Hidden Markov Models

Richard Hughey
Department of Computer Engineering
Center for Biomolecular Science and Engineering
University of California, Santa Cruz

http://www.cse.ucsc.edu/research/compbio/

HMM Tutorial — Acknowledgments

- Tutorials from ISMB96 and ISMB99. C. Barrett, M. Cline, L. Grate, R. Hughey, K. Karplus, K. Sjolander. BioComp00 presentation by K. Karplus.
- SAM-Tx developed by Kevin Karplus with Christian Barrett and Richard Hughey Additional collaborators: David Haussler, Melissa Cline, Mark Diekhans, Leslie Grate, Kimmen Sjölander.
- SAM hidden Markov model package originally developed by Anders Krogh and David Haussler.
- The sequence logo program used is by J. Casper, UCSC, based on the Sequence Logos program by Tom Schneider.
- Gene finding. D. Kulp, D. Haussler, et al.
- Supported in part by NSF grants DBI-9808007 and EIA-9905322, DOE grant DE-FG03-99ER62849.

Bioinformatics: A New Field in Engineering Education

Bioinformatics at UCSC

1991 David Haussler moves from machine learning to bioinformatics.

- Hidden Markov Models for sequence analysis

1993 Stochastic Context Free Grammars for RNA

1993 Dirichlet Mixtures

1994 Kestrel SIMD parallel processor for HMMs

1994 Kevin Karplus VLSI/CAD ⇒ Bioinformatics

- Protein structure prediction

1996 Gene finding

Bioinformatics at UCSC

1998 Center for Biomolecular Science and Engineering

http://www.cse.ucsc.edu/research/centers/cbe

1998 Microarray data analysis, Support Vector Machines

2000 Human genome assembly (Jim Kent)

2001 BS in Bioinformatics

2001 Todd Lowe hired for Biomolecular Engineering

2002 Carol Rohl hired for Biomolecular Engineering

2003 Bioinformatician hired

2003? MS/PhD in Bioinformatics approved

2003? Biomolecular Engineering Department

Outline

- What is an HMM?
  - Alignments to Profiles to HMMs
  - Building Models
  - Using SAM
- How is an HMM used effectively?
  - Iterative HMM construction
  - SAM-Tx
- Gene-finding HMM topologies

rat	ACGGA
ecoli	ACA
cow	AA
corn	AUA

rat	A	C	G	G	A
ecoli	A	C	-	-	A
cow	A	-	-	-	A
corn	A	U	-	-	A

rat	A	C	G	G	A
ecoli	A	C	-	-	A
cow	A	-	-	-	A
corn	A	U	-	-	A

A	100%	0%	0%	0%	100%
C	0%	50%	0%	0%	0%
G	0%	0%	25%	25%	0%
U	0%	25%	0%	0%	0%
-	0%	25%	75%	75%	0%

- The Maximum Likelihood probabilities for each column are computed by simply using the observed letter counts for each column.
- Probability of letter X in column i is:

$$P_i(X_i) = (\text{Counts of } X \text{ in } i)/(\text{Total counts in } i)$$

- Given a Profile and Alignment, we can compute the probability of a sequence in the alignment.
- Assume:
  columns are independent.
  This lets us compute the probability as:

$$P(\text{seq}|\text{alignment, profile}) = \prod_i P_i(c_i)$$

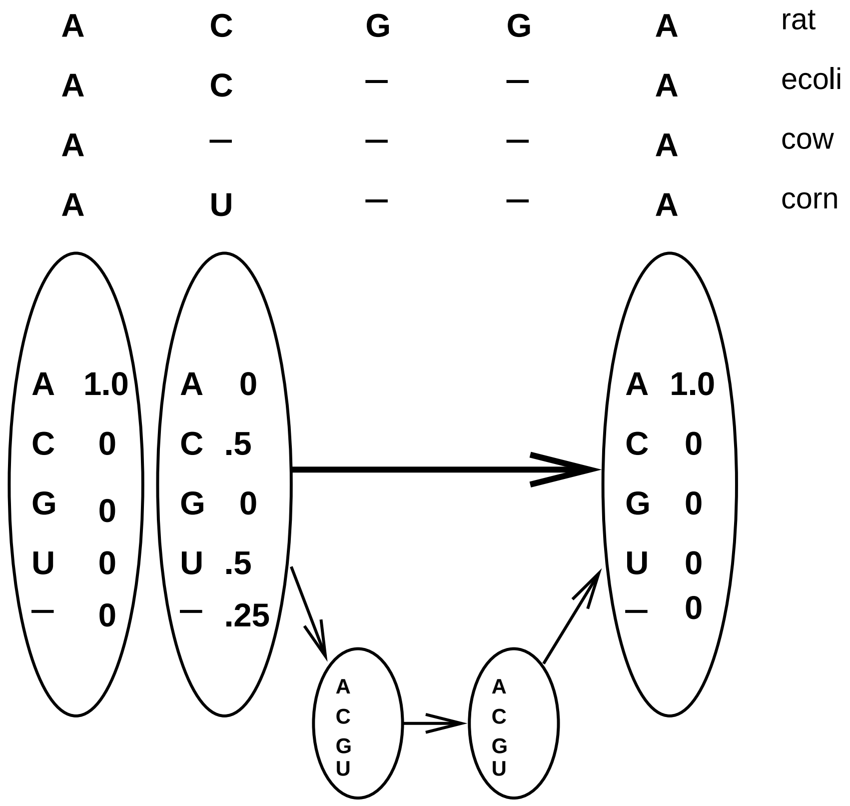
rat	A	C	G	G	A
A	1.0	×	×	×	1.0
C	×	0.5	×	×	×
G	×	×	0.25	0.25	×
U	×	0.25	×	×	×
-	×	0.25	0.75	0.75	×

$$P(rat|profile, alignment) = 1.0 * 0.5 * 0.25 * 0.25 * 1.0$$

A	C	G	G	A	rat
A	C	–	–	A	ecoli
A	–	–	–	A	cow
A	U	–	–	A	corn

A	1.0	A	0	A	0	A	0	A	1.0
C	0	C	.5	C	0	C	0	C	0
G	0	G	0	G	.25	G	.25	G	0
U	0	U	.5	U	0	U	0	U	0
–	0	–	.25	–	.75	–	.75	–	0

- In the previous alignment, only 'Rat' has two Gs
- The Gs seem to be extra
- We'll turn them into an explicit insertion.

