

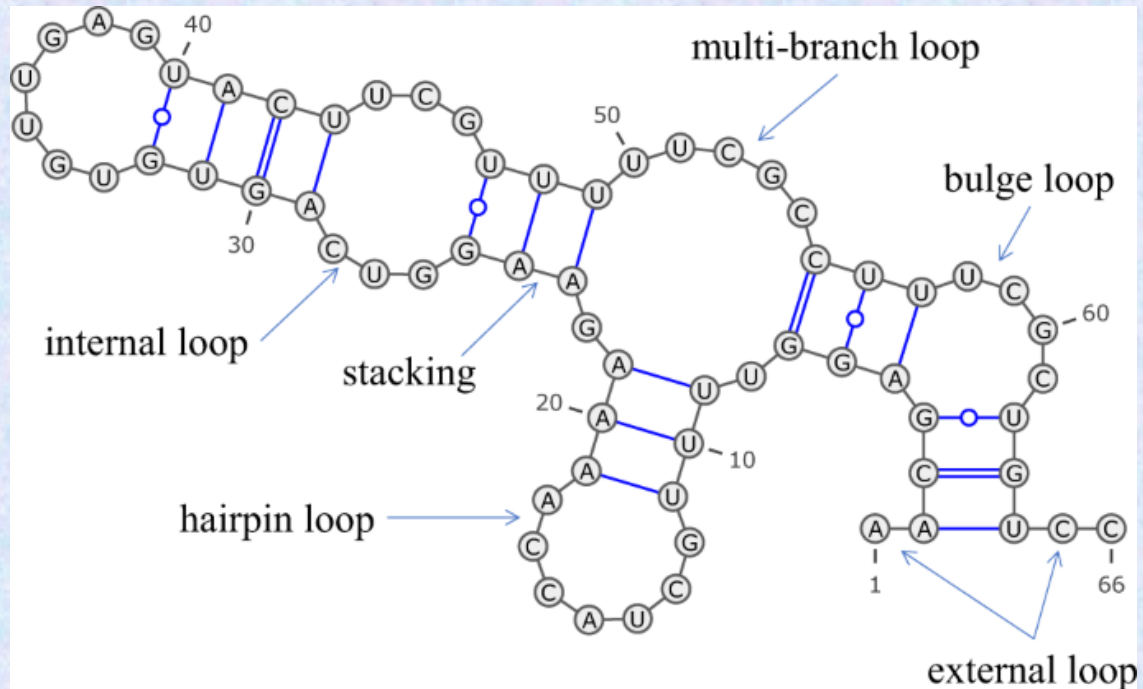
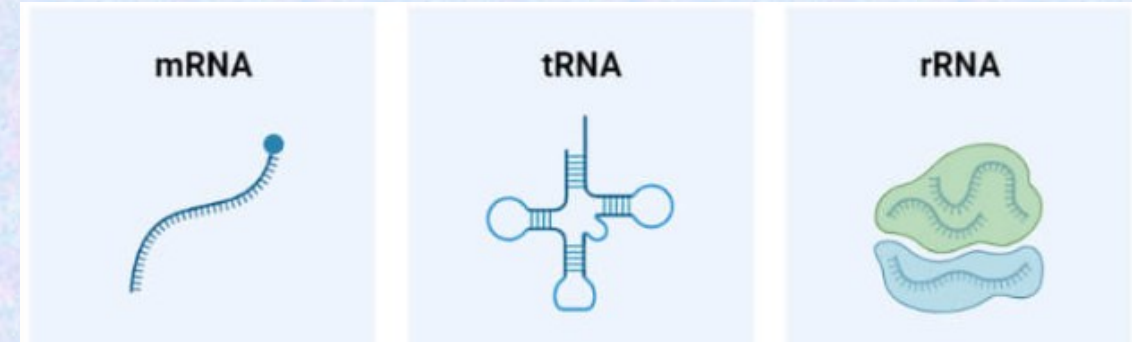
# RNA SECONDARY STRUCTURE PREDICTION USING STOCHASTIC CONTEXT FREE GRAMMARS

Sudhanshu Bharadwaj

# THE BIOLOGY OF RNA

2

- Single stranded chain of Nucleotides  $\{A, U, C, G\}$ , with a variety of functions.
- Base pairing rules:  $\{A \leftrightarrow U, C \leftrightarrow G \text{ and wobbly } (U \leftrightarrow G)\}$ , but upto 40% of base pairs are non-canonical.



- This leads to RNA adopting various secondary structures
- Knowing these secondary structures is important to predict function and sometimes sufficient for our purposes.

# SO HOW DO WE PREDICT THESE STRUCTURES?

- Given a sequence of RNA nucleotides, can we predict where secondary structural motifs will occur?

# SO HOW DO WE PREDICT THESE SECONDARY STRUCTURES?

- Given a sequence of RNA nucleotides, can we predict where secondary structural motifs will occur?

