Text Algorithms
Lecture: PWM, HMM, SCFG, Transducers

Jaak Vilo

2008 fall

Probabilistic models and other extensions

- Position Weiht Matrices PWM
  - Position specific scoring matrices
- Hidden Markov models (HMM)
- Stochastic Context Free Grammars (SCFG)
- Finite state transducers (FST)
- Attributed automata

- It is not possible to always define the problem using strings, regular expressions, or approximate matching with few mismatches.
- Sometimes probabilistic models are better suited
- And correspond better to real world. For example, the affinity between a protein and the DNA is the physical property.
- Biological systems are not discrete digital computers but rather analog ones

- Durbin, R., Eddy, S., Krogh, A., Mitchison Biological sequence analysis **Probabilistic models of proteins and nucleic acids** (Cambridge University Press, 1998)
- Molecular Information Theory and the Theory of Molecular Machines (Tom Schneider) http://www.lecb.ncifcrf.gov/%7Etoms/
- G. Z. Hertz, Gary D. Stormo: Identifying DNA and protein patterns with statistically significant alignments of multiple sequences. Bioinformatics 15(7): 563577 (1999)

http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?cmd=Retrieve&db=PubMed&list_uids=10487864&dopt=Abstract

- A **position weight matrix (PWM)**, also called **position-specific weight matrix (PSWM)** or **position-specific scoring matrix (PSSM)**, is a commonly used representation of motifs (patterns) in biological sequences.

(Wikipedia)

- **See compression algorithms for Entropy and Information Content**
- Lets have 8 sequences with certain properties (e.g. a protein is known to bind to the respective DNA stretches)
- How to summarize that info? What is the true signal?
- $C_{i,j}$ nr of occurrences of character i in column j.

GATGAG
GATGGG
GCTGAG
GAAGAG
AATGAG
GATGAA
GTTGAG
GATGAT

- Counts: $C_{i,j}$

```
A  1 6 1 0 7 1
C  0 1 0 0 0 0
G  7 0 0 8 1 6
T  0 1 7 0 0 1

   G A T G A G (consensus?)
```

GATGAG
GATGGG
GCTGAG
GAAGAG
AATGAG
GATGAA
GTTGAG
GATGAT

Match

```
A G A T G A G T G
1 0 1 0 1 1          = 4
  7 6 7 8 7 6        = 41
    1 0 0 0 1 1      = 3
      0 0 1 8 0 6    = 15
```

```
A     1 6 1 0 7 1
C     0 1 0 0 0 0
G     7 0 0 8 1 6
T     0 1 7 0 0 1
```

GCTGAA = 7 1 7 8 7 1 = 31

```
A     1 6 1 0 7 1
C     0 1 0 0 0 0
G     7 0 0 8 1 6
T     0 1 7 0 0 1

      G A T G A G (consensus
```

Does consensus represent the information? No.

Consensus with one mismatch? no

But mismatch can occur at an unlikely position then.

- $p_{i,j}$ probability of letter i at positiopn j
- $p_{i,j} = f_{i,j} / N$ (N = nr of sequences)

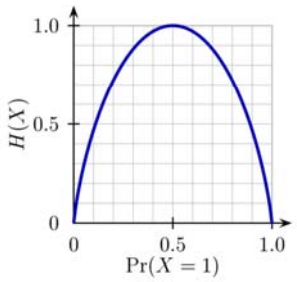
```
Probabilities

A 0.125  0.75   0.125  0   0.875   0.125
C 0      0.125  0      0   0       0
G 0.875  0      0      1   0.125   0.75
T 0      0.125  0.875  0   0       0.125
```

- Is 0 probability really 0?
- $p_{i,j} = (f_{i,j} + n_i) / (N+n)$ (n_i pseudocounts, sum n)
- Proportions are the same as in absolute numbers

Information content

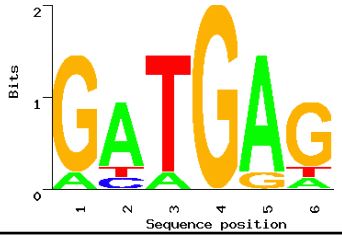
- At column i the sum of probabilities is 1: $\sum_{i=1..c} p_{i,j} = 1$
- Lets look at column j
- $\log_2 P_i$ Level of "surprise" in bits P_i
- If p is small, the more surprised we would be, the more information the position carries.
- Overall information content is counted over all possible characters

$$H = \sum_{i=1..c} p_i \log_2 p_i$$



A	0.375	0.311	0.375	0	0.169	0.375
C	0	0.375	0	0	0	0
G	0.169	0	0	1	0.375	0.311
T	0	0.375	0.169	0	0	0.375
H=	0.544	1.061	0.544	0	0.544	1.061

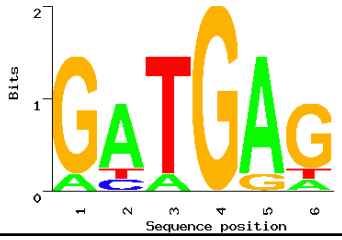
A	0.375	0.311	0.375	0	0.169	0.375
C	0	0.375	0	0	0	0
G	0.169	0	0	1	0.375	0.311
T	0	0.375	0.169	0	0	0.375
H=	0.544	1.061	0.544	0	0.544	1.061



- Logo: Column height is 2 H
- Logo: Letter height is the proportion of the column: p_i*(2H)
- Logo: Sort letters according to probability p_i*(2H)

PWM, PSSM

A	0.375	0.311	0.375	0	0.169	0.375
C	0	0.375	0	0	0	0
G	0.169	0	0	1	0.375	0.311
T	0	0.375	0.169	0	0	0.375
H=	0.544	1.061	0.544	0	0.544	1.061



GATGAG
GATGGG
GCTGAG
GAAGAG
AATGAG
GATGAA
GTTGAG
GATGAT

- PATMATCH http://www.bioinf.ebc.ee/EP/EP/
- Try: 1:…TGAAAATTTT….
- -1:….TG.GATGAG.T….