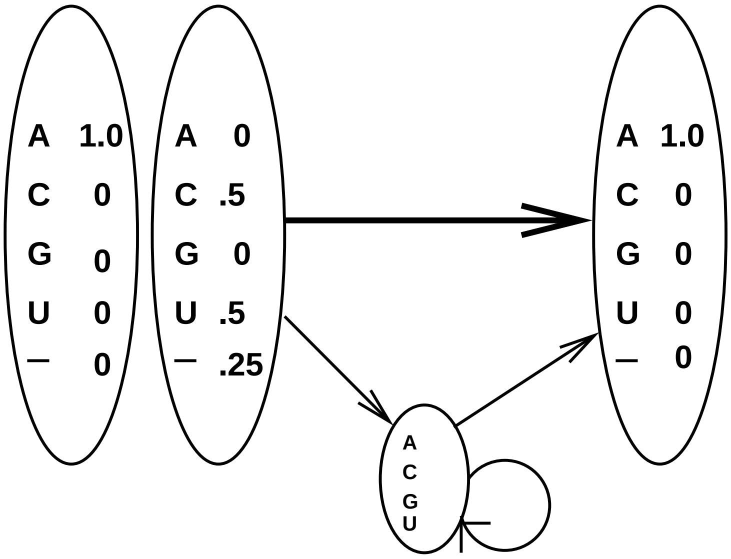


There's a problem with this:

- How many insertions do we need?
- Should we allow arbitrary-length insertion chains?
- We could, in a generalized hidden Markov model.
- But perhaps we could construct something simpler
  - (And faster to evaluate)

Multiple alignments to HMMs

A	C	G	G	A	rat
A	C	–	–	A	ecoli
A	–	–	–	A	cow
A	U	–	–	A	corn

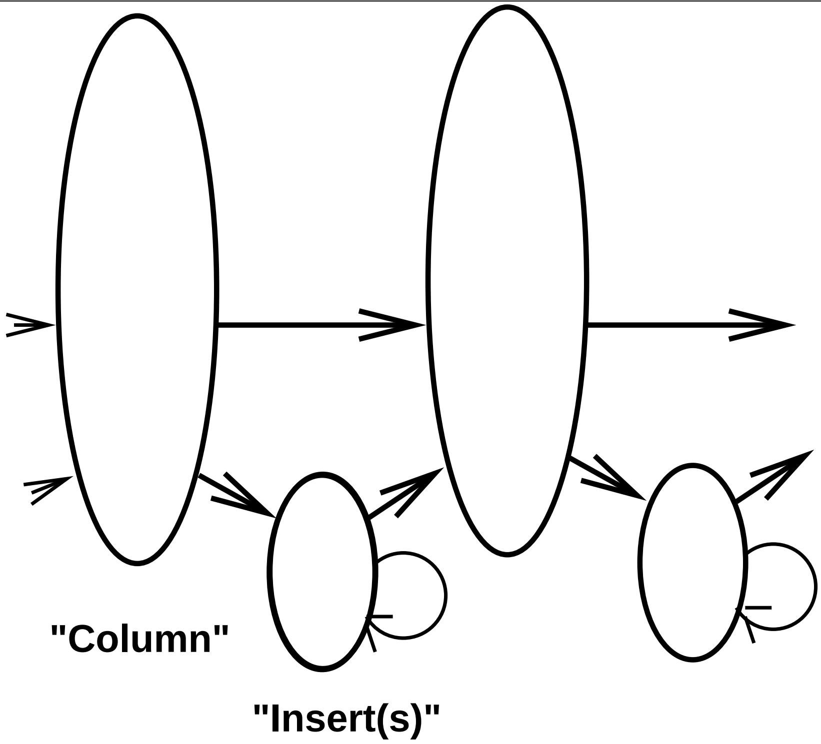


Hidden Markov Models

What are hidden Markov models?

- An extension of profiles with gaps
- Add transitions between columns
- Add insertions between columns
- Assign probabilities to all transitions

And then figure out how to train them

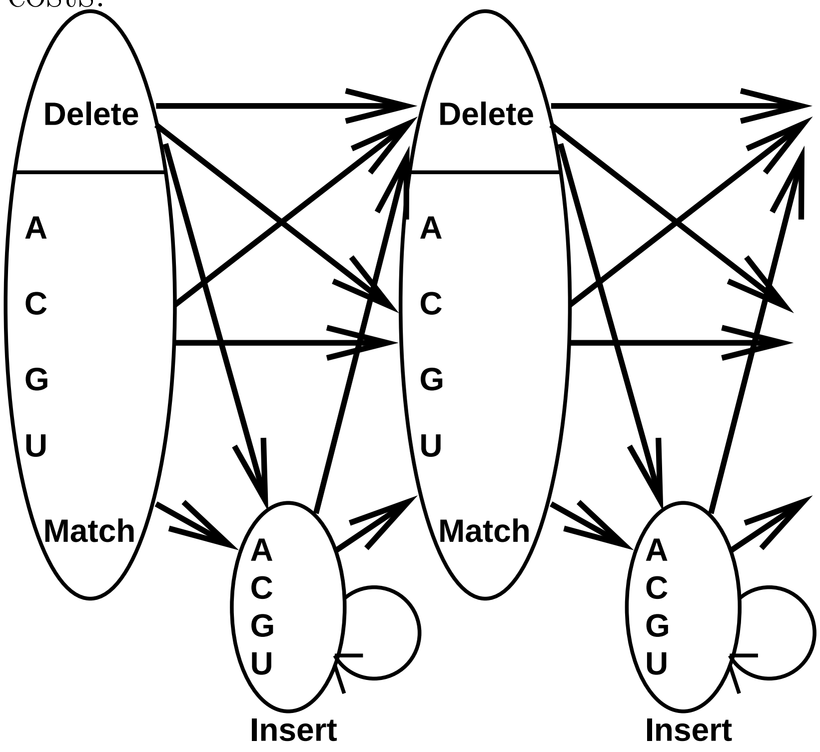


These insertions are just like "affine insertion costs": a charge to get into an insert, and a different charge to continue inserting.

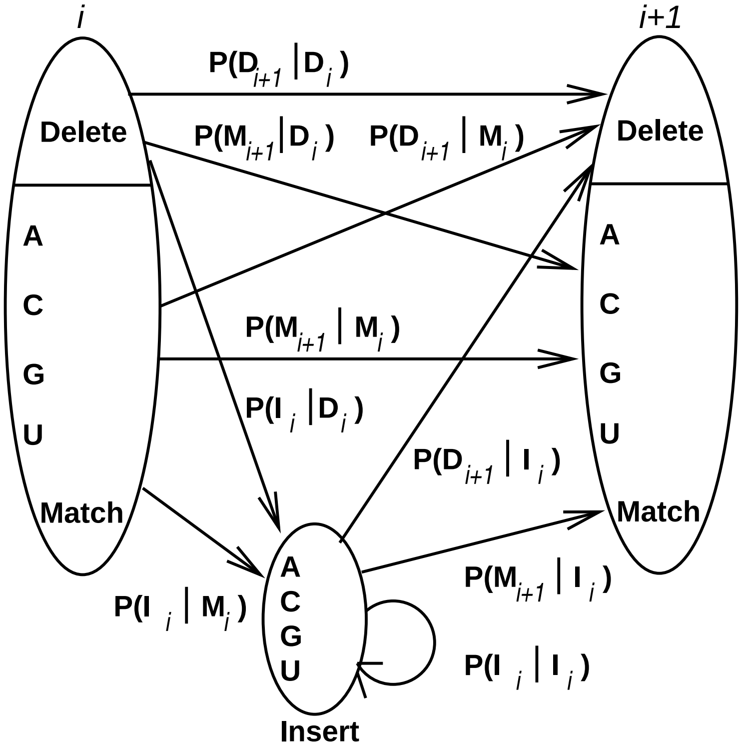
Why 'Hidden'?

