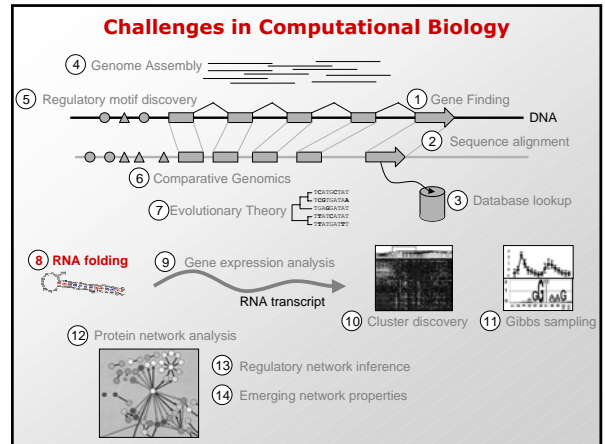
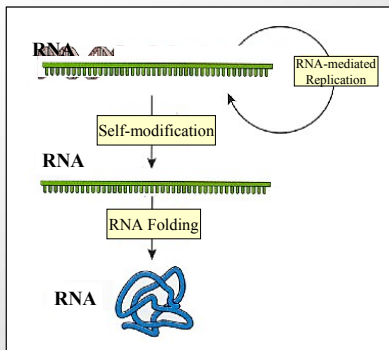


## RNA secondary structure

- Lecture 1 - Introduction
- Lecture 2 - Hashing and BLAST
- Lecture 3 - Combinatorial Motif Finding
- Lecture 4 - Statistical Motif Finding
- Lecture 5 - Sequence alignment and Dynamic Programming



## The world before DNA or Protein



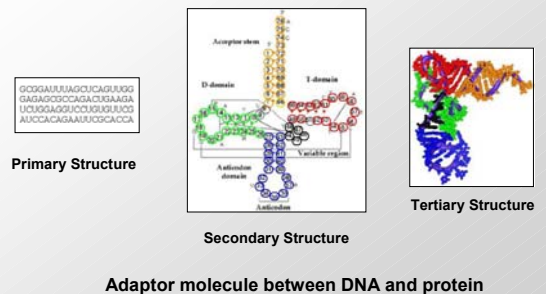
## RNA World

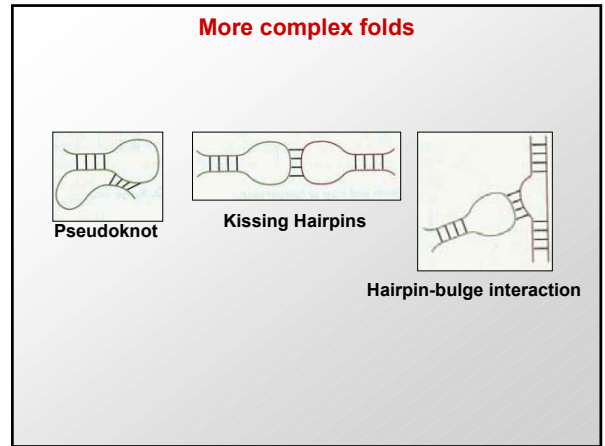
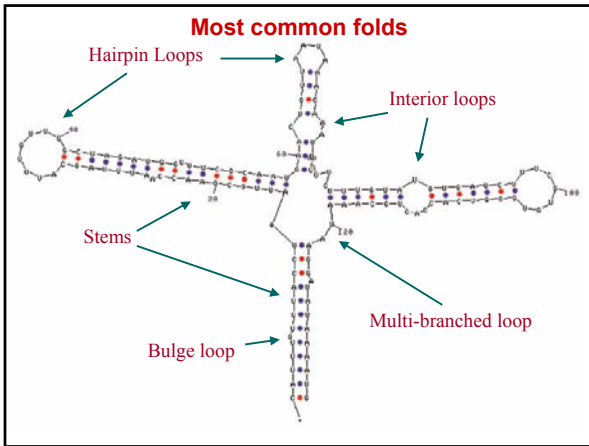
- RNA can be protein-like
  - Ribozymes can catalyze enzymatic reactions by RNA secondary fold
  - Small RNAs can play structural roles within the cell
  - Small RNAs play versatile roles in gene regulatory
- RNA can be DNA-like
  - Made of digital information, can transfer to progeny by complementarity
  - Viruses with RNA genomes (single/double stranded)
  - RNA can catalyze RNA replication
- RNA world is possible
  - Proteins are more efficient (larger alphabet)
  - DNA is more stable (double helix, less flexible)

## RNA invented its successors

- RNA invents protein
  - Ribosome precise structure was solved this past year
  - Core is all RNA. Only RNA makes DNA contact
  - Protein component only adds structural stability
- RNA and protein invent DNA
  - Stable, protected, specialized structure (no catalysis)
  - Proteins catalyze: RNA → DNA reverse transcription
  - Proteins catalyze: DNA → DNA replication
  - Proteins catalyze: DNA → RNA transcription
- Viruses still preserved from those early days of life
  - Any type genome: dsDNA, ssRNA, dsRNA, hybrid
  - Simplest self-replicating life form

