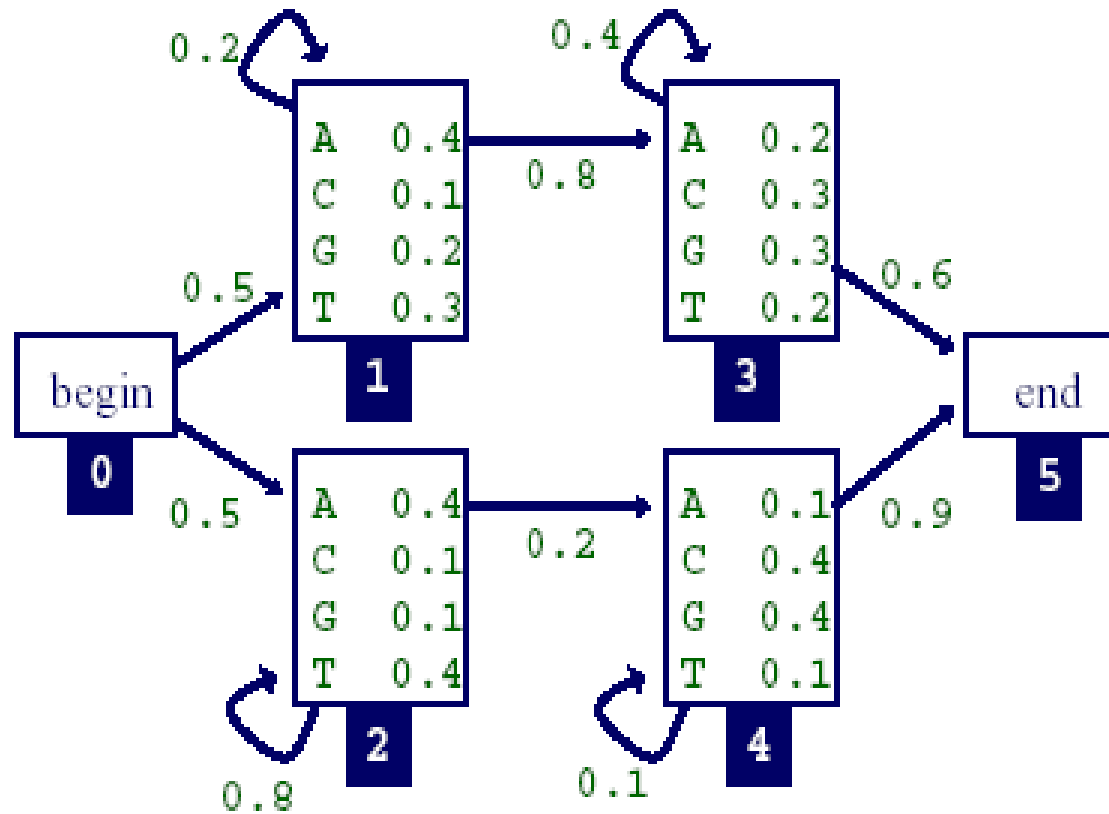
- For silent state

$$f_l(i) = \sum_k f_k(i) a_{kl}$$

- Termination

$$\Pr(x) = \Pr(x_1 \dots x_L) = f_N(L) = \sum_k f_k(L) a_{kN}$$

Forward Algorithm Example



Given the sequence $x=TAGA$

Forward Algorithm Example

- Initialization
 - $f_0(0)=1, f_1(0)=0 \dots f_5(0)=0$
- Computing other values
 - $f_1(1)=e_1(T)*(f_0(0)a_{01}+f_1(0)a_{11})$
 $=0.3*(1*0.5+0*0.2)=0.15$
 - $f_2(1)=0.4*(1*0.5+0*0.8)$
 - $f_1(2)=e_1(A)*(f_0(1)a_{01}+f_1(1)a_{11})$
 $=0.4*(0*0.5+0.15*0.2)$
 - ...
 - $\Pr(\text{TAGA})= f_5(4)=f_3(4)a_{35}+f_4(4)a_{45}$

Three Important Questions

- How likely is a given sequence?
- What is the most probable “path” for generating a given sequence?
- How can we learn the HMM parameters given a set of sequences?

Finding the Most Probable Path: The Viterbi Algorithm

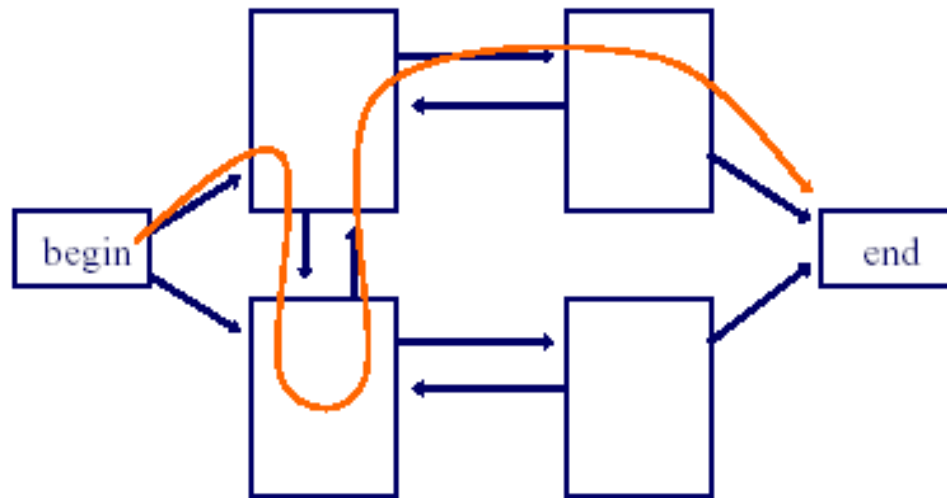
- Define $v_k(i)$ to be the probability of the most probable path accounting for the first i characters of x and ending in state k
- We want to compute $v_N(L)$, the probability of the most probable path accounting for all of the sequence and ending in the end state
- Can define recursively
- Can use DP to find $v_N(L)$ efficiently

Three Important Questions

- How likely is a given sequence?
- What is the most probable “path” for generating a given sequence?
- How can we learn the HMM parameters given a set of sequences?