- We observe the letters generated by the model
- We do not observe the sequence of states that generated those letters
- But we can guess what they are from an alignment

We can also make Deletes be their own state, rather than a dash, to allow affine deletion costs.

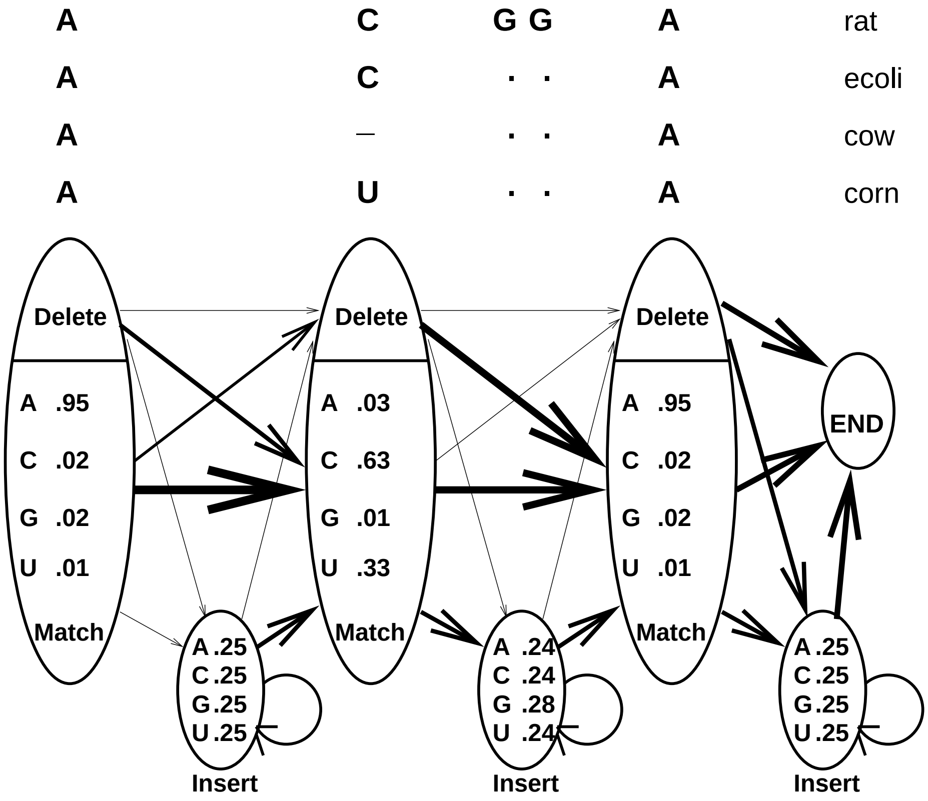


And then add probabilities



9 transition and 40 character probabilities **per column**

Hidden Markov Models



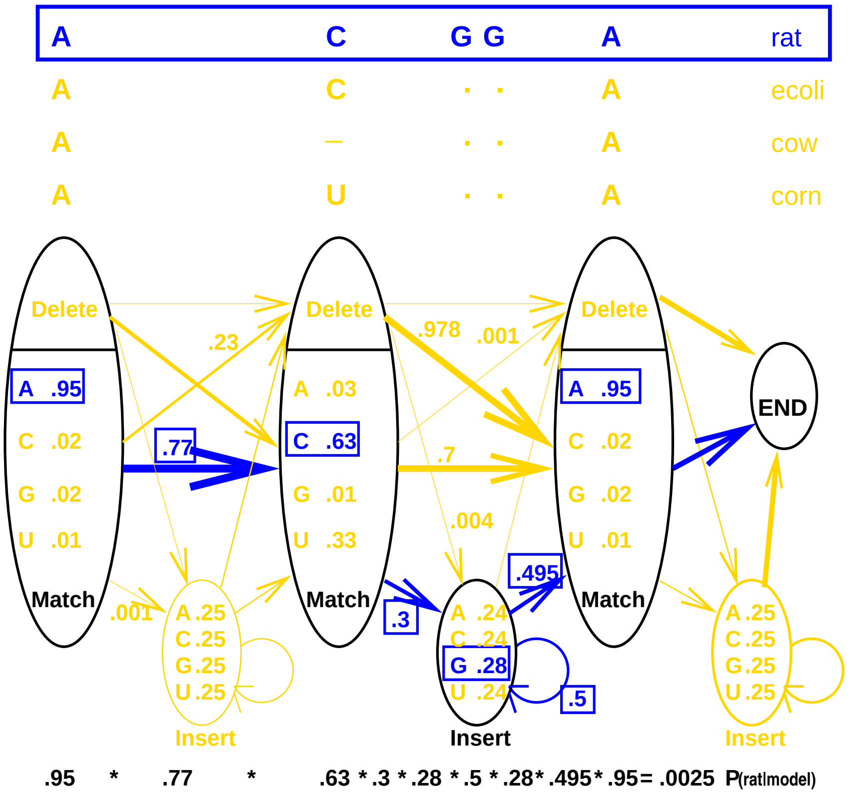
Hidden Markov Model Alignment

As before, given a **Model** and an alignment, we can calculate the probability of that sequence given that alignment to the model. The result is similar to profiles, but with affine gap costs.

$$P(\text{seq}|\text{model, alignment}) = \prod_{\text{path}} P_i(c_i) \cdot T(i+1|i)$$