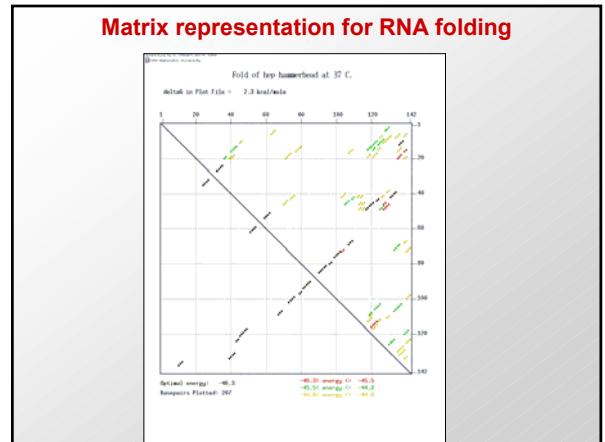
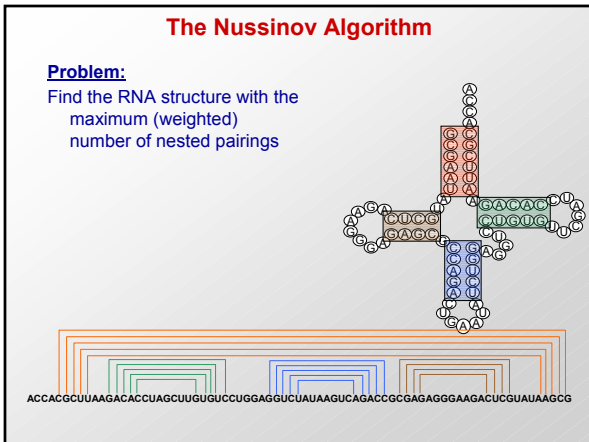
## Example: tRNA secondary and tertiary structure





### Dynamic programming algorithm for secondary structure determination

- ### First DP Algorithm: Nussinov
- one possible technique: base pair maximization
  - Algorithms for Loop Matching (Nussinov et al., 1978)
  - too simple for accurate prediction, but stepping-stone for later algorithms



## The Nussinov Algorithm

Given sequence  $X = x_1 \dots x_N$ ,  
Define DP matrix:

$F(i, j)$  = maximum number of bonds if  $x_i \dots x_j$  folds optimally

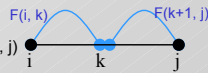
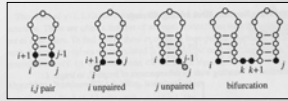
Two cases, if  $i < j$ :

1.  $x_i$  is paired with  $x_j$

$$F(i, j) = s(x_i, x_j) + F(i+1, j-1)$$

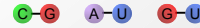
•  $x_i$  is not paired with  $x_j$

$$F(i, j) = \max\{k: i \leq k < j\} F(i, k) + F(k+1, j)$$



## Initial Concepts

- only consider base pairs



- folding of an  $N$  nucleotide sequence can be specified by a symmetric  $N \times N$  matrix
- $M_{ij} = 1$  if bases form a pair
- $M_{ij} = 0$  otherwise

## Naïve Example 1

|   |   | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
|   | G | G | G | A | A | A | U | C | C |   |
| 1 | G | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 2 | G | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 3 | G | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 4 | A | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 5 | A | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 6 | A | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 7 | U | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 8 | C | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | C | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |



## Matching "blocks"

- visually inspect matrices for diagonal lines of 1's
- manually piece them together into an optimal folded shape

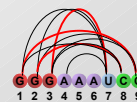
## Naïve Example 1

|   |   | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
|   | G | G | G | A | A | A | U | C | C |   |
| 1 | G | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 2 | G | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 3 | G | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 4 | A | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 5 | A | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 6 | A | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 7 | U | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 8 | C | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | C | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |



## Naïve Example 1

|   |   | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
|   | G | G | G | A | A | A | U | C | C |   |
| 1 | G | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 2 | G | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 3 | G | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 4 | A | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 5 | A | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 6 | A | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 7 | U | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 8 | C | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | C | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |



## Naïve Example 1

|   |   | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
|   | G | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 1 | G | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 2 | G | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 3 | G | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 4 | A | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 5 | A | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 6 | A | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 7 | U | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 8 | C | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | C | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |



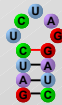
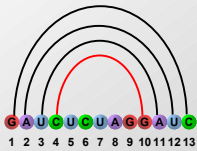
## Refinement

- unfortunately, this finds chemically infeasible structures
- i.e. insufficient space, inflexibility of paired base regions
- next step is to specify better constraints
- solution: a dynamic programming algorithm [Nussinov et al., 1978]

## Structure Representation

- secondary structure described as a graph
- base pairs are described via pairs of indices  $(i, j)$ , indicating links between base vertices

$S = \{(1,13), (2,12), (3,11), (4,10)\}$

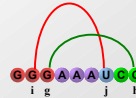


## Basic Constraints

1. Each edge contains vertices (bases) linking compatible base pairs
2. No vertex can be in more than one edge
3. Edges must be drawn without crossing

Edges  $(g, h)$  and  $(i, j)$

if  $i < g < j < h$  or  $g < i < h < j$ , both edges cannot belong to the same "matching."



## Basic Constraints

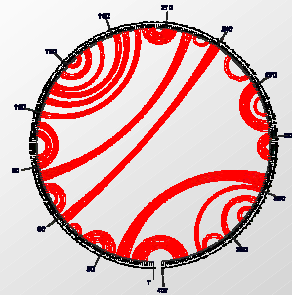
1. Each edge contains vertices (bases) linking compatible base pairs
2. No vertex can be in more than one edge
3. Edges must be drawn without crossing

Edges  $(g, h)$  and  $(i, j)$

if  $i < g < j < h$  or  $g < i < h < j$ , both edges cannot belong to the same "matching."



## Circular Representation



ENERGY -- 45.7 *Escherichia coli* P RNA P RNA

Image source: Zuker, M. (2002) "Lectures on RNA Secondary Structure Prediction" <http://www.bioinfo.rpi.edu/~zukerm/lectures/RNAfold.html/node1.html>

Courtesy of Michael Zuker. Used with permission.