RNA secondary structure prediction from sequence alignments using a network of k-nearest neighbor classifiers	2006	Bindewald et al.	Hierarchical network of k-nearest neighbor model
Developing parallel ant colonies filtered by deep learned constrains for predicting RNA secondary structure with pseudo-knots	2020	Quan et al.	Bi-LSTM
RNA Secondary Structure Prediction Based on Long Short-Term Memory Model	2018	Wu et al.	Bi-LSTM
Predicting RNA secondary structure via adaptive deep recurrent neural networks with energy-based filter	2019	Lu et al.	Bi-LSTM
A New Method of RNA Secondary Structure Prediction Based on Convolutional Neural Network and Dynamic Programming	2019	Zhang et al.	CNN
DMfold: A Novel Method to Predict RNA Secondary Structure with Pseudoknots Based on Deep Learning and Improved Base Pair Maximization Principle	2019	Wang et al.	Bi-LSTM
Improving RNA secondary structure prediction via state inference with deep recurrent neural networks	2020	Willmott et al.	Bi-LSTM


# SO HOW DO WE PREDICT THESE STRUCTURES?

Given a sequence of RNA nucleotides, can we predict where secondary structural motifs will occur?

More interpretable classical algorithms

- Minimize the overall energy : Zuker Algorithm
- Maximize the number of base pairings : Nussinov Algorithm
- Using Stochastic Context free grammars?

*(Note: Pseudonots are a secondary structure that all the above methods cannot take into consideration)*



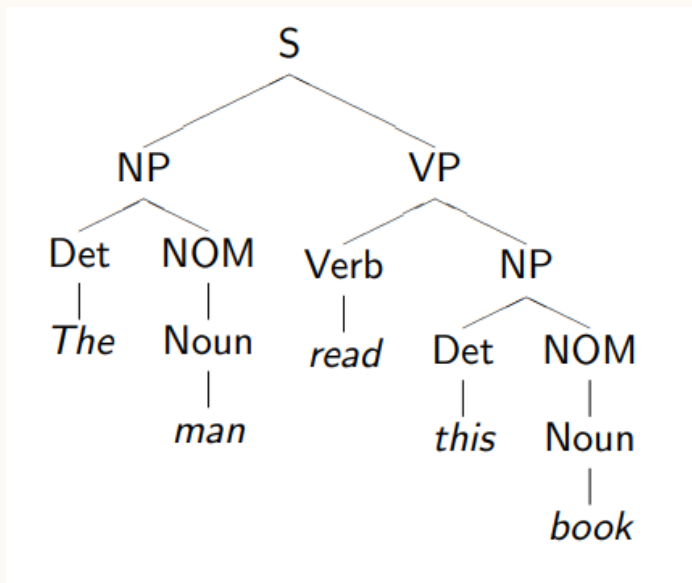
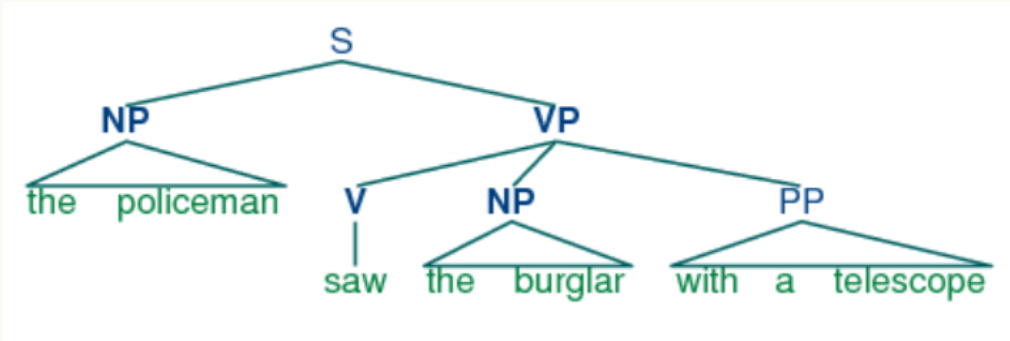
# STOCHASTIC CONTEXT FREE GRAMMARS?

A small detour

# GENERATIVE GRAMMARS

- **$\Sigma$** : A set of terminal symbols. i.e. the characters in the final string eg:  **$\{A, U, C, G\}$**
- **$N$** : A set of non-terminal symbols  
(*more abstract symbols – like a “loop”, “hairpin loop” etc.*)
- **$S$** : A start non-terminal symbol
- **$P$** : A set of production rules that maps strings  $v \rightarrow w$ . (*Where  $V$  contains at least one non-terminal\**)

# CONTEXT-FREE GRAMMARS



- **$\Sigma$** : A set of terminal symbols. i.e. the characters in the final string eg:  **$\{A, U, C, G\}$**
- **$N$** : A set of non-terminal symbols  
(*more abstract symbols – like a “loop”, “hairpin loop” etc.*)
- **$S$** : A start non-terminal symbol
- **$P$** : A set of production rules that maps non-terminals to strings  $a \rightarrow w$  ( $a \in N$ )

And production rules can be reduced to the form\*\*:

$v \rightarrow yz$       where  $v, y, z \in N$  (all non – terminals)

$v \rightarrow A$       where  $v \in N, A \in \Sigma$  (non – terminal  $\rightarrow$  terminal)



# STOCHASTIC CONTEXT-FREE GRAMMARS

9

- A collection of  $\{S, \Sigma, N, P\}$
- Production rules can be reduced to the form:

$v \rightarrow yz$       where  $v, y, z \in N$  (all non terminals)

$v \rightarrow A$       where  $v \in N, A \in \Sigma$  (non – terminal  $\rightarrow$  terminal)