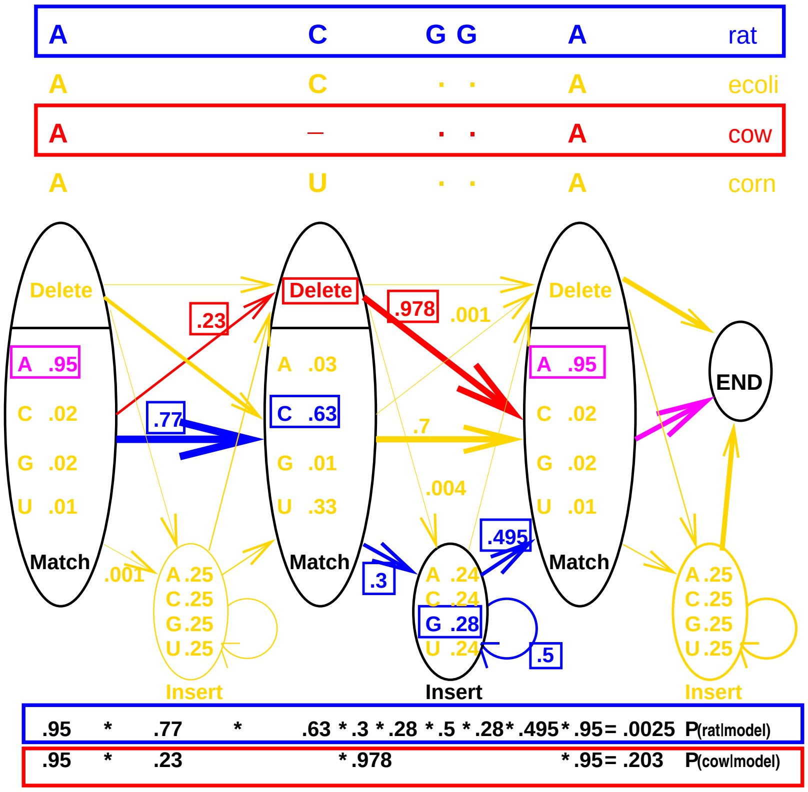
Given a sequence and a model, we can align the sequence to the model by finding the most probable path through the model.

This uses dynamic programming similar to standard sequence analysis.



If we align two sequences to the model we can:

- Determine which sequence is more typical of the family being modeled.
- Construct a multiple alignment among the sequences aligned to the model.



Thus, $P(\text{rat}|\text{model,align}) = 0.0025$ and $P(\text{cow}|\text{model,align}) = 0.203$, so cow is much more typical, right?

- Sequences of different length will have different average scores
- We need to adjust for length with a null model
- We can use log-odds to avoid large negative exponents

$$\log \frac{P(\text{rat}|\text{model,align})}{P(\text{rat}|\text{Null})} \quad \stackrel{?}{<} \quad \log \frac{P(\text{cow}|\text{model,align})}{P(\text{cow}|\text{Null})}$$

Comparing Hidden Markov Model Probabilities

- How about a simple null model: the probability of each character is 0.25.
- $P(\text{rat}|\text{Null}) = (0.25)^5 = 0.0009766$
- $P(\text{cow}|\text{Null}) = (0.25)^2 = 0.0625$
- Log-odds-score(rat)= $\log_2 \frac{P(r|m,a)}{P(r|N)} = 1.36$ bits
- Log-odds-score(cow)= $\log_2 \frac{P(c|m,a)}{P(r|N)} = 1.69$ bits

Note: SAM used $\log_e$ and costs (negative scores)

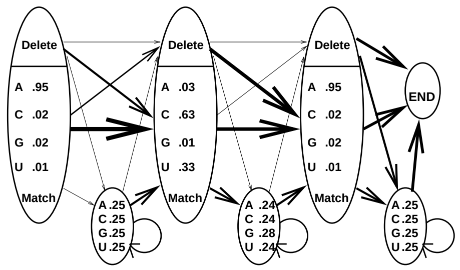
Dynamic Programming: Affine Costs

$$c^D_{i,j} = \min(\, c^M_{i,j-1} + g^{M\to D},\ c^I_{i,j-1} + g^{I\to D},\ c^D_{i,j-1} + g^{D\to D}) \ + \mathrm{dist}(\phi, b_j)$$
$$c^M_{i,j} = \min(\, c^M_{i-1,j-1} + g^{M\to M},\ c^I_{i-1,j-1} + g^{I\to M},\ c^D_{i-1,j-1} + g^{D\to M}) \ + \mathrm{dist}(a_i, b_j)$$
$$c^I_{i,j} = \min(\, c^M_{i-1,j} + g^{M\to I},\ c^I_{i-1,j} + g^{I\to I},\ c^D_{i-1,j} + g^{D\to I}) \ + \mathrm{dist}(a_i, \phi)$$

	ϕ	A	G	A	C
ϕ	D0 I× M×	D1.1 I× M×	D1.2 I× M×	D1.3 I× M×	D1.4 I× M×
A	D× I1.1 M×	D2.2 I2.2 M0	D1.1 I2.3 M2.1	D1.2 I2.4 M1.2	D1.3 I2.5 M3.3
A	D× I1.2 M×	D2.3 I1.1 M3.1	D2.2 I2.2 M2	D2.3 I2.3 M1.1	D2.2 I2.4 M3.2
C	D× I1.3 M×	D2.4 I1.2 M3.2	D2.3 I2.3 M3.1	D2.4 I2.2 M4	D2.5 I1.1 M2.5

Dynamic Programming: Linear HMM

All costs are position-dependent



ϕ	D0 I× M×	D I× M×	D I× M×	D I× M×
A	D I M	D I M	D I M	D I M
C	D I M	D I M	D I M	D I M
A	D I M	D I M	D I M	D I M

There are several ways to do the dynamic programming

- Viterbi, or single-best-path, uses max and sum
- Forward-Backward adds up probabilities along all possible paths
- Posterior-decoded alignment performs Viterbi dynamic programming on the results of Forward-Backward.
- Unfortunately, the slower methods are better!

To create an HMM.....

- Start with model(s) of some length.
  - Average sequence length
  - Expected motif length
- Add noise to make states different
- Use DP to align each sequence
- Calculate frequencies
- Update model probabilities
- Change the model if needed
- Repeat until little change