## Energy Minimization

- objective is a folded shape for a given nucleotide chain such that the energy is minimized
- $E_{ij} = l$  for each possible compatible base pair,  $E_{ij} = 0$  otherwise

## The Nussinov Algorithm

### Initialization:

$F(i, i-1) = 0;$  for  $i = 2$  to  $N$   
 $F(i, i) = 0;$  for  $i = 1$  to  $N$

### Iteration:

For  $i = 2$  to  $N$ :

For  $i = 1$  to  $N - i$

$j = i + i - 1$

$$F(i, j) = \max \begin{cases} F(i+1, j-1) + s(x_i, x_j) \\ \max_{i \leq k < j} \{ F(i, k) + F(k+1, j) \} \end{cases}$$

### Termination:

Best structure is given by  $F(1, N)$   
(Need to trace back)

## Algorithm Behavior

- recursive computation, finding the best structure for small subsequences
- works outward to larger subsequences
- four possible ways to get the best RNA structure:

## Case 1: Adding unpaired base $i$

- Add unpaired position  $i$  onto best structure for subsequence  $i+1, j$

Image removed due to copyright considerations.

Please see:

Durbin, Richard, et. al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*  
Cambridge, UK: Cambridge University Press, 1999. ISBN: 0521629713.

## Case 2: Adding unpaired base $j$

- Add unpaired position  $i$  onto best structure for subsequence  $i+1, j$

Image removed due to copyright considerations.

Please see:

Durbin, Richard, et. al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*  
Cambridge, UK: Cambridge University Press, 1999. ISBN: 0521629713.

## Case 3: Adding $(i, j)$ pair

- Add base pair  $(i, j)$  onto best structure found for subsequence  $i+1, j-1$

Image removed due to copyright considerations.

Please see:

Durbin, Richard, et. al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*  
Cambridge, UK: Cambridge University Press, 1999. ISBN: 0521629713.

### Case 4: Bifurcation

- combining two optimal substructures  $i, k$  and  $k+1, j$

Image removed due to copyright considerations.

Please see:

Durbin, Richard, et. al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge, UK: Cambridge University Press, 1999. ISBN: 0521629713.

### Nussinov RNA Folding Algorithm

- Initialization:

$$\begin{aligned} \gamma(i, i-1) &= 0 && \text{for } i = 2 \text{ to } L; \\ \gamma(i, i) &= 0 && \text{for } i = 2 \text{ to } L. \end{aligned}$$

Image removed due to copyright considerations.

Please see:

Durbin, Richard, et. al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge, UK: Cambridge University Press, 1999. ISBN: 0521629713.

### Nussinov RNA Folding Algorithm

- Initialization:

$$\begin{aligned} \gamma(i, i-1) &= 0 && \text{for } i = 2 \text{ to } L; \\ \gamma(i, i) &= 0 && \text{for } i = 2 \text{ to } L. \end{aligned}$$

Image removed due to copyright considerations.

Please see:

Durbin, Richard, et. al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge, UK: Cambridge University Press, 1999. ISBN: 0521629713.

### Nussinov RNA Folding Algorithm

- Initialization:

$$\begin{aligned} \gamma(i, i-1) &= 0 && \text{for } i = 2 \text{ to } L; \\ \gamma(i, i) &= 0 && \text{for } i = 2 \text{ to } L. \end{aligned}$$

Image removed due to copyright considerations.

Please see:

Durbin, Richard, et. al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge, UK: Cambridge University Press, 1999. ISBN: 0521629713.

### Nussinov RNA Folding Algorithm

- Recursive Relation:
- For all subsequences from length 2 to length L:

$$\gamma(i, j) = \max \begin{cases} \gamma(i+1, j) & \text{Case 1} \\ \gamma(i, j-1) & \text{Case 2} \\ \gamma(i+1, j-1) + \delta(i, j) & \text{Case 3} \\ \max_{i < k < j} [\gamma(i, k) + \gamma(k+1, j)] & \text{Case 4} \end{cases}$$

### Nussinov RNA Folding Algorithm

$$\gamma(i, j) = \max \begin{cases} \gamma(i+1, j) \\ \gamma(i, j-1) \\ \gamma(i+1, j-1) + \delta(i, j) \\ \max_{i < k < j} [\gamma(i, k) + \gamma(k+1, j)] \end{cases}$$

Image removed due to copyright considerations.

Please see:

Durbin, Richard, et. al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge, UK: Cambridge University Press, 1999. ISBN: 0521629713.

### Nussinov RNA Folding Algorithm

$$\gamma(i, j) = \max \begin{cases} \gamma(i+1, j) \\ \gamma(i, j-1) \\ \gamma(i+1, j-1) + \delta(i, j) \\ \max_{i < k < j} [\gamma(i, k) + \gamma(k+1, j)] \end{cases}$$

Image removed due to copyright considerations.

Please see:

Durbin, Richard, et al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge, UK: Cambridge University Press, 1999. ISBN: 0521629713.

### Nussinov RNA Folding Algorithm

$$\gamma(i, j) = \max \begin{cases} \gamma(i+1, j) \\ \gamma(i, j-1) \\ \gamma(i+1, j-1) + \delta(i, j) \\ \max_{i < k < j} [\gamma(i, k) + \gamma(k+1, j)] \end{cases}$$

Image removed due to copyright considerations.

Please see:

Durbin, Richard, et al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge, UK: Cambridge University Press, 1999. ISBN: 0521629713.