- Each productions rule has probability associated it

Transition:  $\mathbb{P}(v \rightarrow yz) = t_v(y, z)$

Emission:  $\mathbb{P}(v \rightarrow A) = e_v(A)$

$$\text{where } \left( \sum_{y,z \in N} t_v(y, z) + \sum_{A \in \Sigma} e_v(A) \right) = 1 \quad \forall v \in N$$

a. Productions

$P = \{$

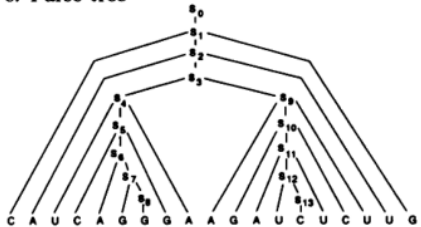
$S_0 \rightarrow S_1,$	$S_7 \rightarrow G S_8,$
$S_1 \rightarrow C S_2 G,$	$S_8 \rightarrow G,$
$S_2 \rightarrow A S_3 U,$	$S_9 \rightarrow A S_{10} U,$
$S_3 \rightarrow S_4 S_9,$	$S_{10} \rightarrow G S_{11} C,$
$S_4 \rightarrow U S_5 A,$	$S_{11} \rightarrow A S_{12} U,$
$S_5 \rightarrow C S_6 G,$	$S_{12} \rightarrow U S_{13},$
$S_6 \rightarrow A S_7,$	$S_{13} \rightarrow C$

$\}$

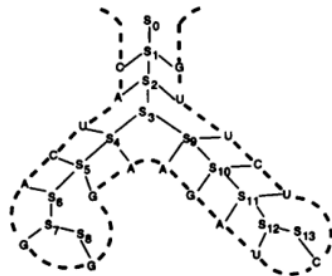
b. Derivation

$S_0 \Rightarrow S_1 \Rightarrow C S_2 G \Rightarrow C A S_3 U G \Rightarrow C A S_4 S_9 U G$   
 $\Rightarrow C A U C S_5 A S_9 U G \Rightarrow C A U C S_6 G A S_9 U G$   
 $\Rightarrow C A U C A S_7 G A S_9 U G \Rightarrow C A U C A G S_8 G A S_9 U G$   
 $\Rightarrow C A U C A G G G A S_9 U G \Rightarrow C A U C A G G G A A S_{10} U U G$   
 $\Rightarrow C A U C A G G G A A G S_{11} C U U G$   
 $\Rightarrow C A U C A G G G A A G A S_{12} U C U U G$   
 $\Rightarrow C A U C A G G G A A G A U S_{13} U C U U G$   
 $\Rightarrow C A U C A G G G A A G A U C U C U U G.$

