

(Positive) Design of RiboNucleic Acids

Yann Ponty^{*,•,†}

^{*} Centre National de la Recherche Scientifique (CNRS)

[•] LIX, Ecole Polytechnique

[†] AMIBio team, Inria Saclay

<http://goo.gl/mejsFh>



Yann Ponty



PhD in Computer Science, Université Paris-Sud (France)



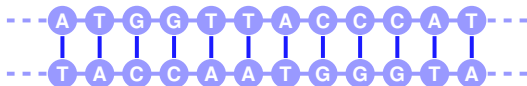
- CNRS research scientist
- Faculty at LIX, Computer Science department of Ecole Polytechnique



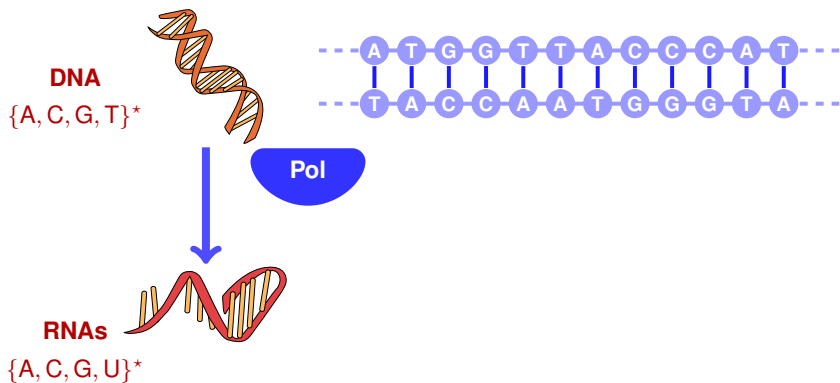
- Group leader – AMIBio team (Ecole Polytechnique and Inria Saclay)
- Postdoc experience in RNA Computational Biology (Boston, Paris) and Discrete Mathematics (Paris)
- Extended sabbatical at Simon Fraser University (Vancouver, Canada)

Fundamental *dogma* of molecular biology

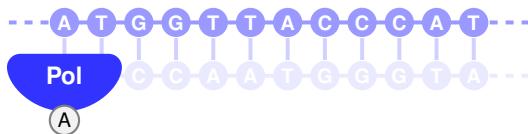
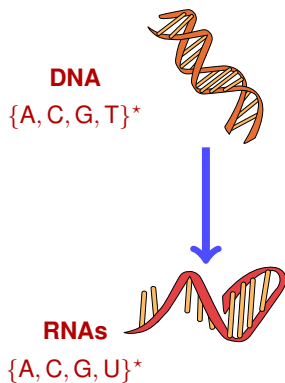
DNA
{A, C, G, T}^{*}



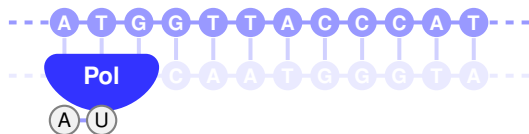
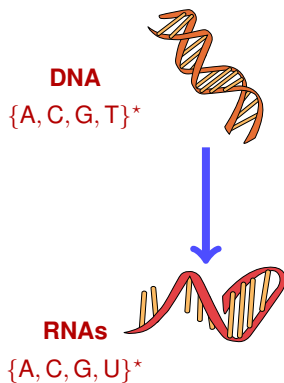
Fundamental *dogma* of molecular biology



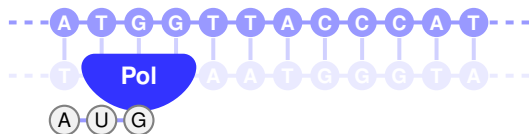
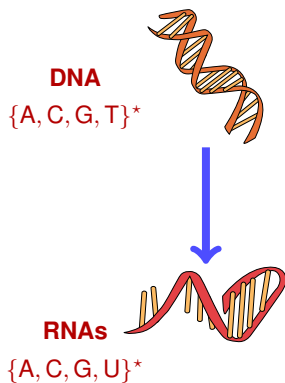
Fundamental *dogma* of molecular biology