### Example Computation

$$\rightarrow \gamma(4, 7) = \max \begin{cases} \gamma(5, 7) \\ \gamma(4, 6) \\ \gamma(5, 6) + \delta(4, 7) \\ \max_{4 < k < 7} [\gamma(4, k) + \gamma(k+1, 7)] \end{cases}$$

Image removed due to copyright considerations.

Please see:

Durbin, Richard, et al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge, UK: Cambridge University Press, 1999. ISBN: 0521629713.

### Example Computation


$$\gamma(4, 7) = \max \begin{cases} \gamma(5, 7) \leftarrow \\ \gamma(4, 6) \\ \gamma(5, 6) + \delta(4, 7) \\ \max_{4 < k < 7} [\gamma(4, k) + \gamma(k+1, 7)] \end{cases}$$


Image removed due to copyright considerations.

Please see:

Durbin, Richard, et al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge, UK: Cambridge University Press, 1999. ISBN: 0521629713.

### Example Computation


$$\gamma(4, 7) = \max \begin{cases} \gamma(5, 7) \\ \gamma(4, 6) \leftarrow \\ \gamma(5, 6) + \delta(4, 7) \\ \max_{4 < k < 7} [\gamma(4, k) + \gamma(k+1, 7)] \end{cases}$$


Image removed due to copyright considerations.

Please see:

Durbin, Richard, et al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge, UK: Cambridge University Press, 1999. ISBN: 0521629713.

### Example Computation


$$\gamma(4, 7) = \max \begin{cases} \gamma(5, 7) \\ \gamma(4, 6) \\ \gamma(5, 6) + \delta(4, 7) \leftarrow \\ \max_{4 < k < 7} [\gamma(4, k) + \gamma(k+1, 7)] \end{cases}$$


Image removed due to copyright considerations.

Please see:

Durbin, Richard, et al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge, UK: Cambridge University Press, 1999. ISBN: 0521629713.

### Example Computation


$$\gamma(4,7) = \max \begin{cases} \gamma(5,7) \\ \gamma(4,6) \\ \gamma(5,6) + \delta(4,7) \\ \max_{k < k < 7} [\gamma(4,k) + \gamma(k+1,7)] \end{cases}$$


Image removed due to copyright considerations.

Please see:

Durbin, Richard, et al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge, UK: Cambridge University Press, 1999. ISBN: 0521629713.

### Example Computation

$$\gamma(4,7) = \max \begin{cases} \gamma(5,7) \\ \gamma(4,6) \\ \gamma(5,6) + \delta(4,7) \\ \max_{k < k < 7} [\gamma(4,k) + \gamma(k+1,7)] \end{cases}$$

Image removed due to copyright considerations.

Please see:

Durbin, Richard, et al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge, UK: Cambridge University Press, 1999. ISBN: 0521629713.

### Completed Matrix

$$\gamma(i,j) = \max \begin{cases} \gamma(i+1,j) \\ \gamma(i,j-1) \\ \gamma(i+1,j-1) + \delta(i,j) \\ \max_{i < k < j} [\gamma(i,k) + \gamma(k+1,j)] \end{cases}$$

Image removed due to copyright considerations.

Please see:

Durbin, Richard, et al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge, UK: Cambridge University Press, 1999. ISBN: 0521629713.

### Traceback

- value at  $\gamma(l, L)$  is the total base pair count in the maximally base-paired structure
- as in other DP, traceback from  $\gamma(l, L)$  is necessary to recover the final secondary structure
- pushdown stack is used to deal with bifurcated structures

### Traceback Pseudocode

Initialization: Push  $(l, L)$  onto stack

Recursion: Repeat until stack is empty:

- pop  $(i, j)$ .
- If  $i \geq j$  continue; // hit diagonal
  - else if  $\gamma(i+1, j) = \gamma(i, j)$  push  $(i+1, j)$ ; // case 1
  - else if  $\gamma(i, j-1) = \gamma(i, j)$  push  $(i, j-1)$ ; // case 2
  - else if  $\gamma(i+1, j-1) + \delta_{ij} = \gamma(i, j)$ ; // case 3
    - record  $i, j$  base pair
    - push  $(i+1, j-1)$ ;
- else for  $k=i+1$  to  $j-1$ : if  $\gamma(i, k) + \gamma(k+1, j) = \gamma(i, j)$ ; // case 4
  - push  $(k+1, j)$ .
  - push  $(i, k)$ .
  - break

### Retrieving the Structure



Image removed due to copyright considerations.

Please see:

Durbin, Richard, et al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge, UK: Cambridge University Press, 1999. ISBN: 0521629713.



## Retrieving the Structure

| PAIRS | STACK | CURRENT |
|-------|-------|---------|
|       | (2,9) | (1,9)   |

Image removed due to copyright considerations.

Please see:

Durbin, Richard, et. al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge, UK: Cambridge University Press, 1999. ISBN: 0521629713.

## Retrieving the Structure

| PAIRS | STACK | CURRENT |
|-------|-------|---------|
| (2,9) | (3,8) | (2,9)   |



Image removed due to copyright considerations.

Please see:

Durbin, Richard, et. al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge, UK: Cambridge University Press, 1999. ISBN: 0521629713.

## Retrieving the Structure

| PAIRS | STACK | CURRENT |
|-------|-------|---------|
| (2,9) | (4,7) | (3,8)   |
| (3,8) |       |         |




Image removed due to copyright considerations.

Please see:

Durbin, Richard, et. al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge, UK: Cambridge University Press, 1999. ISBN: 0521629713.

## Retrieving the Structure

| PAIRS | STACK | CURRENT |
|-------|-------|---------|
| (2,9) | (5,6) | (4,7)   |
| (3,8) |       |         |
| (4,7) |       |         |




Image removed due to copyright considerations.