PDF

http://bilt.cs.ut.ee/~vilo/edu/2005-06/Text_Algorithms/L10_Probabilistic/Articles/Hertz_Stormo_Bioinformatics_1999_Significance.pdf

G.Z.Hertz and G.D.Stormo

a) Alignment Matrix b) Weight Matrix

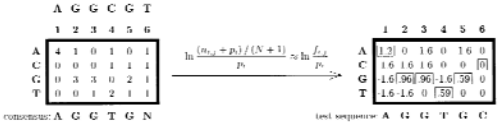


Fig. 1. Examples of the simple matrix model for summarizing a DNA alignment. (a) An alignment matrix describing the alignment of the four 6-mers on top. The matrix contains the number of times, $n_{i,j}$, that letter i is observed at position j of this alignment. Below the matrix is the consensus sequence corresponding to the alignment (N indicates that there is no nucleotide preference). (b) A weight matrix derived from the alignment in (a). The formula used for transforming the alignment matrix to a weight matrix is shown above the arrow. In this formula, N is the total number of sequences (four in this example), p_i is the *a priori* probability of letter i (0.25 for all the bases in this example) and $f_{i,j} = n_{i,j}/N$ is the frequency of letter i at position j. The numbers enclosed in blocks are summed to give the overall score of the test sequence. The overall score is 4.3, which is also the maximum possible score with this weight matrix.

- Example from http://www.biomedcentral.com/14712180/2/29
- Observed frequencies (counted from the data)
- Odds score (ratio of observed over background frequenceies)
- The odds score is the frequency observed divided by the theoretical frequency expected (i.e., the background frequency of the base, usually averaged over the genome ~0.25/base).
- In DNA a crude (and wrong) estimate is: $P_A=P_C=P_T=P_G=1/4$
- If f is 0.79 and bkg prob. is 0.25, is odds score 0.79/0.25 = 3.16.
- Finally, odds scores were converted to log odds scores by taking the logarithm base 2.
- $W_{i,j} = \log_2(F_{i,j}/P_i)$ $W_{i,j}$ is score for character i in column j.
- As the logarithm of zero is infinity, a zero occurrence of a particular base in the matrix creates a problem.
- A formula proposed by Hertz and Stormo can be used
- $W_{i,j} = \log_2 (C_{i,j} + p_i)/ (N + 1)P_i$
- (Which is approximately ≈ $\log_2(F_{i,j}/P_i)$)
- Where $C_{i,j}$ is nr of occurrences of i'th character in position j and N is the nr. of sequences

- The statistical significance of such alignment can be calculated
- $I_{seqalign} = \sum_i \sum_i f_{i,j} \log (f_{i,j} / p_{i,j})$
- The information content (aka normaliseeritud logodds ratio) Other names include KullbackLeibler information (Kullback, Leibler, 1951), relative entropy. Or, *largedeviation rate function* (Bucklew, 1990).
- Formula has properties which satisfy the intuition about the information content
- It is for example a distance measure from the center of the distribution with $f_{i,j} = p_{i,j}$.
- Distance is minimal when $f_{i,j} = p_{i,j}$
- Distance is maximal when the least expected character is exclusive, or $f_{m,j}=1$ ja $p_m \leq p_i$ for every i.
- Schneider et.al. (1986) noticed that $2^{I}_{seqalign}$ is approximately equal to frequency of the occurrences of DNA binding sites in *e.coli*
- Stormo etal: $2^{I}_{seqalign}$ is the upper limit for the expectation with whic the words from the alignment occur in the random sequence

- Log likelihood

Perceptron

- PWM is like a simple neural network, the 1node perceptron
- $F(X=x_1x_2..x_m) = \sum w_i x_i$
- DNA signals can also be represented in more complex neural networks (multilayer)

Matching of PWM

- Alphabet Σ and PWM ($W_{i,j}$) where for a character i at column j there is a weight $W_{i,j}$
- Like brute force exact match

Algorithm: Search with PWM
Input: PWM ($W_{i,j}$), i∈1..|Σ|, j∈1..m, string S, threshold K
Output: All occurrences of $W_{i,j}$ in S with quality at least K

```
for p=1 .. |S|m+1
    sum = 0 ;
    for j=1 .. m
        sum += W[ S[p+j1] ][ j ]
    if sum ≥ K then report match at position p with score sum
```

Matching of PWM

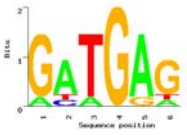
- Can it be improved when K is high? Stop when K cannot be reached
- Use some precalculated scores?
- Aho Corasick?