Iterations of Building a Model

- iteration 1

rat	a	c	G	G	A
ecoli	.	.	A	C	A
cow	.	.	-	A	A
corn	.	.	A	U	A

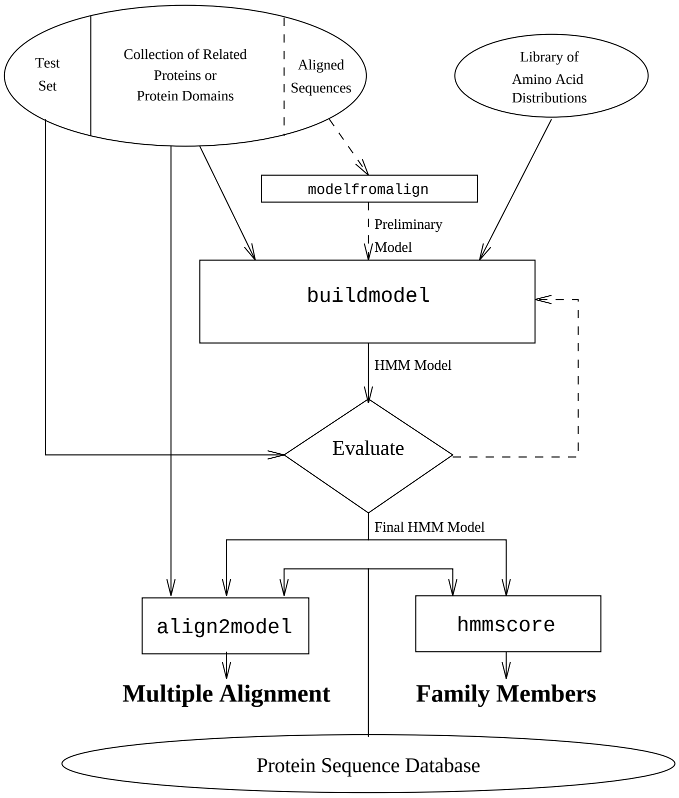
- iteration 2

rat	A	C	g	g	A
ecoli	A	C	.	.	A
cow	-	A	.	.	A
corn	A	U	.	.	A

- iteration 3

rat	A	C	g	g	A
ecoli	A	C	.	.	A
cow	A	-	.	.	A
corn	A	U	.	.	A

Creating an HMM: SAM Overview



SAM Example

Build a model from 10 sequences (create `test.mod`)

```
buildmodel test -train trna10.seq
```

Produce an alignment (create `test.a2m`)

```
align2model test -modelfile test.mod -db trna10.seq
```

Make it look nice (create `test.pretty`)

prettyalign test.a2m > test.pretty

Score the sequences (create `test.dist`)

```
hmmscore test -i test.mod -db trna10.seq -sw 2
```


The model file `test.mod`

```
% SAM: buildmodel v3.2 (July 31, 2000) compiled 09/26/00_09:45:11
% (c) 1992-2000 Regents of the University of California, Santa Cruz
%        Sequence Alignment and Modeling Software System
%       http://www.cse.ucsc.edu/research/compbio/sam.html
% ----------------------------------------------------------------------
% test   Host: alpha    Tue Sep 26 11:49:58 2000
% rph    Dir:  /auto/projects/kestrel/rph/sam32/demos
% ----------------------------------------------------------------------
% Internal sequence weighting method 1
% Regularizer FIM_method_train training letter frequencies (1)
% Regularizer Insert_method_train training letter frequencies (1)
% Model initial FIM_method_train training letter frequencies (1)
% Model initial insert_method_train training letter frequencies (1)
% Generic, Insert, and FIM tables dynamically reset to
%   train_reset_inserts geometric mean of match probabilities (6)
% All models generated from regularizer.
% Sequence-model (global) (SW = 0)
%
% Model lengths:    80  76  68
% Itera-   Average
% tion     distance
%    1   108.468   106.628   105.733
%    2   105.218   104.040   105.258
%    3   104.705   103.232   105.711
%    4   105.156   102.847   106.682
%    5   105.301   102.479   104.985
%    6   104.089   102.756   105.025
%    7   103.728   101.465   103.730
%    8   103.563   102.322   103.831
%    9   102.418    99.662   103.857
%   10   101.137    99.964   102.380
%   11   101.401    99.000   102.622
```

The model file `test.mod`

```
%    13    99.984    93.130    99.436
%    14    97.497    90.376    98.053
%    15    81.217    66.553    79.458
%    16    74.898    65.075    74.750
%    17    71.717    64.372    72.674
%    18    70.804    64.035    71.138
%    19    70.194    63.854    70.361
%    20    69.751    63.769    70.090
%    21    69.566    63.769    69.977
%    22    69.468    63.769    69.878
% Model 1 (counting from 0) wins!!
% ---------------------------------------------------------------------
%  TRAIN 10 sequences (average length 74).  Distance statistics...
%     Min:-40.28 (  8) Max:-28.18 (  7) Ave:-34.41 SampDev:  3.56
%     MinSDs:  -1.65  MaxSDs:   1.75
%
%  Total CPU time:  user   0: 0:14    system   0: 0: 0
%  Finished at:  Tue Sep 26 11:50:13 2000
% ---------------------------------------------------------------------
%
% Parameters from command line and insert files:
%
% trainseed 422
% randseed 0
train ../demos/trna10.seq
a RNA
print_frequencies 1
%
% ---------------------------------------------------------------------
% ---------------------------------------------------------------------
MODEL --  Final model for run test
```

The alignment file `test.pretty`

```
;  SAM: ../src/prettyalign v3.2 (July 31, 2000) compiled 09/26/00_09:45:33
;  (c) 1992-2000 Regents of the University of California, Santa Cruz
;
;          Sequence Alignment and Modeling Software System
;         http://www.cse.ucsc.edu/research/compbio/sam.html
;
; ----------------- Citations (SAM, SAM-T99, HMMs) --------------------
; R. Hughey, A. Krogh, Hidden Markov models for sequence analysis:
;  Extension and analysis of the basic method, CABIOS 12:95-107, 1996.
; K. Karplus, C. Barrett, R. Hughey, Hidden Markov models for detecting
;  remote protein homologies, Bioinformatics 14(10):846-856, 1999.
; A. Krogh et al., Hidden Markov models in computational biology:
;  Applications to protein modeling, JMB 235:1501-1531, Feb 1994.
; ---------------------------------------------------------------------
               10             20           30          40          50          60          70
                |              |            |           |           |           |           |
TRNA1  GGGGAUGUAGCUCAG-.UGG...U.AGAGCGCAUGCUUCGCAUGUAUGAGGCCCCGGGUUCGAUCCCCGGCAUCU---CCA
TRNA2  GCGGCCGUCGUCUAGU.CUGgauU.AGGACGCUGGCCUCCCAAGCCAGCAAUCCCGGGUUCGAAUCCCGGCGGCCG---CA
TRNA3  GGCCCUGUGGC-UAGC.UGG...UcAAAGCGCCUGUCUAGUAAACAGGAGAUCCUGGGUUCGAAUCCCAGCGGGGCCUCCA
TRNA4  GGGCGAAUAGUGUCAG.CGG...G.AGCACACCAGACUUGCAAUCUGGUAGGGA-GGGUUCGAGUCCCUCUUUGUCCACCA
TRNA5  GCCGGGAUAGCUCAGU.UGG...U.AGAGCAGAGGACUGAAAAUCCUCGUGUCACCAGUUCAAAUCUGGUU---CCUGGCA
TRNA6  GGGGCCUUAGCUCAGC.UGG...G.AGAGCGCCUGCUUUGCACGCAGGAGGUCAGCGGU-CGA-CCCGCUAGGCUCCACCA
TRNA7  GGGCACAUGGCGCAGU.UGG...U.AGCGCGCUUCCCUUGCAAGGAAGAGGUCAUCGGUUCGAUUCCGGUUGCGU---CCA
TRNA8  GGGCCCGUGGCCUAGU.CUGga.U.ACGGCACCGGCCUUCUAAGCCGGGGAUCGGGGGUUCAAAUCCCUCCGGGU---CCG
TRNA9  CGGCACGUAGCGCAGCcUGG...U.AGCGCACCGUCCUGGGGUUGCGGGGGUCGGAGGUUCAAAUCCUCUCGUGCCGACCA
TRNA10 UCCGUCGUAGUCUAGG.UGGu..U.AGGAUACUCGGCUUUCACCCGAGAGA-CCCGGGUUCAAGUCCCGGCGACGGAACCA
```