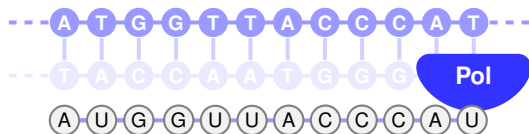
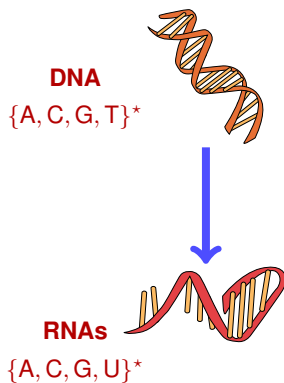
Fundamental *dogma* of molecular biology



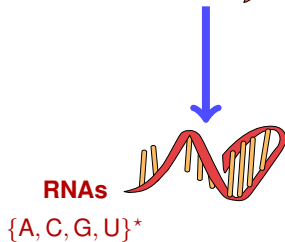
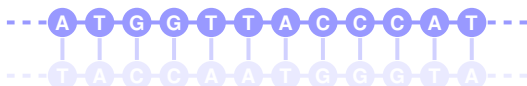
Fundamental *dogma* of molecular biology



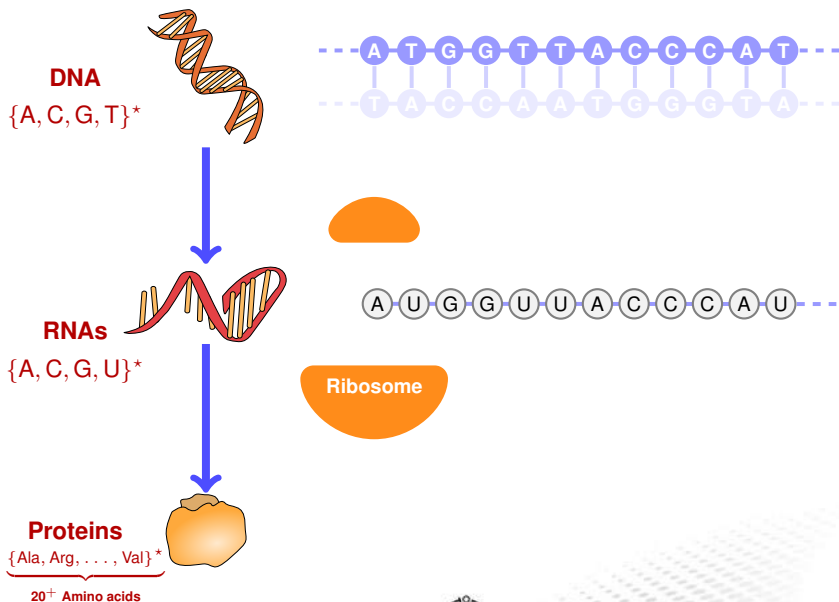
Fundamental *dogma* of molecular biology



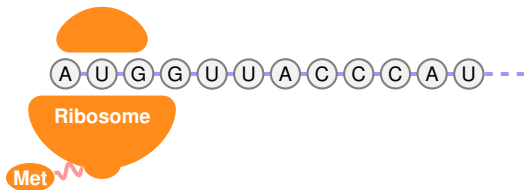
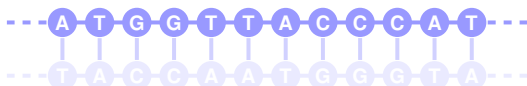
Fundamental *dogma* of molecular biology



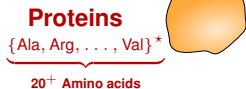
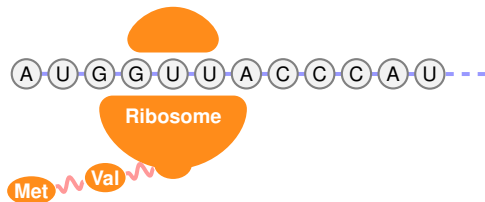
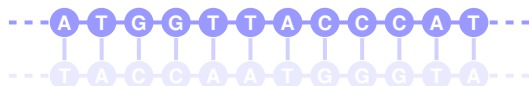
Fundamental *dogma* of molecular biology