Some local research

- Marek Zäuram – BSc
- Jelena Zaitseva, Pavlos Pavlidis – PWM matching with
  - indels (e.g like edit distance – add/remove)
  - and closures (repeat the last column…)
- PATMATCH
- Still room for completing some of that research

Matches above threshold t

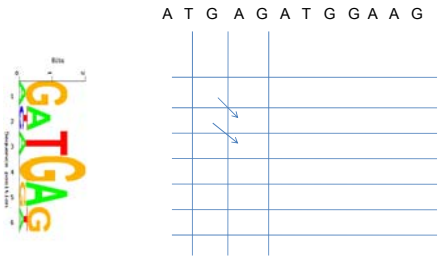
1. Match scores of potentially most significant positions
2. If remaining best possible scores can not achieve the threshold, stop



E.g. You can afford only Max- t "mistakes"

Matching v2

Shift(Prev_col) + vector[A]



- Motif combinations
- Indels, closures
- What is the "best signal" ?

Research is going on…

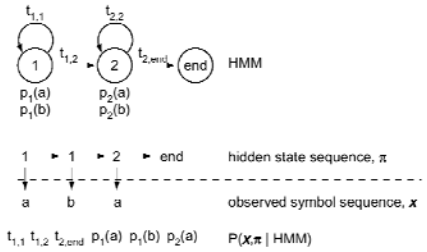
- Cinzia Pizzi, Esko Ukkonen: **Fast profile matching algorithms — A survey** *Theoretical Computer Science* Volume 395, Issues 23, 1 May 2008, Pages 137157
- GAPWM: a genetic algorithm method for optimizing a position weight matrix Li, Liang, Bass (2007) http://bioinformatics.oxfordjournals.org/cgi/content/short/23/10/1188
- A Feature-Based Approach to Modeling Protein–DNA Interactions
  Eilon Sharon,[#1] Shai Lubliner,[#1] and **Eran Segal**[1,2*]
  PLoS Comput Biol. 2008 August; 4(8): e1000154. doi:10.1371/journal.pcbi.1000154.
  http://www.pubmedcentral.nih.gov/articlerender.fcgi?tool=pubmed&pubmedid=18725950

HMM (see also previous notes)

- **Introductory material on hidden Markov models**
- *``Profile hidden Markov models'' Sean R. Eddy, Bioinformatics 14(9):75563, 1998.* (PDF)

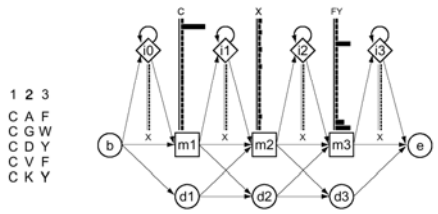
http://bioit.cs.ut.ee/%7Evilo/edu/2005-06/Text_Algorithms/L10_Probabilistic/Articles/hmmreview-bioinformatics-98.pdf

- *``Hidden Markov Models and Protein Sequence Analysis'' Rachel Karchin, UCSC.*



- Given an HMM, what is the p that the state has been passed through and respective sequence output?
- Given the sequences, generate the HMM that maximises the probability that these sequences are generated by that HMM
- Examples generate the model, model generates the examples
- Describe a family of sequences with a single HMM (e.g. protein family)
- For a new sequence what is the probability to be from the same family?

More complex example
Proteins, alphabet of size 20
Would produce 3/character long sequences
Plus states for **i**nsertion and **d**eletion



In state m1 the most probable is C, in state m2 any, and m3 F, Y and W.

Using hidden Markov models

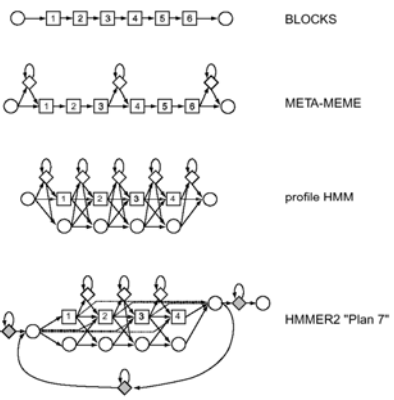
http://en.wikipedia.org/wiki/Hidden_Markov_model

There are three canonical problems associated with HMM:

- Given the parameters of the model, compute the probability of a particular output sequence, and the probabilities of the hidden state values given that output sequence. This problem is solved by the forwardbackward algorithm.
- Given the parameters of the model, find the most likely sequence of hidden states that could have generated a given output sequence. This problem is solved by the Viterbi algorithm.
- Given an output sequence or a set of such sequences, find the most likely set of state transition and output probabilities. In other words, discover the parameters of the HMM given a dataset of sequences. This problem is solved by the BaumWelch algorithm.

HMM training

- Different representations, topologies
- Usually the training algorith learns the parameters (weights), not the topology
- It is up to the designer to come up with good topology.



Stochastic Context Free Grammars

- Stohhastilised kontekstivabad grammatikad
- Articles http://biit.cs.ut.ee/%7Evilo/edu/2005-06/Text_Algorithms/L10_Probabilistic/Articles/
- Sakakibara et.al. 1993

$$
\begin{aligned}
P = \{\ & S_0 \to S_1, && S_7 \to \mathtt{G}\,S_8,\\
& S_1 \to \mathtt{C}\,S_2\,\mathtt{G}, && S_8 \to \mathtt{G},\\
& S_1 \to \mathtt{A}\,S_2\,\mathtt{U}, && S_8 \to \mathtt{U},\\
& S_2 \to \mathtt{A}\,S_3\,\mathtt{U}, && S_9 \to \mathtt{A}\,S_{10}\,\mathtt{U},\\
& S_3 \to S_4\ S_9, && S_{10} \to \mathtt{C}\,S_{10}\,\mathtt{G},\\
& S_4 \to \mathtt{U}\,S_5\,\mathtt{A}, && S_{10} \to \mathtt{G}\,S_{11}\,\mathtt{C},\\
& S_5 \to \mathtt{C}\,S_6\,\mathtt{G}, && S_{11} \to \mathtt{A}\,S_{12}\,\mathtt{U},\\
& S_6 \to \mathtt{A}\,S_7, && S_{12} \to \mathtt{U}\,S_{13},\\
& S_7 \to \mathtt{U}\,S_7, && S_{13} \to \mathtt{C} \qquad \}
\end{aligned}
$$