The distance file `test.dist`

```
% ---------------------------------------------------------------------
% Inserted Files:  test.mod
% Database Files:  ../demos/trna10.seq
%
% Subsequence-submodel (local) (SW = 2)
% Simple scores adjusted by ln(seq len) (adjust_score = 2)
% Track 0 FIMs added (geometric mean of match probabilities (6))
% Single Track Model:  test.mod
% DP Method: forward all-paths (dpstyle = 0)
% 10 sequences, 747 residues, 77 nodes, 0.23 seconds
%
% Sequence scores selected:  All  (select_score=8)
%
% Simple: NLL-NULL using FIM probabilities
% Reverse: NLL-NULL for the reverse sequence NULL model
%    Calculated when Simple < simple_threshold (0.00)
% E-value: n_seq_scored(=10) / (1 + exp(emlambda(=1.0000) * Reverse))
%    Calculated when Simple < simple_threshold (0.00)
%    Rescale E-values or use -dbsize for multiple scoring runs.
% Scores sorted by E-value, best first
%
% Sequence ID   Length      Simple     Reverse     E-value      X count
TRNA7              73       -45.74      -38.09     2.9e-16
TRNA1              72       -41.48      -33.68     2.3e-14
TRNA8              75       -41.52      -33.27     3.6e-14
TRNA3              76       -40.14      -32.56     7.2e-14
TRNA2              76       -39.72      -32.33     9.1e-14
TRNA9              77       -38.13      -30.74     4.5e-13
TRNA5              73       -39.08      -30.23     7.4e-13
TRNA4              75       -37.26      -29.79     1.2e-12
TRNA10             76       -37.28      -29.69     1.3e-12
TRNA6              74       -38.46      -27.54     1.1e-11
```

Sequences and Subsequences

Training to find a motif

- SAM includes *Free Insertion Modules* (FIMs)
- Allow an arbitrary number of insertions at low cost
- When placed at ends of model, model can freely place its self anywhere in a sequence during training
- FIM training will find conserved motifs within a family of sequences.

FIMs violate probability

Scoring styles

- `-sw 0` Model-Sequence (global)
- `-sw 1` Model-subsequence (semilocal)
- `-sw 2` **Submodel-subsequence (local)**
- `-sw 3` Submodel-Sequence (domain)

Multiple domains

- `-selectmdalign 4` Perform multiple-domain alignment on high-scoring (by E-value) sequences
  - Iteratively look for multiple similarlity regions

Important SAM options:

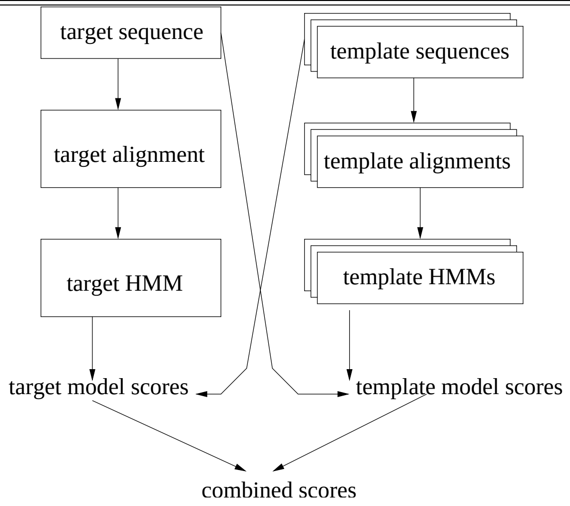
–train trainseq.seq	Specify training sequences
–i modelfile.mod	Use an existing model
–alignfile align.seq	Use a seed alignment
–priorlibrary uprior9.plib	Use a Dirichlet mixture
–db search.seq	A database for searching
–sw 2	Local scoring and alignment
–selectalign 8	Align all sequences
–selectalign 4	Align sequences with good evalue
–Emax 0.001	Evalue theshold

- HMMs are statistical multiple alignments
- HMMs use algorithms similar to S&W DP
- HMMs have many parameters to optimize
- HMMs can be used for search and alignment
- HMMs need effective regularization
- HMMs can be built from a single sequence using an iterative method like SAM-Tx

- Developed by Kevin Karplus
- Motivated by CASP Protein Structure Prediction Workshop
- Protein-only
- Geared to superfamily-level remote homolog detection
- A top method in CASP2, CASP3, and CASP4
- Not top in CASP5
  - Kevin's new fragment-packing tool, Undertaker, used to create 3-d strucutures
  - This reduced the quality of some HMM alignments
  - Lesson: Don't use CASP as your beta-test!

Overview of method for building a target HMM, given a single sequence (or a seed alignment):

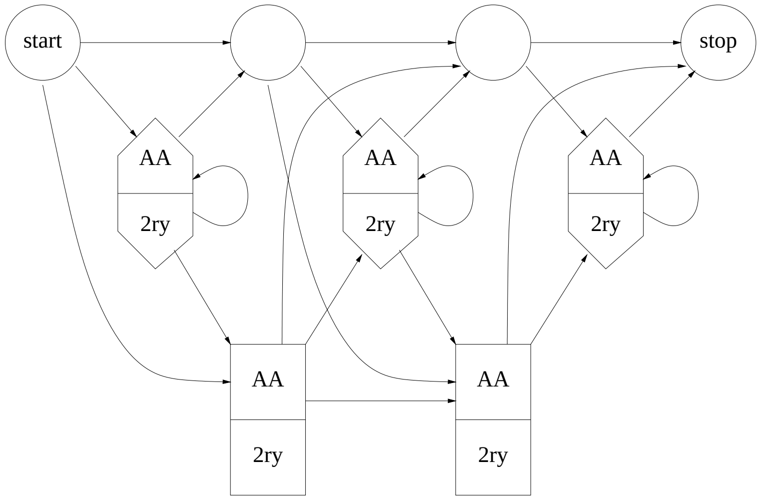
1. Construct a profile HMM with one state for each letter of sequence (or column of multiple alignment).
2. Find sequences in a large database of protein sequences that score well with M. This is the *training set*.
3. Retrain M (using forward-backward algorithm) to re-estimate all probabilites, based on the training set.
4. Make a multiple alignment (using Viterbi algorithm) of all sequences in the training set. The multiple alignment has one alignment column for each fat state of the HMM.
5. Repeat from step 1, with thresholds in step 2 loosened.