c. Parse tree

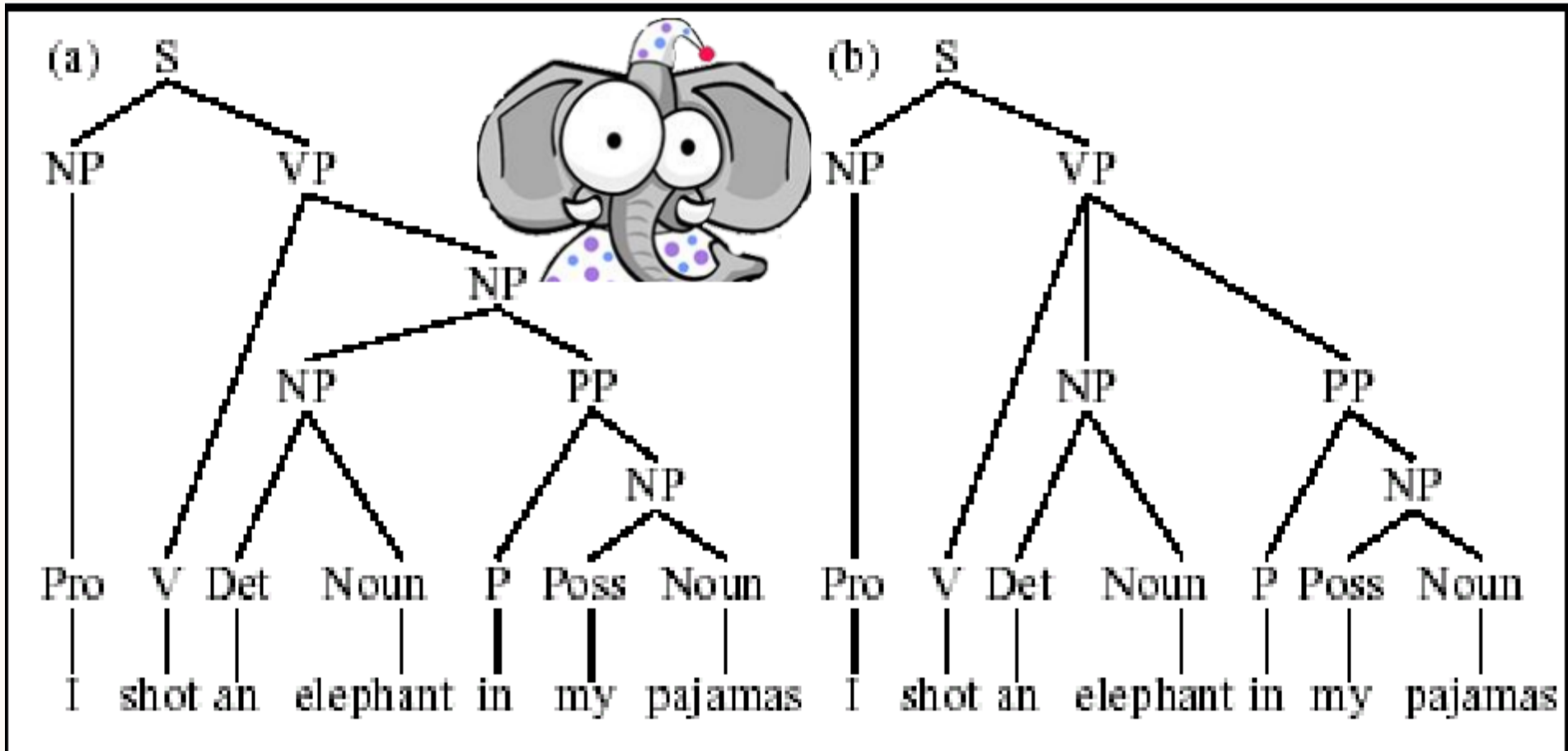


d. Secondary Structure



# AMBIGUITY IN PARSING

DIFFERENT SETS OF RULES CAN GIVE RISE TO THE SAME SEQUENCE.



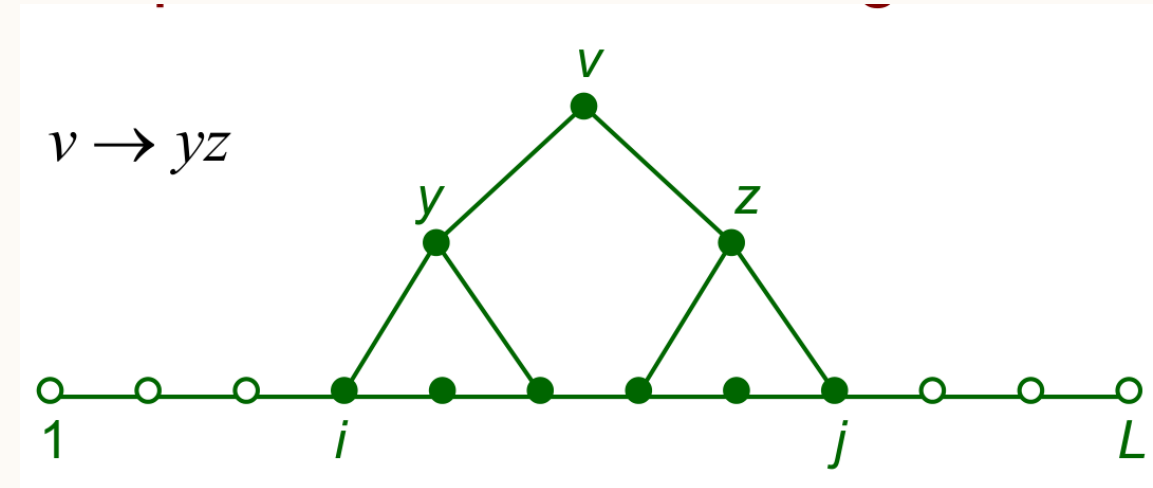
# How likely is a given sequence?

## The Inside algorithm.

A dynamic programming problem.

- Let  $T$  be the given text, generated from a SCFG whose rules are known.
- Let there be  $M$  non – terminals
- Let  $i, j$  represent indices in  $T$
- Let  $v$  represent a nonterminal.

Involves filling  $\alpha(i, j, v)$ , a 3D matrix representing the probability of generating  $T[i, j]$  from a non – terminal  $v$ .



# How likely is a given sequence?

## The Inside algorithm.

### The algorithm:

- Initialization ( $i = 1:L, v = 1:M$ )  
 $\alpha(i, i, v) = e_v(T_i)$
- Recursion ( $i = L - 1:1, j = i + 1:L, v = 1:M$ )

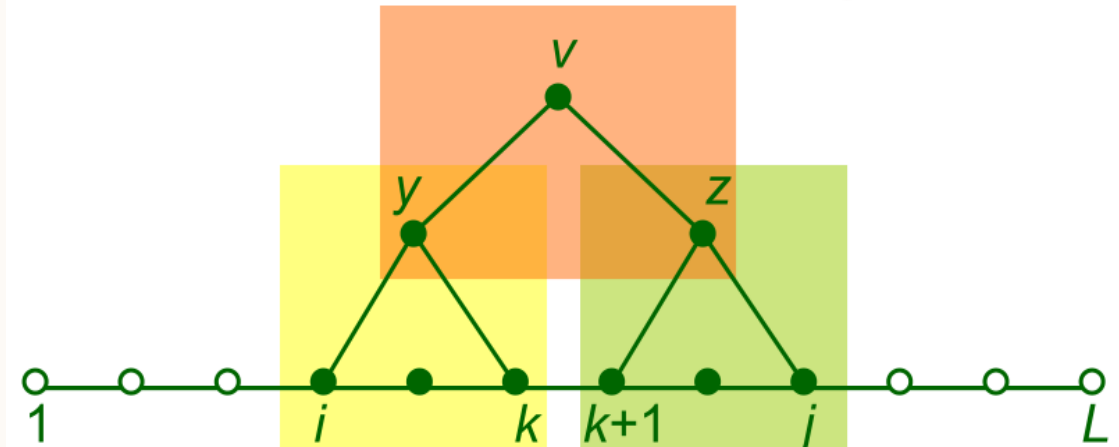
$$\alpha(i, j, v) = \sum_{y=1}^M \sum_{z=1}^M \sum_{k=i}^{j-1} t_v(y, z) \times \alpha(i, k, y) \times \alpha(k + 1, j, z)$$

- Termination

$$\mathbb{P}(T|\theta) = \alpha(1, L, 1)$$

- Time Complexity

$$O(L^3 M^3)$$



Here,  $\alpha(i, j, v)$  is a 3D matrix representing the probability of generating  $T[i, j]$  from a non-terminal  $v$ .