Figure 1: This set of productions P generates RNA sequences with a certain restricted structure, such as that in Figure 2. $S_0, S_1, \ldots, S_{13}$ are nonterminals; A, U, G and C are terminals representing the four nucleotides.

$$
\begin{aligned}
S_0 &\Rightarrow S_1 \Rightarrow \mathtt{C}S_2\mathtt{G} \Rightarrow \mathtt{CA}S_3\mathtt{UG} \Rightarrow \mathtt{CA}S_4S_9\mathtt{UG}\\
&\Rightarrow \mathtt{CAU}S_5\mathtt{A}S_9\mathtt{UG} \Rightarrow \mathtt{CAUC}S_6\mathtt{GA}S_9\mathtt{UG}\\
&\Rightarrow \mathtt{CAUCA}S_7\mathtt{GA}S_9\mathtt{UG} \Rightarrow \mathtt{CAUCAG}S_8\mathtt{GA}S_9\mathtt{UG}\\
&\Rightarrow \mathtt{CAUCAGGGA}S_9\mathtt{UG} \Rightarrow \mathtt{CAUCAGGGAA}S_{10}\mathtt{UUG}\\
&\Rightarrow \mathtt{CAUCAGGGAAG}S_{11}\mathtt{CUUG}\\
&\Rightarrow \mathtt{CAUCAGGGAAGA}S_{12}\mathtt{UCUUG}\\
&\Rightarrow \mathtt{CAUCAGGGAAGAU}S_{13}\mathtt{UCUUG}\\
&\Rightarrow \mathtt{CAUCAGGGAAGAUCUCUUG}.
\end{aligned}
$$

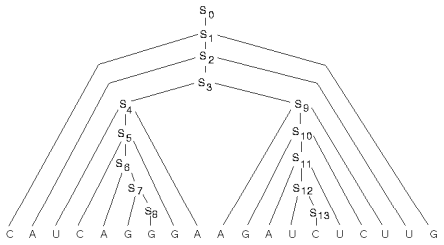
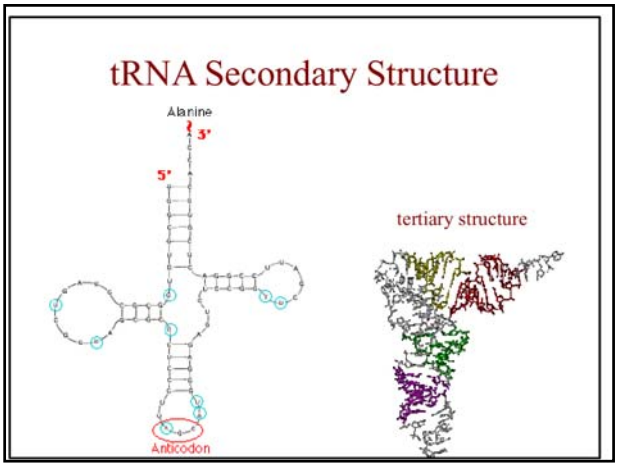


Figure 2: For the RNA sequence **CAUCAGGG AAGAUCUCUUG**, the grammar whose productions are given in Figure 1 yields this parse tree. This parse tree reflects a specific secondary structure.



$$
\begin{aligned}
\mathrm{Prob}(s \mid G) &= \sum_{\substack{\text{all derivations (or}\\ \text{parse trees) } d}} \mathrm{Prob}(S_0 \overset{d}{\Rightarrow} s \mid G)\\
&= \sum_{\alpha_1, \ldots, \alpha_n} \mathrm{Prob}(S_0 \Rightarrow \alpha_1 \mid G)\\
&\quad \cdot \mathrm{Prob}(\alpha_1 \Rightarrow \alpha_2 \mid G) \cdot \cdots\\
&\quad \cdot \mathrm{Prob}(\alpha_n \Rightarrow s \mid G)
\end{aligned}
$$

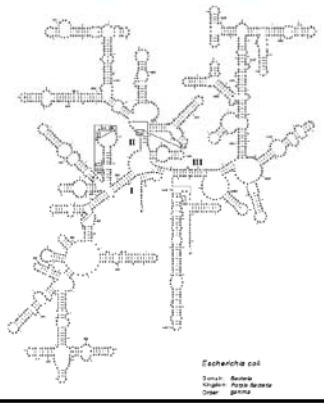
Efficiently computing this quantity, $\mathrm{Prob}(s \mid G)$, presents a problem because the number of possible parse trees for s is exponential in the length of the sequence. However, a dynamic programming technique analogous to the Cocke-Kasami-Young or Early methods [1] for non-stochastic CFGs can accomplish this task efficiently (in time proportional to the cube of the length of s). We define the negative logarithm of the probability of a sequence given by the grammar, $-\log(\mathrm{Prob}(s \mid G))$, as the *negative log likelihood (NLL) score* of the sequence. This quantifies how well the sequence s fits the grammar.