Please see:

Durbin, Richard, et. al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge, UK: Cambridge University Press, 1999. ISBN: 0521629713.

## Retrieving the Structure

| PAIRS | STACK | CURRENT |
|-------|-------|---------|
| (2,9) | (6,6) | (5,6)   |
| (3,8) |       |         |
| (4,7) |       |         |

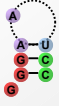


Image removed due to copyright considerations.

Please see:

Durbin, Richard, et. al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge, UK: Cambridge University Press, 1999. ISBN: 0521629713.

## Retrieving the Structure

| PAIRS | STACK | CURRENT |
|-------|-------|---------|
| (2,9) | -     | (6,6)   |
| (3,8) |       |         |
| (4,7) |       |         |




Image removed due to copyright considerations.

Please see:

Durbin, Richard, et. al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge, UK: Cambridge University Press, 1999. ISBN: 0521629713.

## Retrieving the Structure



Image removed due to copyright considerations.

Please see:

Durbin, Richard, et. al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge, UK: Cambridge University Press, 1999. ISBN: 0521629713.

## Evaluation of Nussinov

- unfortunately, while this does maximize the base pairs, it does not create viable secondary structures
- in Zuker's algorithm, the correct structure is assumed to have the lowest equilibrium free energy ( $\Delta G$ ) (Zuker and Stiegler, 1981; Zuker 1989a)

## Minimizing free energy

## The Zuker algorithm – main ideas

Models energy of an RNA fold

1. Instead of base pairs, pairs of base pairs (more accurate)
2. Separate score for bulges
3. Separate score for different-size & composition loops
4. Separate score for interactions between stem & beginning of loop

Can also do all that with a SCFG, and train it on real data

## Free Energy ( $\Delta G$ )

- $\Delta G$  approximated as the sum of contributions from loops, base pairs and other secondary structures

Image removed due to copyright considerations.

Please see:

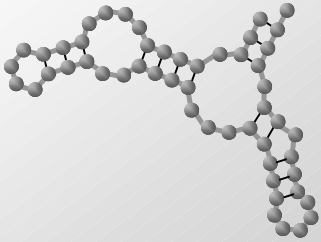
Durbin, Richard, et. al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge, UK: Cambridge University Press, 1999. ISBN: 0521629713.

## Basic Notation

- secondary structure of sequence  $s$  is a set  $S$  of base pairs  $i \cdot j$ ,  $1 \leq i < j \leq |s|$
- we assume:
  - each base is only in one base pair
  - no pseudoknots
  - sharp "U-turns" prohibited; a hairpin loop must contain at least 3 bases

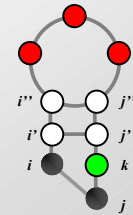
## Secondary Structure Representation

- can view a structure  $S$  as a collection of loops together with some external unpaired bases



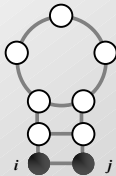
## Accessible Bases

- Let  $i < k < j$  with  $i \cdot j \in S$
- $k$  is *accessible* from  $i \cdot j$  if for all  $i' \cdot j' \in S$  if it is not the case that  $i < i' < k < j' < j$



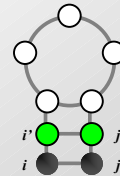
## Exterior Base Pairs

- base pair  $i \cdot j$  is the exterior base pair of (or closing) the loop consisting of  $i \cdot j$  and all bases *accessible* from it



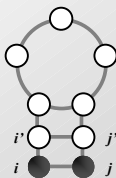
## Interior Base Pairs

- if  $i'$  and  $j'$  are accessible from  $i \cdot j$
- and  $i' \cdot j' \in S$
- then  $i' \cdot j'$  is an interior base pair, and is accessible from  $i \cdot j$



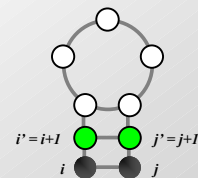
## Hairpin Loop

- if there are no interior base pairs in a loop, it is a hairpin loop



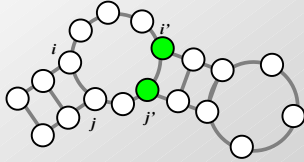
## Stacked Pair

- a loop with one interior base pair is a stacked pair if  $i' = i+1$  and  $j' = j-1$



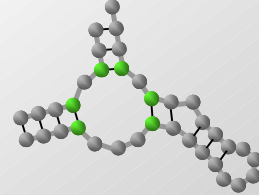
### Internal Loop

- if it is not true that the interior base pair  $i \cdot j$  that  $i' = i+1$  and  $j' = j-1$ , it is an internal loop



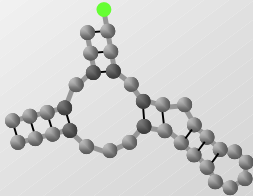
### Multibranch Loops

- loops with more than one interior base pair are multibranch loops



### External Bases and Base Pairs

- any bases or base pairs not accessible from any base pair are called external



### Assumptions

- structure prediction determines the most stable structure for a given sequence
- stability of a structure is based on free energy
- energy of secondary structures is the sum of independent loop energies

### Recursion Relation

- four arrays are used to hold the minimal free energy of specific structures of subsequences of  $s$
- arrays are computed interdependently
- calculated recursively using pre-specified free energy functions for each type of loop

### $V(i,j)$

- energy of an optimal structure of subsequence  $i$  through  $j$  closed by  $i \cdot j$ :

$$V(i, j) = \min \begin{cases} eH(i, j) \\ eS(i, j) + V(i+1, j-1) \\ VBI(i, j) \\ VM(i, j) \end{cases}$$