# What is the most likely parse for a sequence?

## The CYK algorithm

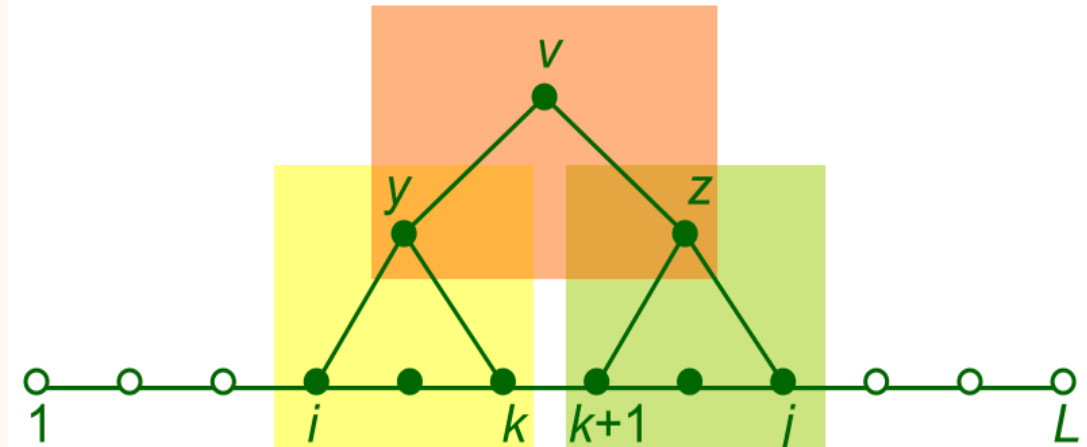
### The algorithm:

- Initialization ( $i = 1:L, v = 1:M$ )  
 $\gamma(i, i, v) = \log e_v(T_i); \tau(i, i, v) = (0, 0, 0)$
- Recursion ( $i = L - 1:1, j = i + 1:L, v = 1:M$ )

$$\gamma(i, j, v) = \max_{k \in (i, j), y, z} \{ \log t_v(y, z) + \gamma(i, k, y) + \gamma(k + 1, j, z) \}$$

$$\tau(i, j, v) = \operatorname{argmax}_{k \in (i, j), y, z} \{ \log t_v(y, z) + \gamma(i, k, y) + \gamma(k + 1, j, z) \}$$

- Termination  
 $\log \mathbb{P}(T, \pi^* | \theta) = \gamma(1, L, 1)$
- Time Complexity  
 $O(L^3 M^3)$



Here,  $\gamma(i, j, v)$  is a 3D matrix representing the probability of the most likely parse of generating  $T[i, j]$  from a non-terminal  $v$ .

And  $\tau(i, j, v)$  stores the tuples  $(k, y, z)$  about the previous transition in the path that gives rise to the most likely parse of generating  $T[i, j]$  from a non-terminal  $v$ .

(Will be important later: For reference)

# The Outside algorithm.

## The algorithm:

- Initialization ( $i = 1:L, v = 1:M$ )  
 $\beta(i, L, v) = (v == \text{Start})$
- Recursion ( $i = 1:L, j = L:i, v = 1:M$ )

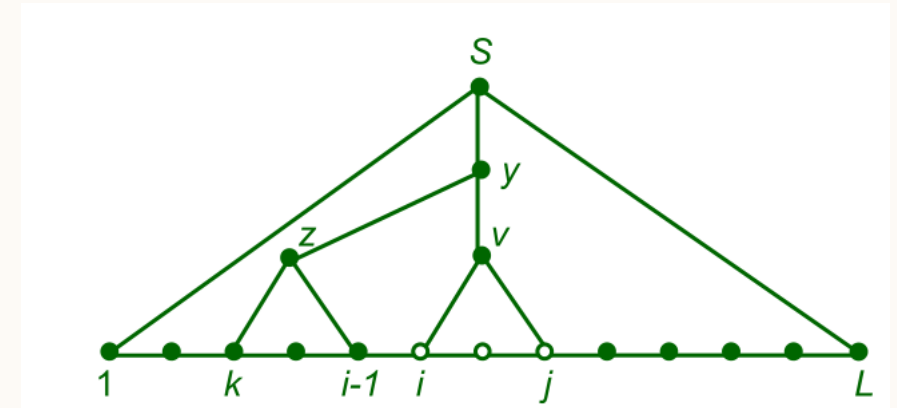
$$\beta(i, j, v) = \sum_{y=1}^M \sum_{z=1}^M \sum_{k=1}^{i-1} t_y(z, v) \times \alpha(k, i-1, z) \times \beta(k, j, y) \quad \text{when } y \rightarrow zv$$

$$+ \sum_{y=1}^M \sum_{z=1}^M \sum_{k=j+1}^L t_y(v, z) \times \alpha(j+1, k, z) \times \beta(i, k, y) \quad \text{when } y \rightarrow v, z$$

- Termination
- Time Complexity

$$\mathbb{P}(T|\theta) = \alpha(1, L, 1)$$

$$O(L^3 M^3)$$



Here,  $\beta(i, j, v)$  is a 3D matrix representing the probability of generating  $T(1:i), T(j:L)$  and  $v$  from  $S$ .  
*i.e*  $\mathbb{P}(S \rightarrow T | V \rightarrow T(i, j))$

$\alpha$  is the probability  $\mathbb{P}(V \rightarrow T(i, j))$  is from the inside algorithm.