Type I	$S \rightarrow SS$	1.0		
Type II	$S \rightarrow A\ S\ A$	0.05	$S \rightarrow A\ S\ C$	0.05
	$S \rightarrow A\ S\ G$	0.05	$S \rightarrow A\ S\ U$	0.1
	$S \rightarrow C\ S\ A$	0.05	$S \rightarrow C\ S\ C$	0.05
	$S \rightarrow C\ S\ G$	0.1	$S \rightarrow C\ S\ U$	0.05
	$S \rightarrow G\ S\ A$	0.05	$S \rightarrow G\ S\ G$	0.05
	$S \rightarrow G\ S\ C$	0.1	$S \rightarrow G\ S\ U$	0.05
	$S \rightarrow U\ S\ A$	0.1	$S \rightarrow U\ S\ G$	0.05
	$S \rightarrow U\ S\ C$	0.05	$S \rightarrow U\ S\ U$	0.05
Type III	$S \rightarrow A\ S$	0.2*	$S \rightarrow G\ S$	0.2*
	$S \rightarrow C\ S$	0.2*	$S \rightarrow U\ S$	0.2*
	$S \rightarrow S$	0.2*		
Type IV	$S \rightarrow A$	0.25	$S \rightarrow G$	0.25
	$S \rightarrow C$	0.25	$S \rightarrow U$	0.25

Figure 3: To obtain production probabilities for this initial grammar, we placed a uniform distribution over each set of same-type productions, but weighted Watson-Crick base pairs twice as heavily. Starred values may differ if no skip productions are needed. For simplicity, this figure omits subscripts.

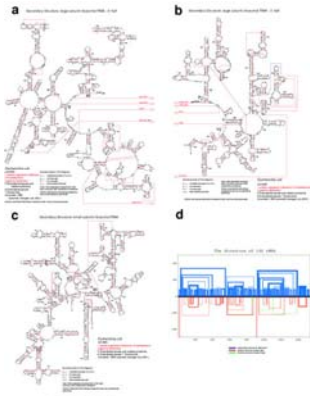
- Proc Int Conf Intell Syst Mol Biol. 2000;8:5766.Links
- **Small subunit ribosomal RNA modeling using stochastic contextfree grammars.**
- **Brown MP**.
- HNC, San Diego, CA 921213278, USA. mpsb@hnc.com
- We introduce a model based on stochastic contextfree grammars (SCFGs) that can construct small subunit ribosomal RNA (SSU rRNA) multiple alignments. The method takes into account both primary sequence and secondary structure basepairing interactions. We show that this method produces multiple alignments of quality close to hand edited ones and outperforms several other methods. We also introduce a method of SCFG constraints that dramatically reduces the required computer resources needed to effectively use SCFGs on large problems such as SSU rRNA. Without such constraints, the required computer resources are infeasible for most computers. This work has applications to fields such as phylogenetic tree construction.
- http://www.ncbi.nlm.nih.gov/pubmed/10977066

Small Subunit Ribosomal RNA




```
UGGAGAGUUUGAUCCUGGCUCAGGGUGAACGCUGGCGG CGUGCCUAAGACAUGCAAGUCGUGCGGCCGCGGGGUA
CUCCGUGGUCAGCGGCGGACGGGUGAGUAACGCGUGGG UGACCUACCCGGAAGAGGGGGACAACCCGGGGAAACUC
GGGCUAAUCCCCCAUGUGGACCCGCCCCUUGGGGUGUG UCCAAAGGGCUUUGCCCGCUUCCGGAUGGGCCCGCGUC
CCAUCAGCUAGUUGGUGGGGUAAUGGCCCACCAAGGCG ACGACGGGUAGCCGGUCUGAGAGGAUGGCCGGCCACAG
GGGCACUGAGACACGGGCCCCACUCCUACGGGAGGCAG CAGUAGGAAUCUUCCGCAAUGGGCGCAAGCCUGACGG
AGCGACGCCGCUUGGAGGAAGAAGCCCUUCGGGGUGUA AACUCCUGAACCCGGGACGAAACCCCCGACGAGGGGAC
UGACGGUACCGGGGUAAUAGCGCCGGCCAACUCCGUGC CAGCAGCCGCGGUAAUACGGAGGGCGCGAGCGUUACCC
GCGUAAAGGGCGUGUAGGCGGCCUGGGGCGUCCCAUGU GAAAGACCACGGCUCAACCGUGGGGGAGCGUGGGAUAC
GCUCAGGCUAGACGGUGGGGAGAGGGUGGUGGAAUUCCC GGAGUAGCGGUGAAAUGCGCAGAUACCGGGAGGAACGC
CGAUGGCGAAGGCAGCCACCUGGUCCACCCGUGACGCU GAGGCGCGAAAGCGUGGGGAGCAAACCGGAUUAACCCG
GGUAGUCCACGCCCUAAACGAUGCGCGCUAGGUCUCUG GGUCUCCUGGGGGCCGAAGCUAACGCGUUAAGCGCGCC
GCCUGGGGAGUACGGCCGCAAGGCUGAAACUCAAAGGA AUUGACGGGGGCCCGCACAAGCGGUGGAGCAUGUGGUU
UAAUUCGCGAAGAACCUUACCAGGCCUUGACAUGCUAG GGAACCCGGGUGAAAGCCUGGGGUGCCCGCGAGGGAGC
CCUAGCACAGGUGCUGGGCCGUCGUCAGCUCGUGCCGU GAGGUGUUGGGUUAAGUCCCGCAACGAGCGCAACCCCC
GCCGUUAGUUGCCAGCGGUUCGGCCGGGCACUCUAACG GGACUGCCCGCGAAAGCGGGAGGAAGGAGGGGACGACG
UCUGGUCAGCAUGGCCCUUACGGCCUGGGCGACACACG UGCUACAAUGCCCUACAAAGCGAUGCCACCCGGCAACG
GGGAGCUAAUCGCAAAAAGGUGGGCCCAGUUCGGAUUG GGGUCUGCAACCCGACCCCAUGAAGCCGGAAUCGCUAG
UAAUGCGCGGAUCAGCCAUGCCGCGGUGAAUACGUUCCC GGGCCUUGUACACACCGCCCGUCACGCCAUGGGAGCGG
GCUCUACCCGAAGUCGCCGGGAGCCUACGGGCAGGCGC CGAGGGUAGGGCCCGUGACUGGGGCGAAGUCGUAACAA
GGUAGCUGUACCGGAAGGUGCGGCUGGAUCACCUC
```