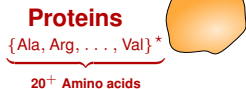
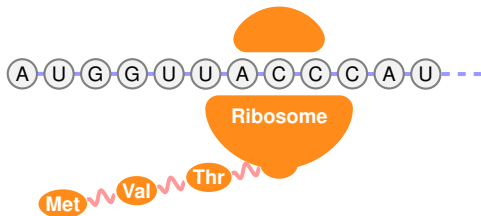
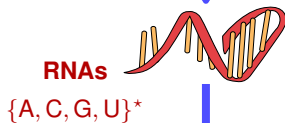
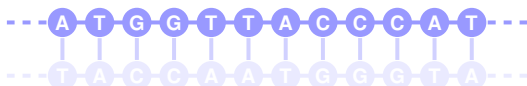
Fundamental *dogma* of molecular biology



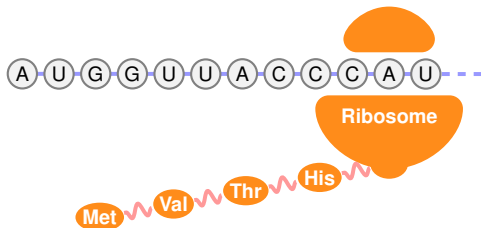
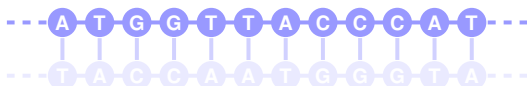
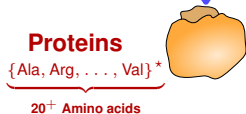
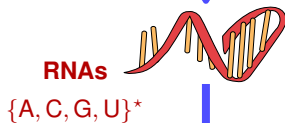
Fundamental *dogma* of molecular biology



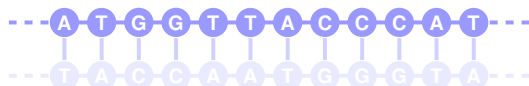
Fundamental *dogma* of molecular biology



Fundamental *dogma* of molecular biology



Fundamental *dogma* of molecular biology



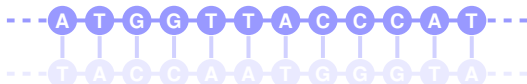
Fundamental *dogma* of molecular biology

THE CODE

(genes)

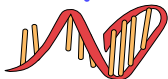
DNA

{A, C, G, T}^{*}



RNAs

{A, C, G, U}^{*}



Proteins

{Ala, Arg, . . . , Val}^{*}

20⁺ Amino acids



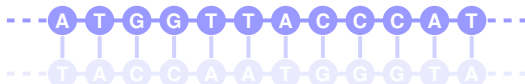
Fundamental *dogma* of molecular biology

THE CODE

(genes)

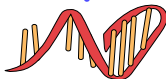
DNA

{A, C, G, T}^{*}



RNAs

{A, C, G, U}^{*}



THE MACHINE

(enzymes)

Proteins

{Ala, Arg, . . . , Val}^{*}

20⁺ Amino acids



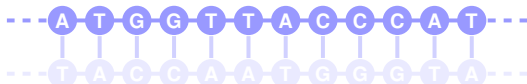
Fundamental *dogma* of molecular biology

THE CODE

(genes)

DNA

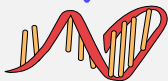
{A, C, G, T}^{*}



MEH...

RNAs

{A, C, G, U}^{*}



THE MACHINE

(enzymes)