BUT PERHAPS THE MOST IMPORTANT QUESTION...

# HOW DOES ONE LEARN THESE GRAMMAR RULES AND THEIR PROBABILITIES?

- A lot of latent variables: Non-terminals are not observable, therefore one can't use simple parameter estimation techniques.
- Expectation Maximization: An ideal method to estimate parameters when there are a lot of latent parameters.

# THE INSIDE-OUTSIDE ALGORITHM

16

$$\hat{e}_v(A) = \frac{c(v \rightarrow A)}{c(v)} = \frac{\sum_{i|x_i=A} \beta(i, i, v) e_v(A)}{\sum_{i=1}^L \sum_{j=i}^L \beta(i, j, v) \alpha(i, j, v)}$$

← cases where  $v$  used to generate  $A$

$$\hat{t}_v(y, z) = \frac{c(v \rightarrow yz)}{c(v)}$$

$$= \frac{\sum_{i=1}^{L-1} \sum_{j=i+1}^L \sum_{k=i}^{j-1} t_v(y, z) \beta(i, j, v) \alpha(i, k, y) \alpha(k+1, j, z)}{\sum_{i=1}^L \sum_{j=i}^L \beta(i, j, v) \alpha(i, j, v)}$$

← cases where  $v$  used to generate any subsequence

Here,  $\alpha$  and  $\beta$  are the DP table used in the inside and outside algorithms.

- At each step, calculate  $\alpha$  and  $\beta$
- Re-estimate the probabilities in the above way to increase the likelihood of the given data. ([IBM Research | Technical Paper Search | A Derivation of the Inside-Outside Algorithm from the EM Algorithm \(Search Reports\)](#) )



BUT PERHAPS THE MOST IMPORTANT QUESTION...

# HOW DOES ONE LEARN THESE GRAMMAR RULES AND THEIR PROBABILITIES?

- A lot of latent variables: Non-terminals are not observable, therefore one can't use simple parameter estimation techniques.
- Expectation Maximization: An ideal method to estimate parameters when there are a lot of latent parameters.
  - Extremely slow convergence, despite guarantee that likelihood will increase in each EM step.
  - Each iteration requires estimation of  $\alpha$  and  $\beta$ , updating  $t_v(y, z)$ , each of which is a  $O(L^3 M^3)$  operation. For  $L \approx 100, M \approx 10$ , the problem quickly becomes infeasible for even 1 iteration and 1 string.

# BETTER ESTIMATION OF SCFG PARAMETERS?

Idea 1: Careful Selection of Rules

$S \rightarrow LS \text{ or } L$

$L \rightarrow s, \text{ or } dFd'$

$F \rightarrow dFd' \text{ or } LS$

where  $s \in \{A, U, C, G\}$ , and  $(d, d')$  is a base pair  
 $S, L, F$  are all Non-terminals

Here  $S$  produces Loops, and  $F$  produces stems.

Perhaps we can reduce the number of free parameters based on real data?

a)  $S \rightarrow LS \rightarrow LLLLLLLS \rightarrow LLLLLLLL$   
 $\rightarrow ssLsssss \rightarrow ssdFdsssss$   
 $\rightarrow ssdddFdddsssss$   
 $\rightarrow ssdddLSdddsssss$   
 $\rightarrow ssdddLLLdddsssss$   
 $\rightarrow ssdddssssdddsssss$

b)

$s \begin{matrix} ss \\ d-d \\ d-d \\ ssd-dsssss \end{matrix} s$

# BETTER ESTIMATION OF SCFG PARAMETERS?

## Idea 1: Careful Selection of Rules

$S \rightarrow LS \text{ or } L, \quad L \rightarrow s, \text{ or } dFd', \quad F \rightarrow dFd' \text{ or } LS$   
 where  $s \in \{A, U, C, G\}$ , and  $(d, d')$  is a base pair,  
 $S, L, F$  are all Non-terminals.

In the above rules:

- $L \rightarrow s$ : Produces all non-paired base pairs
- $F \rightarrow dFd'$  „  $F \rightarrow dFd'$ : Produce all paired up base-pairs

If we estimate the frequency of different  $(s)$  in unpaired regions(loops) and  $(d, d')$  in paired up regions (stems), and use this information to infer rules. We can estimate this using RNA whose secondary structure has already been found. (Labelled data)

a)  $S \rightarrow LS \rightarrow LLLLLLLS \rightarrow LLLLLLLL$   
 $\rightarrow ssLsssss \rightarrow ssdFdsssss$   
 $\rightarrow ssdddFdddsssss$   
 $\rightarrow ssdddLSdddsssss$   
 $\rightarrow ssdddLLLLdddsssss$   
 $\rightarrow ssdddssssdddsssss$

b)

$s \begin{matrix} ss \\ s \end{matrix}$   
 $d-d$   
 $d-d$   
 $ssd-dsssss$

# BETTER ESTIMATION OF SCFG PARAMETERS?

## Idea 1: Careful Selection of Rules

$S \rightarrow LS \text{ or } L, \quad L \rightarrow s, \text{ or } dFd', \quad F \rightarrow dFd' \text{ or } LS$   
 where  $s \in \{A, U, C, G\}$ , and  $(d, d')$  is a base pair,  
 $S, L, F$  are all Non – terminals.