RNA secondary Structure representation



Large subunits rRNA

Small subunit rRNA

30S Small subunit of Ribosomal RNA

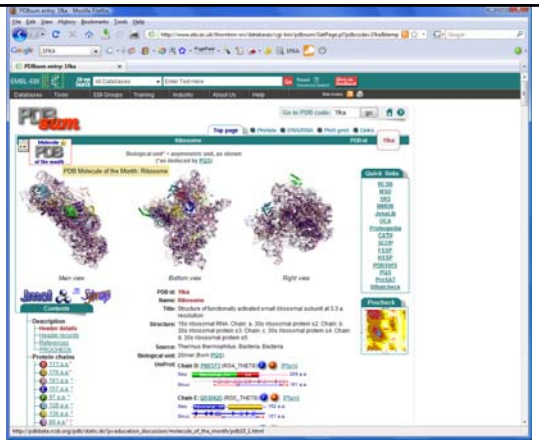
The **16S** rRNA is a 1542 nt long component of the small prokaryotic ribosomal subunit (30S) and has several functions

30S is the smaller subunit of the 70S ribosome of prokaryotes. It is a complex of ribosomal RNA and ribonucleoproteins which functions in mRNA translation. It includes the subunit 16S, which is composed entirely of ribosomal RNA.

The 30S subunit is the site of inhibition for antibiotics such as tetracycline.

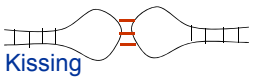
http://en.wikipedia.org/wiki/30S

Atomic structure of the 30S Subunit from *Thermus thermophilus*. Proteins are shown in blue and the single RNA strand in orange.[1]



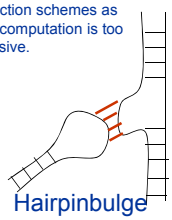
Examples of known interactions of RNA secondary structural elements

Pseudoknot



Kissing hairpins

These patterns are excluded from the prediction schemes as their computation is too intensive.

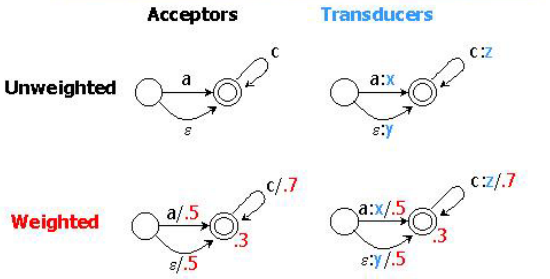


Hairpinbulge contact

- **Stochastic Context Free Grammars (SCFG)**
- Loengumaterjal, Mark Craven

- **Main questions**
- How probable is the sequence for this grammar? the Inside algorithm
- What is the most probable parse tree? the CockeYoungerKasami (CYK) algorithm
- How to learn the parameters from the examples? the InsideOutside algorithm

Finite state machines



JHU 600.405 - Finite-State Methods in NLP - Fall 2000 - J. Eisner 6

Definition of FST

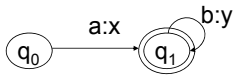
A FST is $(Q, \Sigma, \Gamma, I, F, \delta)$

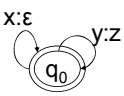
- Q: a finite set of states
- Σ: a finite set of input symbols
- **Γ: a finite set of output symbols**
- I: the set of initial states
- F: the set of final states
- $\delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times (\Gamma \cup \{\varepsilon\}) \times Q$: the transition relation between states.