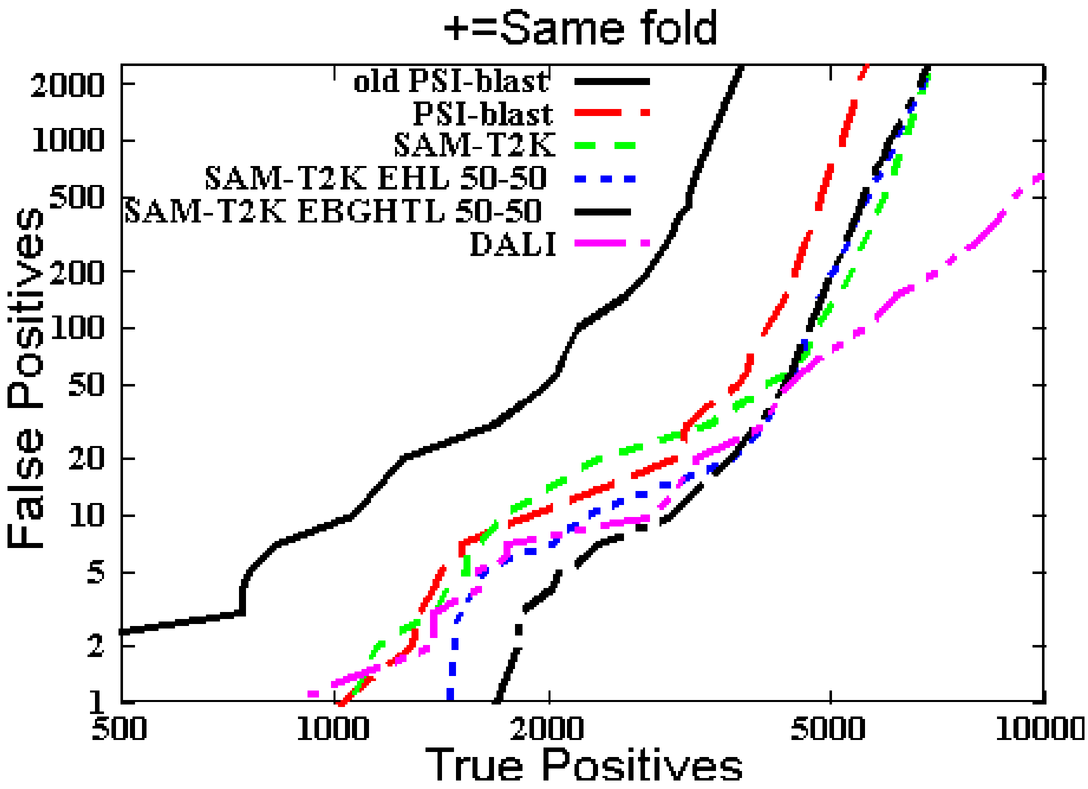


- Choose (somehow) the alignment based on the target model or the alignment based on the template model.
- This method for fold-recognition is available (with only amino-acid target HMMs, not 2-track target HMMs) on `http://www.cse.ucsc.edu/research/compbio/hmm-apps/`.
- The library currently has 5473 templates.

We can also use alignments built using a two-track target HMM:

- Amino-acid track
- Secondary-structure track (probabilities of {E, H, L} from neural net). The correct letters are defined by STRIDE.
- Can align template (AA+2ry) to target model.