Create two tables

- $T_1(4 \times 1)$ : which counts frequencies of different bases in loop regions
- $T_2(4 \times 4)$ : which counts frequencies of different base pairs in stem regions.

Now  $p(L \rightarrow dFd') = \sum_{(d,d') \in \Sigma^2} (p(L \rightarrow dFd')) \times T_2(d, d')$

i.e. we only estimate the frequency of the overall rule  $L \rightarrow dFd'$ , each of 16 individual rules's probability is found by multiplying the above by the frequency in labelled data. (Similarly for other rules)

### Single Frequencies in Loop Region

A : 0.34

C : 0.21

G : 0.18

U : 0.28

### Double Frequencies in Stem Region

Base:	A	C	G	U
A	0.00	0.01	0.00	0.20
C	0.01	0.00	0.24	0.00
G	0.00	0.24	0.00	0.04
U	0.20	0.00	0.04	0.00

$S \rightarrow LS$  (86.9%) |  $L$  (13.1%)

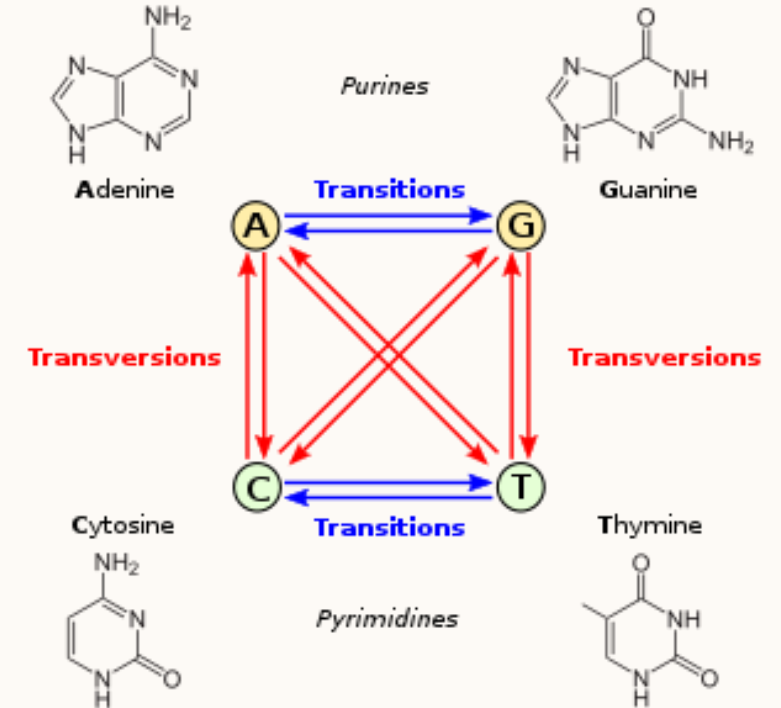
$F \rightarrow dFd$  (78.8%) |  $LS$  (21.2%)

$L \rightarrow s$  (89.5%) |  $dFd$  (10.5%)

# IDEA 2: FOR BETTER SECONDARY STRUCTURE ESTIMATION: USING SIMILAR SEQUENCES

21

- If we take “related sequences” (*this is done by constructing a phylogeny of evolutionarily similar RNA*) and align them, even they will have some mutations, the secondary structure motifs are likely to be in similar places.
- We can then find the probability of co-occurrences of different bases and base-pair combinations in loop/stem regions. Using this data under the assumptions of a model known as the [General Reversible Model](#), we can estimate the mutation rates of one base/ base pair to another.
- These rates depend on biological likelihood. Especially in paired up bases( stems), Transitions (which preserve base pairing) are more common than transversions.



Transitions (which preserve base pairing to some extent) are more common than transversions.

# IDEA 2: FOR BETTER SECONDARY STRUCTURE ESTIMATION: USING SIMILAR SEQUENCES

22

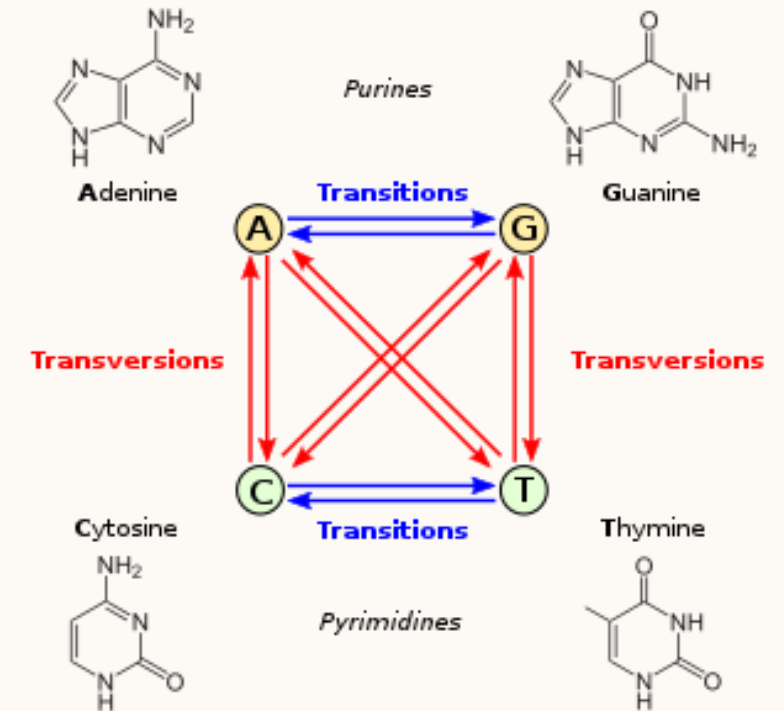
**Table 2.** The entries,  $r_{XY}$ , for the loop rate matrix. Transitions are more frequent than transversions