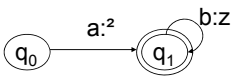
➔ FSA can be seen as a special case of FST

Fei Xia; *courses.washington.edu/ling570/fei_fall07/10_10_**FST**.ppt*

An example of composition operation







Fei Xia; *courses.washington.edu/ling570/fei_fall07/10_10_**FST**.ppt*

FST Algorithms

- **Recognition**: Is a given pair of strings accepted by an FST?
  - (x,y) ➔ yes/no
- **Composition**: Given two FSTs T_1 and T_2 defining regular relations R_1 and R_2, create the FST that computes the composition of R_1 and R_2.
  - R1={(x,y)}, R2={(y,z)} ➔ {(x,z) | (x,y) 2 R_1, (y,z) 2 R_2}
- **Transduction**: given an input string and an FST, provide the output as defined by the regular relation?
  - x ➔ y

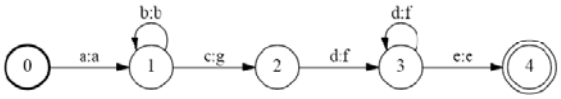
Fei Xia; *courses.washington.edu/ling570/fei_fall07/10_10_**FST**.ppt*

Finite-state transducers

A (2-way) finite-state transducer is a quintuple $M = (K, s, F, \Sigma \times \Sigma, \delta)$ where:
1. K is a finite set of states
2. s is a designated initial state
3. F is a designated set of final states
4. Σ is an alphabet of symbols, and
5. δ is a transition relation from $K \times (\Sigma \cup \epsilon \times \Sigma \cup \epsilon)$ to K

Richard Sproat: *compling.ai.uiuc.edu/rws/newindex/acm.ppt* 56

An FST



Richard Sproat: *compling.ai.uiuc.edu/rws/newindex/acm.ppt* 57

Composition

- In addition to *union, concatenation* and *Kleene closure,* regular relations are closed under *composition*
- Composition is to be understood here the same way as composition in algebra:
  - $R_1 o R_1$ means take the output of R_1 and feed it to the input of R_2

Richard Sproat: *compling.ai.uiuc.edu/rws/newindex/acm.ppt* 58

Composition: an illustration

Consider the following simple example using rewrite rules:

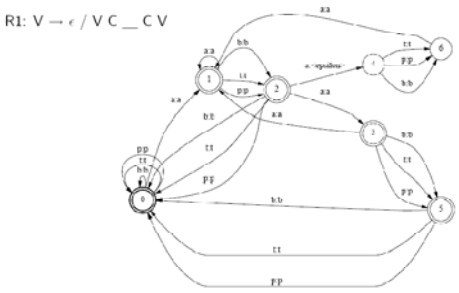
R1: V → ϵ / V C __ C V
R2: b → p / __ t

tabata (via R1) → tabta (via R2) → tapta
As it happens, rewrite rules such as the above are implementable using FST's.

Richard Sproat: *compling.ai.uiuc.edu/rws/newindex/acm.ppt* 59

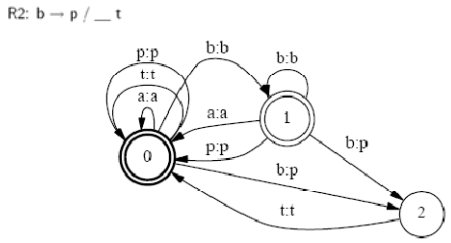
R1 as a transducer

R1: V → ϵ / V C __ C V



Richard Sproat: *compling.ai.uiuc.edu/rws/newindex/acm.ppt* 60

R2 as a transducer



Richard Sproat: *compling.ai.uiuc.edu/rws/newindex/acm.ppt* 61

R1○R2

Richard Sproat: *compling.ai.uiuc.edu/rws/newindex/acm.ppt* 62

Full NLP with FST-s?

Description of a natural language = one big transducer

Speech-Text FST ○ Morph-FST ○ Syntax-FST ○ Semantics-FST

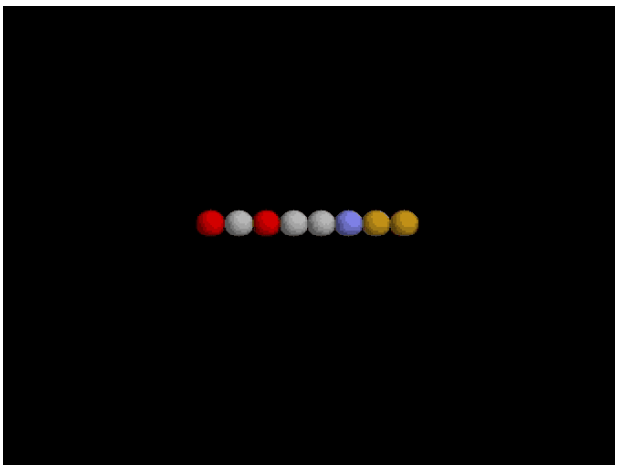
analysis →

generation ←

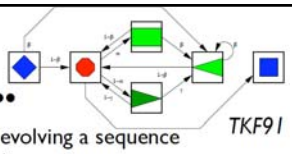
Heli Uibo

Transducers

- Help to model the evolution
- "Transform sequence into another"
- Compose on different branches of evolutionary tree
- http://biowiki.org/PhyloFilm
- http://biowiki.org/~yam/fruitfly/TKFmovie/

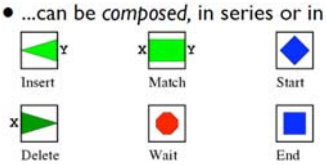


Transducers...