## Multiple alignment and RNA folding

Given K homologous aligned RNA sequences:



If  $i^{\text{th}}$  and  $j^{\text{th}}$  positions are always base paired and covary, then they are likely to be paired

## Mutual information

$$M_{ij} = \sum_{a,b \in \{a,c,g,u\}} f_{ab}(i,j) \log_2 \frac{f_{ab}(i,j)}{f_a(i) f_b(j)}$$

Where  $f_{ab}(i,j)$  is the # of times the pair a, b are in positions i, j

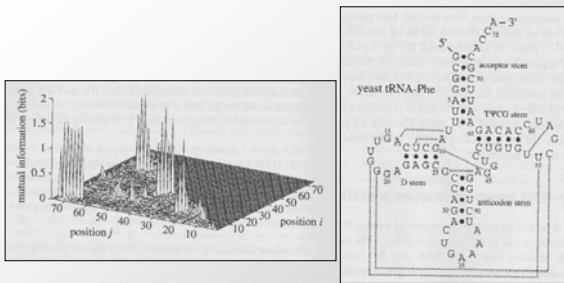
Given a multiple alignment, can infer structure that maximizes the sum of mutual information, by DP

### In practice:

1. Get multiple alignment
2. Find covarying bases – deduce structure
3. Improve multiple alignment (by hand)
4. Go to 2

A manual EM process!!

## Results for tRNA



- Matrix of co-variations in tRNA molecule

## Context Free Grammars for representing RNA folds

### A Context Free Grammar

$S \rightarrow AB$  Nonterminals: S, A, B  
 $A \rightarrow aAc \mid a$  Terminals: a, b, c, d  
 $B \rightarrow bBd \mid b$

### Derivation:

$S \rightarrow AB \rightarrow aAcB \rightarrow \dots \rightarrow aaaaccbB \rightarrow aaaaccbbBd \rightarrow \dots \rightarrow aaaaccbbbbbddd$

Produces all strings  $a^{i+1}cb^{j+1}d$ , for  $i, j \geq 0$

### Example: modeling a stem loop

$S \rightarrow a W_1 u$   
 $W_1 \rightarrow c W_2 g$   
 $W_2 \rightarrow g W_3 c$   
 $W_3 \rightarrow g L c$   
 $L \rightarrow agucg$



What if the stem loop can have other letters in place of the ones shown?

### Example: modeling a stem loop

$S \rightarrow aW_1u \quad | \quad gW_1u$   
 $W_1 \rightarrow cW_2g$   
 $W_2 \rightarrow gW_3c \quad | \quad gW_3u$   
 $W_3 \rightarrow gLc \quad | \quad aLu$   
 $L \rightarrow agucg \quad | \quad agccg \quad | \quad cugucg$

ACGG AG  
 UGCC CG

More general: Any 4-long stem, 3-5-long loop:

$S \rightarrow aW_1u \quad | \quad gW_1u \quad | \quad gW_1c \quad | \quad cW_1g \quad | \quad uW_1g \quad | \quad uW_1a$   
 $W_1 \rightarrow aW_2u \quad | \quad gW_2u \quad | \quad gW_2c \quad | \quad cW_2g \quad | \quad uW_2g \quad | \quad uW_2a$   
 $W_2 \rightarrow aW_3u \quad | \quad gW_3u \quad | \quad gW_3c \quad | \quad cW_3g \quad | \quad uW_3g \quad | \quad uW_3a$   
 $W_3 \rightarrow aLu \quad | \quad gLu \quad | \quad gLc \quad | \quad cLg \quad | \quad uLg \quad | \quad uLa$

GCGA AG  
 UGCU CG

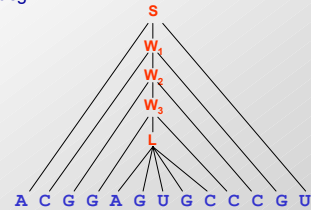
$L \rightarrow aL_1 \quad | \quad cL_1 \quad | \quad gL_1 \quad | \quad uL_1$   
 $L_1 \rightarrow aL_2 \quad | \quad cL_2 \quad | \quad gL_2 \quad | \quad uL_2$   
 $L_2 \rightarrow a \quad | \quad c \quad | \quad g \quad | \quad u \quad | \quad aa \quad | \quad \dots \quad | \quad uu \quad | \quad aaa \quad | \quad \dots \quad | \quad uuu$

GCGA CUG  
 UGUU CG

### A parse tree: alignment of CFG to sequence

- $S \rightarrow aW_1u$
- $W_1 \rightarrow cW_2g$
- $W_2 \rightarrow gW_3c$
- $W_3 \rightarrow gLc$
- $L \rightarrow agucg$

ACGG AG  
 UGCC CG



### Alignment scores for parses

We can define each rule  $X \rightarrow s$ , where  $s$  is a string, to have a score.