$X \backslash Y$	A	C	G	U
A	-0.75	0.16	0.32	0.26
C	0.40	-1.57	0.24	0.93
G	0.55	0.17	-0.96	0.24
U	0.35	0.51	0.19	-1.05

**Table 4.** Marginal rate matrix for stems. This matrix is similar to the above matrix for loops, except that this one was estimated from stem regions. Notice the high transition/transversion ratio relative to loops

$X \backslash Y$	A	C	G	U
A	-1.15	0.13	0.79	0.23
C	0.09	-0.84	0.16	0.59
G	0.45	0.13	-0.70	0.11
U	0.18	0.70	0.16	-1.03

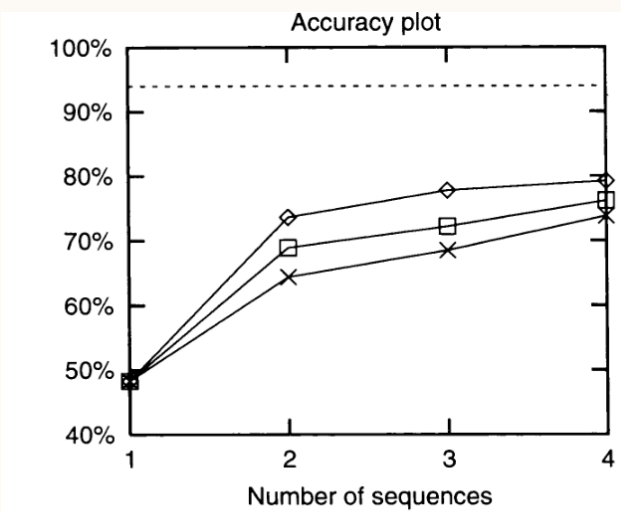
- The mutation rates are different for loops and stems because of base pairing. (These parameters can again be learnt from biological data)
- Therefore, we take multiple related RNA sequence, align them and then estimate mutation rates; use this to estimate structure before
- This can be incorporated with the CYK algorithm in a Bayesian fashion (details will be skipped here) to give better results.



Transitions (which preserve base pairing to some extent) are more common than transversions in paired-up bases.

# EXPERIMENTS WITH REAL DATA

- As said earlier, current approaches use Deep Learning based methods. Therefore, it was hard to find a good implementation of the above method online.
- Will finish implementation and show its results in the report



**Fig. 4.** A comparison of results with and without phylogeny. Diamonds (◇ ) denote the curve for predictions with phylogeny, while boxes (□) denote the one without. Crosses (×) denote results using CLUSTAL W alignments and phylogeny estimation. The dotted line at 94% represents the maximum possible prediction accuracy with regard to the pseudoknots.

Grammar	Full Benchmarking Set Sensitivity % (PPV %)	R		
G1	17 (12)	1 (1)	2 (2)	1 (1)
G3	34 (31)	37 (35)	28 (28)	31 (22)
G4	10 (8)	10(8)	19 (17)	4(2)
G5	3(4)	3(4)	2(3)	4(3)
G6	47 (45)	49 (49)	47 (49)	44 (33)
G2	36 (25)	31 (23)	59 (48)	33 (17)
G7	45 (43)	46 (46)	50 (52)	40 (30)
G8	46 (44)	46 (46)	52 (53)	44 (32)
G6 <sup>s</sup>	49 (45)	50 (50)	50 (51)	44 (32)
mfold v3.1.2	56 (48)	56 (51)	70 (66)	46 (30)
Vienna RNA	55 (47)	55 (51)	67 (64)	45 (30)
PKNOTS	50 (41)	53 (46)	58 (55)	38 (24)
RNAstructure	56 (47)	58 (52)	62 (58)	46 (30)
Pfold	39 (69)	42 (76)	35 (64)	33 (54)

# THANK YOU

Sudhanshu B

[sudhanshub@iisc.ac.in](mailto:sudhanshub@iisc.ac.in)

## *Main References:*

- Knudsen, B., & Hein, J. (1999). RNA secondary structure prediction using stochastic context-free grammars and evolutionary history. In *Bioinformatics* (Vol. 15, Issue 6, pp. 446–454). Oxford University Press (OUP).  
<https://doi.org/10.1093/bioinformatics/15.6.446>
- *Bjarne Knudsen and Jotun Hein. Pfold: RNA secondary structure pre-diction using stochastic context-free grammars. Nucleic Acids Research, 31(13):3423–3428, 07 2003*
- *Several figures have been borrowed from Andrew McCallum and Anthony Gitter slides*