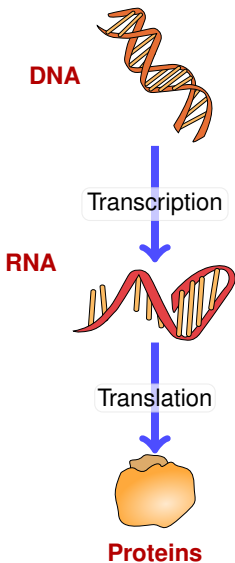
Proteins

{Ala, Arg, . . . , Val}^{*}

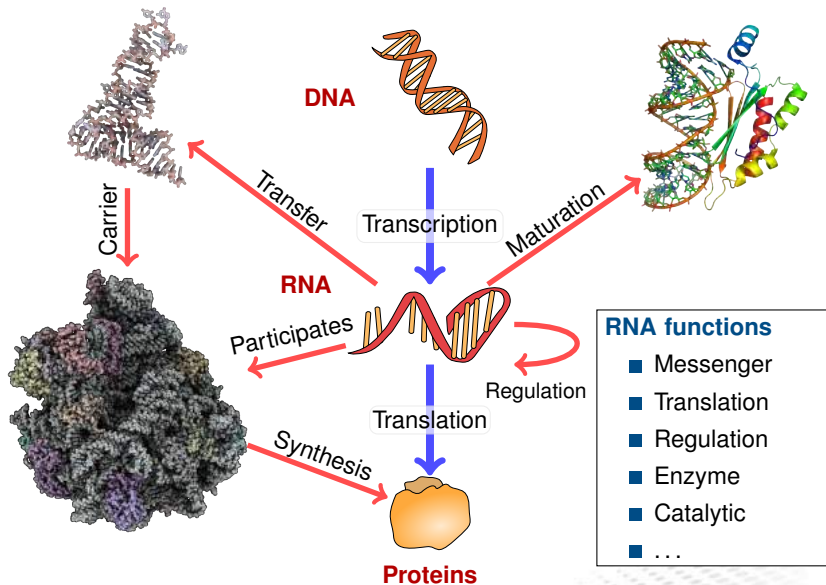
20⁺ Amino acids



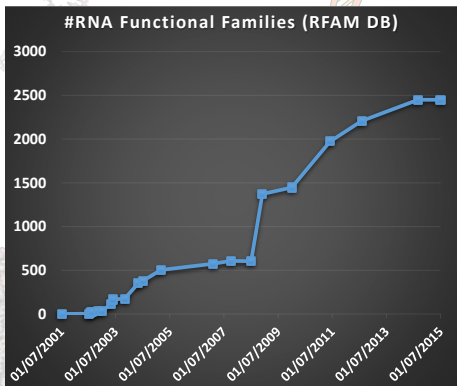
Fundamental *dogma* of molecular biology



Fundamental *dogma* of molecular biology (v2.0)



Fundamental *dogma* of molecular biology



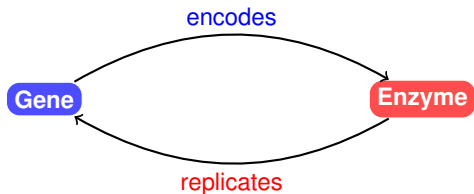
RNA functions

- Messenger
- Translation
- Regulation
- Enzyme
- Catalytic
- ...

Proteins



RNA world: Resolving the *chicken vs egg* paradox at the origin of life...



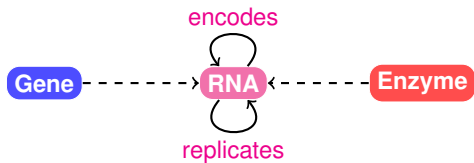
A **gene** big enough to specify **an enzyme** would be too big to replicate accurately without the aid of **an enzyme** of the very kind that it is trying to specify. So the system *apparently cannot get started*.

[...] This is the **RNA World**. To see how plausible it is, we need to look at why proteins are good at being enzymes but bad at being replicators; at why DNA is good at replicating but bad at being an enzyme; and finally why *RNA might just be good enough at both roles to break out of the Catch-22*.

R. Dawkins. *The Ancestor's Tale: A Pilgrimage to the Dawn of Evolution*



RNA world: Resolving the *chicken vs egg* paradox at the origin of life...



A **gene** big enough to specify **an enzyme** would be too big to replicate accurately without the aid of **an enzyme** of the very kind that it is trying to specify. So the system *apparently cannot get started*.

[...] This is the **RNA World**. To see how plausible it is, we need to look at why proteins are good at being enzymes but bad at being replicators; at why DNA is good at replicating but bad at being an enzyme; and finally why *RNA might just be good enough at both roles to break out of the Catch-22*.