#### Example:

$W \rightarrow aW'u: 3$  (forms 3 hydrogen bonds)  
 $W \rightarrow gW'c: 2$  (forms 2 hydrogen bonds)  
 $W \rightarrow gW'u: 1$  (forms 1 hydrogen bond)  
 $W \rightarrow xW'z: -1$ , when  $(x, z)$  is not an  $a/u, g/c, g/u$  pair

#### Questions:

- How do we best align a CFG to a sequence? (DP)
- How do we set the parameters? (Stochastic CFGs)

### The Nussinov Algorithm and CFGs

Define the following grammar, with scores:

$S \rightarrow aSu: 3 \quad | \quad uSa: 3$   
 $gSc: 2 \quad | \quad cSg: 2$   
 $gSu: 1 \quad | \quad uSg: 1$

$SS: 0$

$aS: 0 \quad | \quad cS: 0 \quad | \quad gS: 0 \quad | \quad uS: 0 \quad | \quad \epsilon: 0$

**Note:**  $\epsilon$  is the "" string

Then, the Nussinov algorithm finds the optimal parse of a string with this grammar

### Reformulating the Nussinov Algorithm

#### Initialization:

$F(i, i-1) = 0;$  for  $i = 2$  to  $N$   
 $F(i, i) = 0;$  for  $i = 1$  to  $N$   $S \rightarrow a \quad | \quad c \quad | \quad g \quad | \quad u$

#### Iteration:

For  $i = 2$  to  $N$ :  
   For  $i = 1$  to  $N - i$ :  
      $j = i + i - 1$   
      $F(i, j) = \max \left\{ \begin{array}{l} F(i+1, j-1) + s(x_i, x_j) \quad S \rightarrow a \quad S \quad u \quad | \quad \dots \\ \max_{\{i \leq k < j\}} \{ F(i, k) + F(k+1, j) \} \quad S \rightarrow S \quad S \end{array} \right.$

#### Termination:

Best structure is given by  $F(1, N)$

### Stochastic Context Free Grammars

## Stochastic Context Free Grammars

In an analogy to HMMs, we can assign probabilities to transitions:

Given grammar

$$X_1 \rightarrow s_{11} \mid \dots \mid s_{1n}$$

...

$$X_m \rightarrow s_{m1} \mid \dots \mid s_{mn}$$

Can assign probability to each rule, s.t.

$$P(X_1 \rightarrow s_{11}) + \dots + P(X_1 \rightarrow s_{1n}) = 1$$

## Computational Problems

- Calculate an optimal alignment of a sequence and a SCFG  
(DECODING)
- Calculate Prob[ sequence | grammar ]  
(EVALUATION)
- Given a set of sequences, estimate parameters of a SCFG  
(LEARNING)

## Normal Forms for CFGs

Chomsky Normal Form:

$$X \rightarrow YZ$$

$$X \rightarrow a$$

All productions are either to 2 nonterminals, or to 1 terminal

### Theorem (technical)

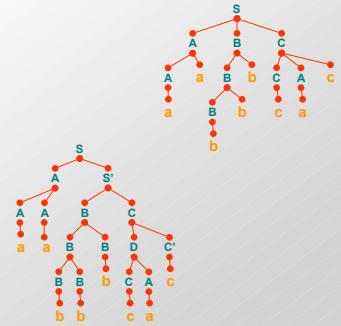
Every CFG has an equivalent one in Chomsky Normal Form

(That is, the grammar in normal form produces exactly the same set of strings)

## Example of converting a CFG to C.N.F.

$$\begin{aligned} S &\rightarrow ABC \\ A &\rightarrow Aa \mid a \\ B &\rightarrow Bb \mid b \\ C &\rightarrow CAc \mid c \end{aligned}$$

Converting:

$$\begin{aligned} S &\rightarrow AS' \\ S' &\rightarrow BC \\ A &\rightarrow AA \mid a \\ B &\rightarrow BB \mid b \\ C &\rightarrow DC' \mid c \\ C' &\rightarrow c \\ D &\rightarrow CA \end{aligned}$$


## Another example

$$\begin{aligned} S &\rightarrow ABC \\ A &\rightarrow C \mid aA \\ B &\rightarrow bB \mid b \\ C &\rightarrow cCd \mid c \end{aligned}$$

Converting:

$$\begin{aligned} S &\rightarrow AS' \\ S' &\rightarrow BC \\ A &\rightarrow C'C'' \mid c \mid A'A \\ A' &\rightarrow a \\ B &\rightarrow B'B \mid b \\ B' &\rightarrow b \\ C &\rightarrow C'C'' \mid c \\ C' &\rightarrow c \\ C'' &\rightarrow CD \\ D &\rightarrow d \end{aligned}$$

## Algorithms for learning Grammars



## Decoding: the CYK algorithm

Given  $x = x_1 \dots x_N$ , and a SCFG  $G$ ,

Find the most likely parse of  $x$   
(the most likely alignment of  $G$  to  $x$ )

Dynamic programming variable:

$\gamma(i, j, V)$ : likelihood of the most likely parse of  $x_i \dots x_j$ ,  
rooted at nonterminal  $V$

Then,

$\gamma(1, N, S)$ : likelihood of the most likely parse of  $x$  by the  
grammar

## The CYK algorithm (Cocke-Younger-Kasami)

### Initialization:

For  $i = 1$  to  $N$ , any nonterminal  $V$ ,  
 $\gamma(i, i, V) = \log P(V \rightarrow x_i)$

### Iteration:

For  $i = 1$  to  $N-1$   
For  $j = i+1$  to  $N$   
For any nonterminal  $V$ ,

$$\gamma(i, j, V) = \max_x \max_y \max_{k \leq i < j} \gamma(i, k, X) + \gamma(k+1, j, Y) + \log P(V \rightarrow XY)$$

### Termination:

$$\log P(x | \theta, \pi^*) = \gamma(1, N, S)$$

Where  $\pi^*$  is the optimal parse tree (if traced back appropriately from above)

## A SCFG for predicting RNA structure

$S \rightarrow aS \mid cS \mid gS \mid uS \mid \epsilon$   
 $\rightarrow Sa \mid Sc \mid Sg \mid Su$   
 $\rightarrow aSu \mid cSg \mid gSu \mid uSg \mid gSc \mid uSa$   
 $\rightarrow SS$

- Adjust the probability parameters to reflect bond strength etc
- No distinction between non-paired bases, bulges, loops
- Can modify to model these events
  - L: loop nonterminal
  - H: hairpin nonterminal
  - B: bulge nonterminal
  - etc