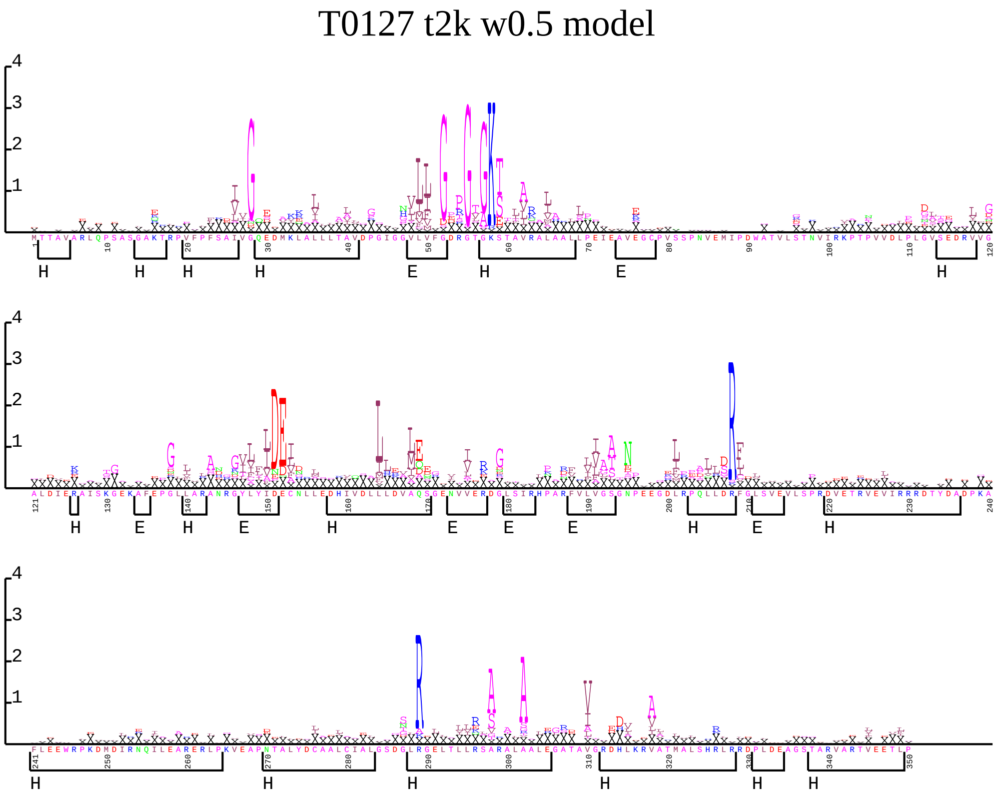




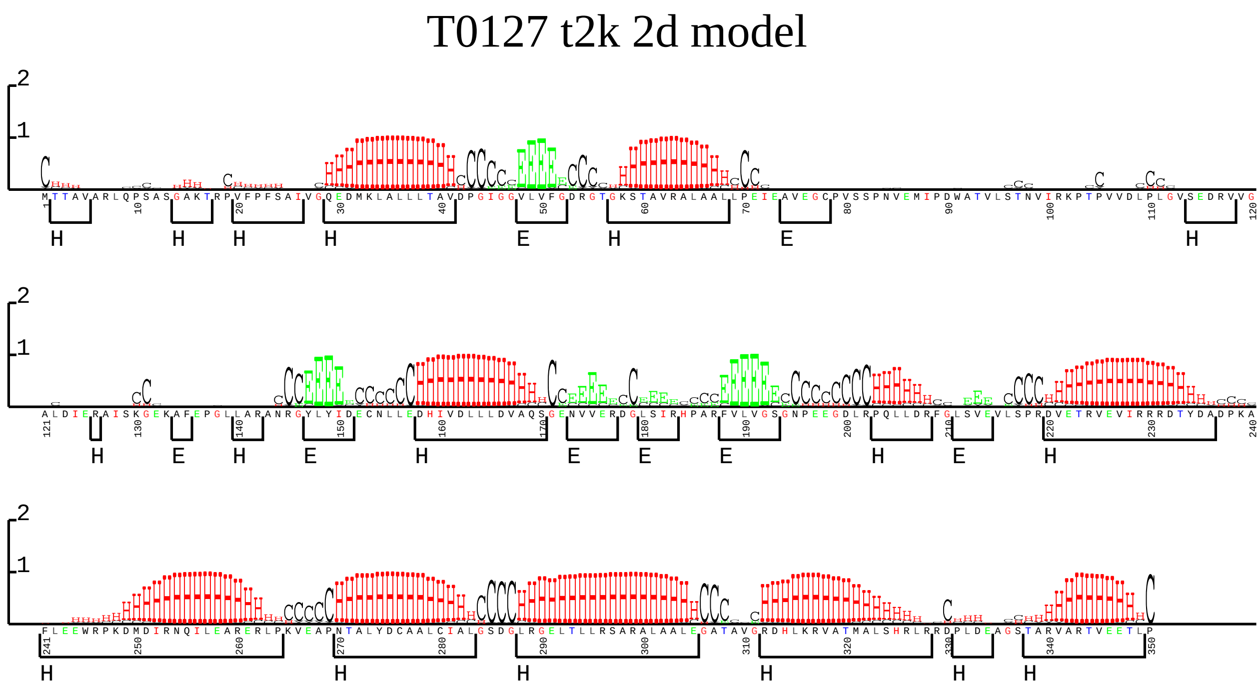
CASP4 example: T0127_2

- SCOP superfamily 3.31.1 with strong scores—like essentially all the automatic servers.
- Best scores with 1do0A (3.31.1.12.5), rather than than 1a5t (3.31.1.12.3), 1d2nA (3.31.1.12.4), 1gky (3.31.1.1.1), or 1nipA (3.31.1.9.3).
- Preference held for both target and template alignments, and both amino-acid only and two-track.
- Second domain, residues 267-350, was better aligned than by other CASP4 predictors.
- Alignment essentially 2-track local alignment. Moved first segment over 1 residue and pulled in final helix from off the end of alignment, based on residue conservation.

CASP4: T0127 sequence logo

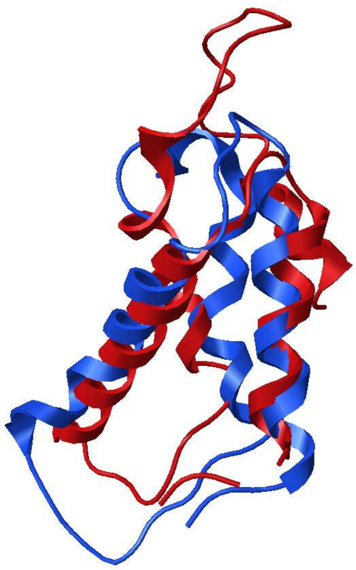


CASP4: T0127 predicted 2ry sequence logo



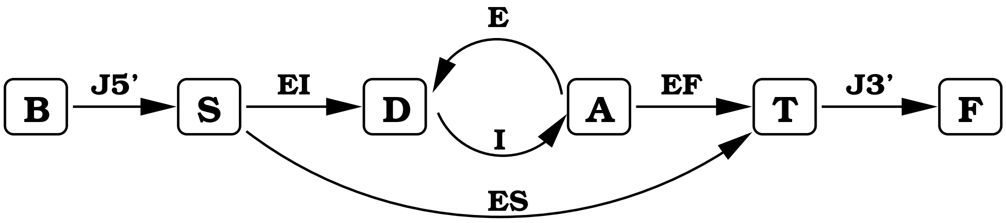
T0127 domain 2 comparison

Predicted=blue solved=red.



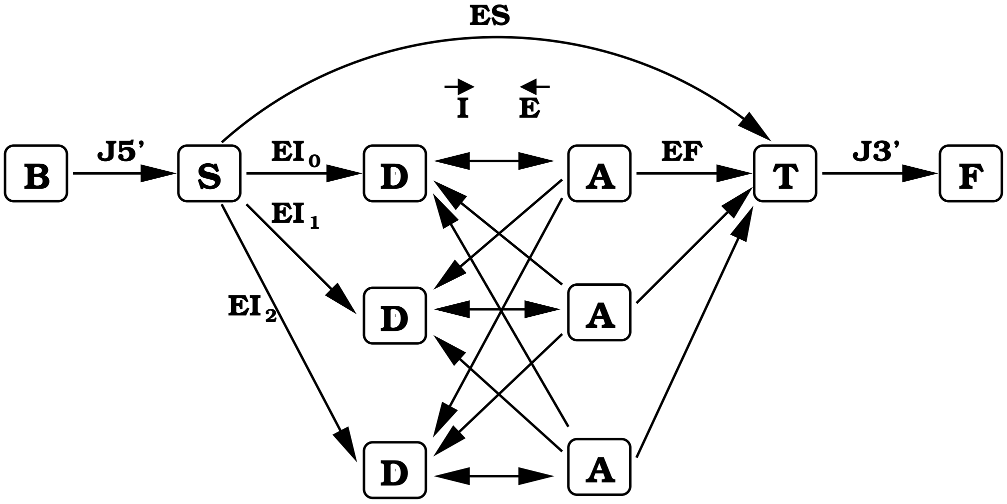
HMMs can find genes, too

- Gene components include:
  - A start signal
  - Initial untranslated region
  - Exons (coding) and introns (non-coding)
  - A stop signal



This are more complicated because of reading frames

- 3 nucleotides = 1 amino acid
- Intron/Exon boundries are in the nucleotide sequence



UCSC bioinformatics info: http://www.cse.ucsc.edu/research/compbio/

SAM tool suite info: http://www.cse.ucsc.edu/research/compbio/sam.html

HMM **servers:** http://www.cse.ucsc.edu/research/compbio/hmm-apps/

SAM-T2K prediction server: http://www.cse.ucsc.edu/research/compbio/hmm-apps/T02-query.html

Superfamily Server: http://stash.mrc-lmb.cam.ac.uk/SUPERFAMILY/