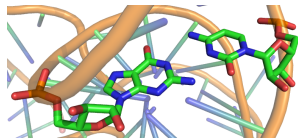
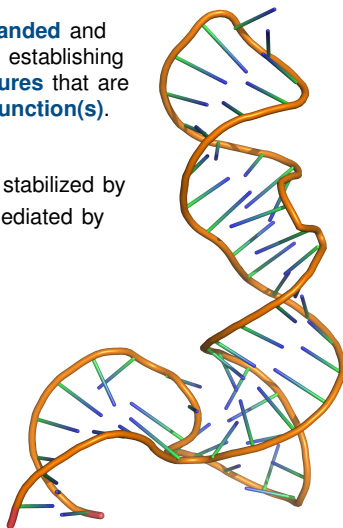
R. Dawkins. *The Ancestor's Tale: A Pilgrimage to the Dawn of Evolution*



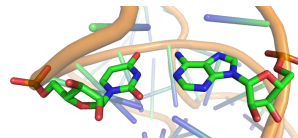
RNA folding

RNA are **single-stranded** and **fold** on themselves, establishing **complex 3D structures** that are **essential to their function(s)**.

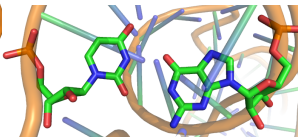
RNA structures are stabilized by **base-pairs**, each mediated by **hydrogen bonds**.



G/C



U/A



U/G

Watson/Crick base-pairs

Wobble base-pair

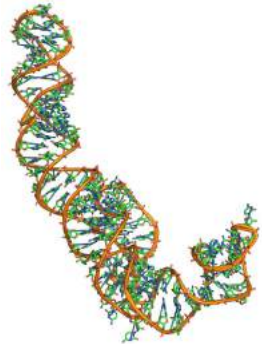
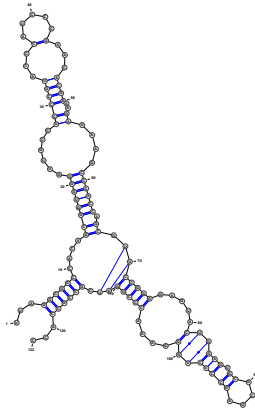
Canonical base-pairs



RNA Design

RNA = Linear Polymer = Sequence in {A, C, G, U}^{*}

```
UUAGGCGGCCACAGC
GGUGGGGUUGCCUCC
CGUACCCAUCCCGAA
CACGGAAGAUAAAGCC
CACCAGCGUUCGGG
GAGUACUGGAGUGCG
CGAGCCUCUGGGAAA
CCCGGUUCGCCGCCA
CC
```



Primary Structure

Secondary Structure

Structure Tertiaire

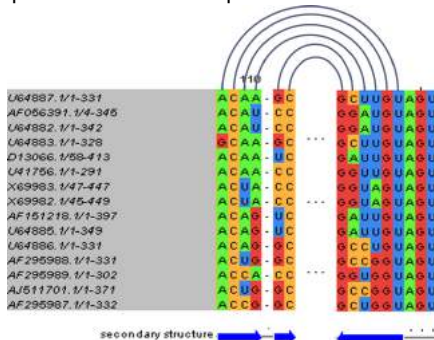
5s rRNA (PDBID: 1K73:B)



Evolution of RNAs

Homologous genes = **Functionally** equivalent, within or across organisms
Usually well-captured by **sequence similarity** in proteins, binding sites. . .

Problem: Many classes of non-(protein) coding RNAs (ncRNAs) poorly conserved at the sequence level **but** adopt a **conserved structure!**