## CYK for RNA folding

### Initialization:

$$\gamma(i, i-1) = \log P(\epsilon)$$

### Iteration:

For  $i = 1$  to  $N$   
For  $j = i$  to  $N$

$$\gamma(i, j) = \max \left\{ \begin{array}{l} \gamma(i+1, j-1) + \log P(x_i S x_j) \\ \gamma(i, j-1) + \log P(S x_i) \\ \gamma(i+1, j) + \log P(x_i S) \\ \max_{i < k < j} \gamma(i, k) + \gamma(k+1, j) + \log P(S S) \end{array} \right.$$

## Evaluation

Recall HMMs:

Forward:  $f_i(i) = P(x_1 \dots x_i, \pi_i = l)$

Backward:  $b_k(i) = P(x_{i+1} \dots x_N \mid \pi_i = k)$

Then,

$$P(x) = \sum_k f_k(N) a_{k0} = \sum_i a_{0i} e_i(x_1) b_i(1)$$

Analogue in SCFGs:

Inside:  $a(i, j, V)$  =  $P(x_i \dots x_j$  is generated by  
nonterminal  $V$ )

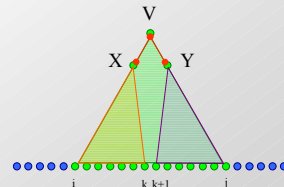
Outside:  $b(i, j, V) = P(x, \text{excluding } x_i \dots x_j \text{ is generated by } S \text{ and}$   
the excluded part is rooted at  $V$ )

## The Inside Algorithm

To compute

$$a(i, j, V) = P(x_i \dots x_j, \text{produced by } V)$$

$$a(i, j, v) = \sum_X \sum_Y a(i, k, X) a(k+1, j, Y) P(V \rightarrow XY)$$



### Algorithm: Inside

**Initialization:**

For  $i = 1$  to  $N$ ,  $V$  a nonterminal,  
 $a(i, i, V) = P(V \rightarrow x_i)$

**Iteration:**

For  $i = 1$  to  $N-1$   
 For  $j = i+1$  to  $N$   
 For  $V$  a nonterminal  
 $a(i, j, V) = \sum_x \sum_y \sum_k a(i, k, X) a(k+1, j, Y) P(V \rightarrow XY)$

**Termination:**

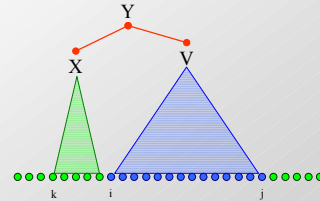
$P(x | \theta) = a(1, N, S)$

### The Outside Algorithm

$b(i, j, V) = \text{Prob}(x_1 \dots x_{i-1}, x_{j+1} \dots x_N, \text{ where the "gap" is rooted at } V)$

Given that  $V$  is the right-hand-side nonterminal of a production,

$$b(i, j, V) = \sum_x \sum_y \sum_{k < i} a(k, i-1, X) b(k, j, Y) P(Y \rightarrow XV)$$



### Algorithm: Outside

**Initialization:**

$b(1, N, S) = 1$   
 For any other  $V$ ,  $b(1, N, V) = 0$

**Iteration:**

For  $i = 1$  to  $N-1$   
 For  $j = N$  down to  $i$   
 For  $V$  a nonterminal  
 $b(i, j, V) = \frac{\sum_x \sum_y \sum_{k < i} a(k, i-1, X) b(k, j, Y) P(Y \rightarrow XV) + \sum_x \sum_y \sum_{k < i} a(j+1, k, X) b(i, k, Y) P(Y \rightarrow VX)}$

**Termination:**

It is true for any  $i$ , that:  
 $P(x | \theta) = \sum_x b(i, i, X) P(X \rightarrow x_i)$

### Learning for SCFGs

We can now estimate

$c(V)$  = expected number of times  $V$  is used in the parse of  $x_1 \dots x_N$

$$c(V) = \frac{1}{P(x | \theta)} \sum_{1 \leq i \leq N} \sum_{i < j \leq N} a(i, i, V) b(i, j, v)$$

$$c(V \rightarrow XY) = \frac{1}{P(x | \theta)} \sum_{1 \leq i \leq N} \sum_{i < j \leq N} \sum_{i < k < j} b(i, j, V) a(i, k, X) a(k+1, j, Y) P(V \rightarrow XY)$$

### Learning for SCFGs

Then, we can re-estimate the parameters with EM, by:

$$P^{\text{new}}(V \rightarrow XY) = \frac{c(V \rightarrow XY)}{c(V)}$$

$$P^{\text{new}}(V \rightarrow a) = \frac{c(V \rightarrow a)}{c(V)} = \frac{\sum_{i: x_i = a} b(i, i, V) P(V \rightarrow a)}{\sum_{1 \leq i \leq N} \sum_{i < j \leq N} a(i, j, V) b(i, j, V)}$$

### Summary: SCFG and HMM algorithms

| <u>GOAL</u>       | <u>HMM algorithm</u> | <u>SCFG algorithm</u> |
|-------------------|----------------------|-----------------------|
| Optimal parse     | Viterbi              | CYK                   |
| Estimation        | Forward<br>Backward  | Inside<br>Outside     |
| Learning          | EM: Fw/Bck           | EM: Ins/Outs          |
| Memory Complexity | $O(N K)$             | $O(N^2 K)$            |
| Time Complexity   | $O(N K^2)$           | $O(N^3 K^3)$          |

Where  $K$ : # of states in the HMM  
 # of nonterminals in the SCFG