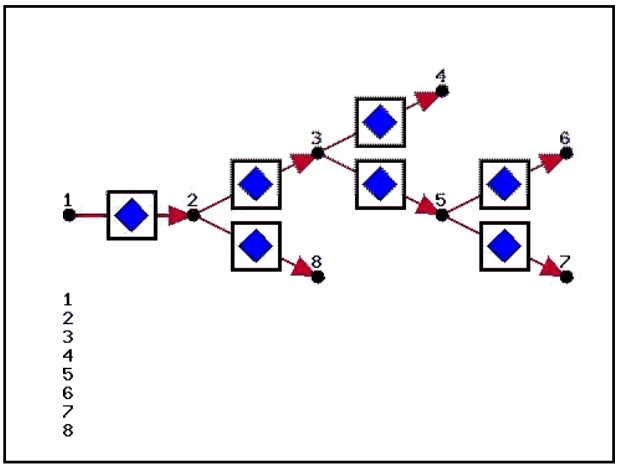


TKF91

- ...model the operation of evolving a sequence along one branch of a phylogenetic tree
- ...are stochastic finite-state machines
- ...have an *input tape*, X, and an *output tape*, Y
- ...can be *composed*, in series or in parallel



Pair HMM: *P*(X,Y)
Transducer: *P*(Y|X)



"Every good work of software starts by scratching a developer's personal itch" - Eric Raymond

2001 (Holmes & Bruno)
MCMC based on TKF91. **Poor performance due to affine gaps, rate variation**

2002 (Holmes & Rubin)
EM algorithm for estimating substitution rates & in particular rate variation

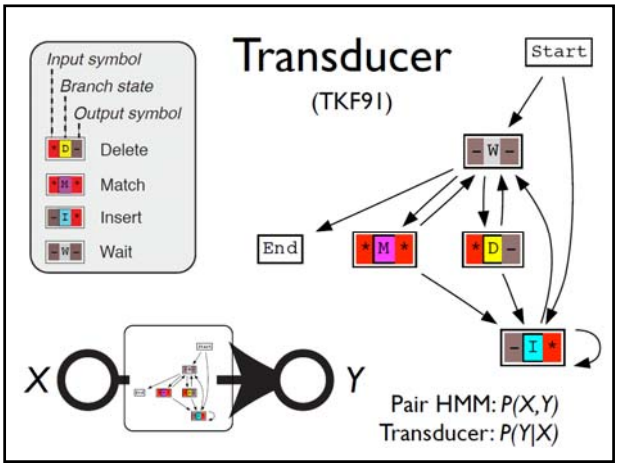
2003 (Holmes)
General algorithm for transducer composition on phylogenetic trees

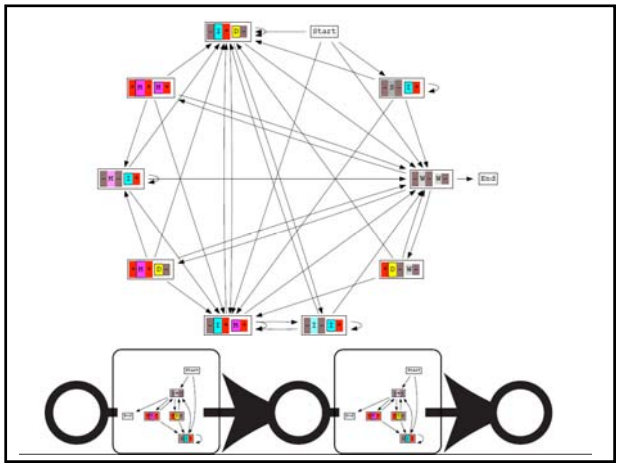
2004 (Miklòs, Lunter & Holmes)
"Long Indel" model: affine-gap Pair HMMs from evolutionary models

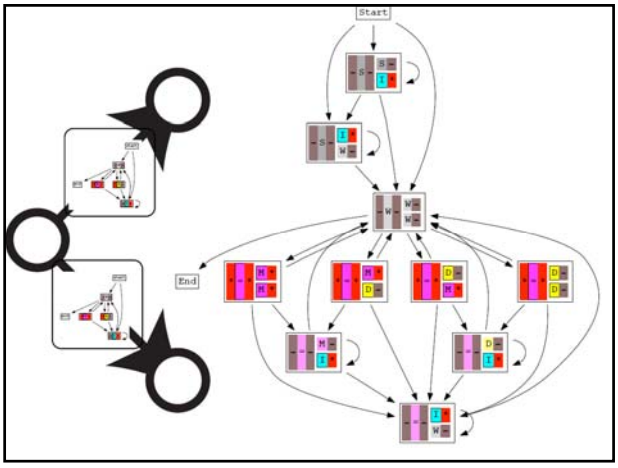
2005-2006 (Holmes; Klosterman *et al*)
(Started extending transducer theory to RNA sequence analysis & SCFGs)
Enormous difficulties debugging transducer composition & DP algorithms!

2007 (Holmes)
Phylocomposer: a transducer compositor, parser-generator & debugger

Ian Holmes (UC Berkeley)



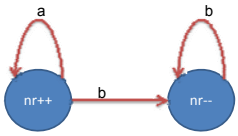




Video lectures

- http://videolectures.net/aop07_bertinoro/
- Mehryar Mohri, **Weighted Transducers and Rational Kernels**
  - http://videolectures.net/aop07_mohri_wtt/
- **Similarity and differences by finite automata** Tamás Gaál, XEROX Research Centre Europe
  - http://videolectures.net/aop07_gaal_sdf/

Attributed automata



$a^n b^n$ (if nr==0)

Graph Edit Distance

- Measuring the distance (or similarity of graphs) is an important task in pr and related areas.
- Graph edit distance (ged) is one of the most general graph distance measures.
- ged measures the distance of a pair of graphs in terms of the minimum number of edit operations required to transform one graph into the other one.

A Quadratic Programming Approach to the Graph Edit Distance Problem June, 2007 2