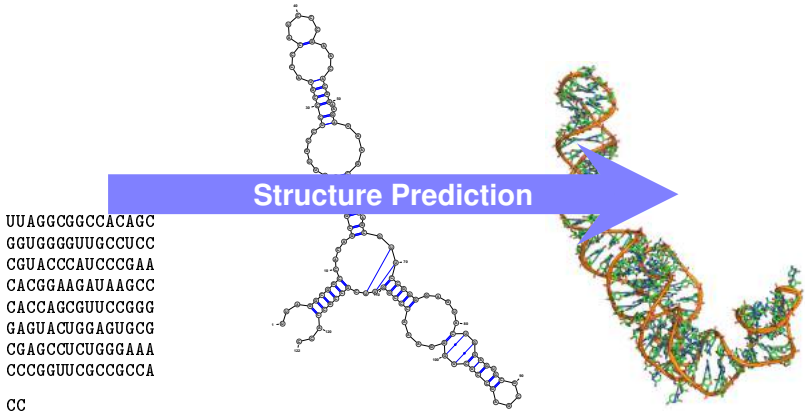


RFAM Bacterial RNase-P class B Alignment



RNA Design

RNA = Linear Polymer = Sequence in {A, C, G, U}^{*}



Primary Structure

Secondary Structure

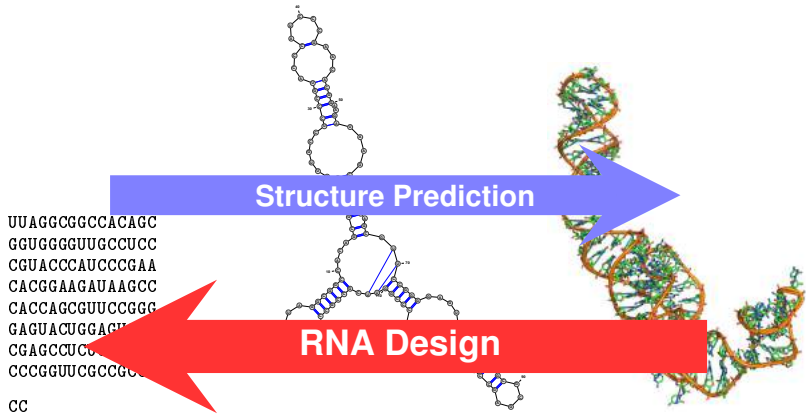
Structure Tertiaire

5s rRNA (PDBID: 1K73:B)



RNA Design

RNA = Linear Polymer = Sequence in $\{A, C, G, U\}^*$



Primary Structure

Secondary Structure

Structure Tertiaire

5s rRNA (PDBID: 1K73:B)



Why we design RNAs

- To create building blocks for synthetic systems
Rationally-designed RNAs increase orthogonality
- To assess the significance of observed phenomenon
Random models should include every established characters...
...including adoption of a single structure
- To test/push our understanding of how RNA folds
Misfolding RNAs reveal gaps in our energy models and descriptors for the conformational spaces
- To help search for homologous sequences
Incomplete covariance models hindered by limited training sets
Design can be used to generalize existing alignments
- To fuel RNA-based therapeutics
Sequence-based (siRNA, synthetic genes), but structure matters
- To perform controlled experiments



Why we design RNAs

- To create building blocks for synthetic systems
Rationally-designed RNAs increase orthogonality
- To assess the significance of observed phenomenon
Random models should include every established characters. . .
. . . including adoption of a single structure
- To test/push our understanding of how RNA folds
Misfolding RNAs reveal gaps in our energy models and descriptors for the conformational spaces
- To help search for homologous sequences
Incomplete covariance models hindered by limited training sets
Design can be used to generalize existing alignments
- To fuel RNA-based therapeutics
Sequence-based (siRNA, synthetic genes), but structure matters
- To perform controlled experiments



Why we design RNAs

- To create building blocks for synthetic systems
Rationally-designed RNAs increase orthogonality
- To assess the significance of observed phenomenon
Random models should include every established characters. . .
. . . including adoption of a single structure
- To test/push our understanding of how RNA folds
Misfolding RNAs reveal gaps in our energy models and descriptors for the conformational spaces
- To help search for homologous sequences
Incomplete covariance models hindered by limited training sets
Design can be used to generalize existing alignments
- To fuel RNA-based therapeutics
Sequence-based (siRNA, synthetic genes), but structure matters
- To perform controlled experiments