Learning Without Hidden State

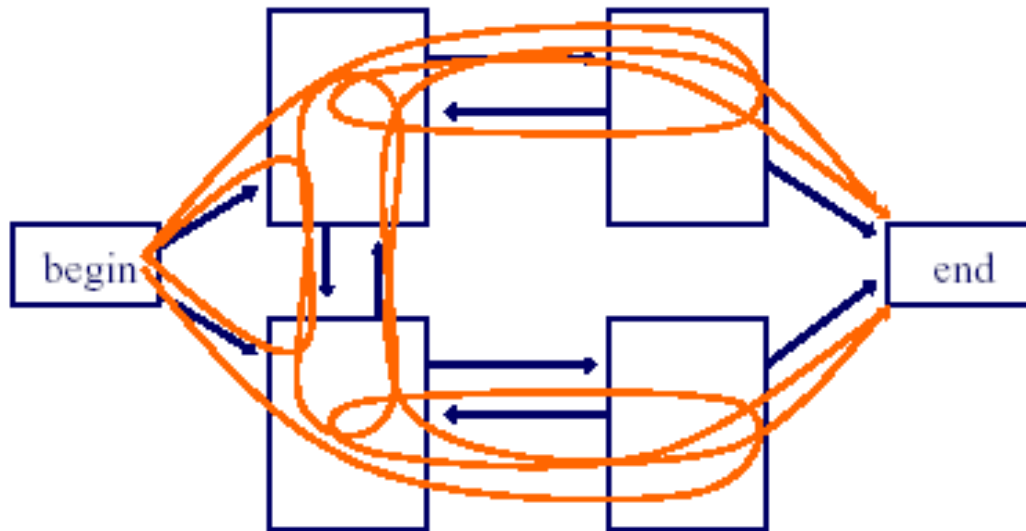
- Learning is simple if we know the correct path for each sequence in our training set



- estimate parameters by counting the number of times each parameter is used across the training set

Learning With Hidden State

- If we don't know the correct path for each sequence in our training set, consider all possible paths for the sequence



- Estimate parameters through a procedure that counts the expected number of times each parameter is used across the training set

Learning Parameters: The Baum-Welch Algorithm

- Also known as the Forward-Backward algorithm
- An Expectation Maximization (EM) algorithm
 - EM is a family of algorithms for learning probabilistic models in problems that involve hidden state
- In this context, the hidden state is the path that best explains each training sequence

Learning Parameters: The Baum-Welch Algorithm

- Algorithm sketch:
 - initialize parameters of model
 - iterate until convergence
 - calculate the *expected* number of times each transition or emission is used
 - adjust the parameters to *maximize* the likelihood of these expected values

Computational Complexity of HMM Algorithms

- Given an HMM with S states and a sequence of length L , the complexity of the Forward, Backward and Viterbi algorithms is

$$O(S^2L)$$

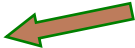
- This assumes that the states are densely interconnected
- Given M sequences of length L , the complexity of Baum Welch on each iteration is

$$O(MS^2L)$$

Markov Models Summary

- We considered models that vary in terms of order, hidden state
- Three Dynamic Programming-based algorithms for HMMs: **Forward, Backward and Viterbi**
- We discussed three key tasks: learning, classification and segmentation
- The algorithms used for each task depend on whether there is hidden state (correct path known) in the problem or not

Mining Sequence Patterns in Biological Data

- A brief introduction to biology and bioinformatics
- Alignment of biological sequences
- Hidden Markov model for biological sequence analysis
- Summary 

Summary: Mining Biological Data

- Biological sequence analysis compares, aligns, indexes, and analyzes biological sequences (sequence of nucleotides or amino acids)
- Biosequence analysis can be partitioned into two essential tasks:
 - pair-wise sequence alignment and multiple sequence alignment
- Dynamic programming approach (notably, BLAST) has been popularly used for sequence alignments
- Markov chains and hidden Markov models are probabilistic models in which the probability of a state depends only on that of the previous state
 - Given a sequence of symbols, x , the **forward** algorithm finds the probability of obtaining x in the model
 - The **Viterbi** algorithm finds the most probable path (corresponding to x) through the model
 - The **Baum-Welch** learns or adjusts the model parameters (transition and emission probabilities) to best explain a set of training sequences.

References

- Lecture notes@M. Craven's website: www.biostat.wisc.edu/~craven
- A. Baxevanis and B. F. F. Ouellette. *Bioinformatics: A Practical Guide to the Analysis of Genes and Proteins* (3rd ed.). John Wiley & Sons, 2004
- R. Durbin, S. Eddy, A. Krogh and G. Mitchison. *Biological Sequence Analysis: Probability Models of Proteins and Nucleic Acids*. Cambridge University Press, 1998
- N. C. Jones and P. A. Pevzner. *An Introduction to Bioinformatics Algorithms*. MIT Press, 2004
- I. Korf, M. Yandell, and J. Bedell. *BLAST*. O'Reilly, 2003
- L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proc. IEEE*, 77:257--286, 1989
- J. C. Setubal and J. Meidanis. *Introduction to Computational Molecular Biology*. PWS Pub Co., 1997.
- M. S. Waterman. *Introduction to Computational Biology: Maps, Sequences, and Genomes*. CRC Press, 1995