Why we design RNAs

- To create building blocks for synthetic systems
Rationally-designed RNAs increase orthogonality
- To assess the significance of observed phenomenon
Random models should include every established characters. . .
. . . including adoption of a single structure
- To test/push our understanding of how RNA folds
Misfolding RNAs reveal gaps in our energy models and descriptors for the conformational spaces
- To help search for homologous sequences
Incomplete covariance models hindered by limited training sets
Design can be used to generalize existing alignments
- To fuel RNA-based therapeutics
Sequence-based (siRNA, synthetic genes), but structure matters
- To perform controlled experiments



Why we design RNAs

- To create building blocks for synthetic systems
Rationally-designed RNAs increase orthogonality
- To assess the significance of observed phenomenon
Random models should include every established characters . . .
. . . including adoption of a single structure
- To test/push our understanding of how RNA folds
Misfolding RNAs reveal gaps in our energy models and descriptors for the conformational spaces
- To help search for homologous sequences
Incomplete covariance models hindered by limited training sets
Design can be used to generalize existing alignments
- To fuel RNA-based therapeutics
Sequence-based (siRNA, synthetic genes), but structure matters
- To perform controlled experiments



Why we design RNAs

- To create building blocks for synthetic systems
Rationally-designed RNAs increase orthogonality
- To assess the significance of observed phenomenon
Random models should include every established characters . . .
. . . including adoption of a single structure
- To test/push our understanding of how RNA folds
Misfolding RNAs reveal gaps in our energy models and descriptors for the conformational spaces
- To help search for homologous sequences
Incomplete covariance models hindered by limited training sets
Design can be used to generalize existing alignments
- To fuel RNA-based therapeutics
Sequence-based (siRNA, synthetic genes), but structure matters
- To perform controlled experiments



Controlled experiments through RNA design

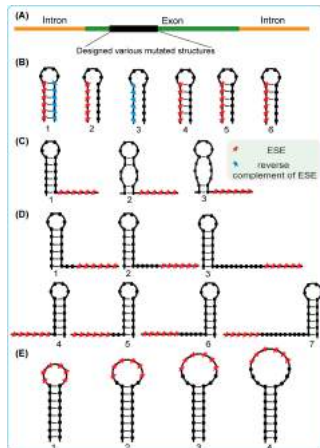
Motivation: Quantifying the impact of structure S on efficacy of a single Exon Splicing Enhancers (ESE):

- Presence of ESE motif E ;
- Different structures S_1, S_2, \dots ;
- Avoid library of ($\sim 1500!$) documented ESEs motifs.

Objectives. Design RNA which:

- 1 Folds into a prescribed structure;
- 2 Features/avoids motifs.
- 3 Control GC%, Boltz. prob.

Structural context of ESE motif in transcript was shown to affect its functionality. [Liu *et al*, FEBS Lett. 2010]



Design objectives

Positive structural design

Optimize **affinity** of designs towards target structure(s)

Examples: Most stable sequence for given fold. . .

Negative structural design

Limit affinity of designs towards **alternative structures**

Examples: Lowest free-energy, High Boltzmann probability/Low entropy. . .

Additional constraints:

- **Forbid** motif list to appear **anywhere** in design
- **Force** motif list to appear **each at least once**
- **Limit** available alternatives at certain positions
- **Control** overall composition (GC-content)



Outline

- I. Single Structure Design (IncaRNAation)
- II. Constrained Design using Formal Languages
- III. Multiple Structures



I. Inverse Folding

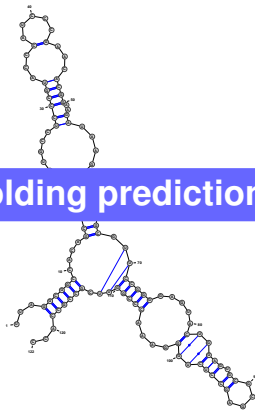
Designing a given structure

RNA sequence and structure(s)

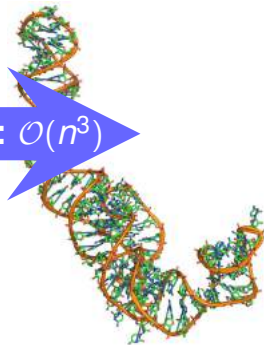
RNA = Linear Polymer = Sequence in {A, C, G, U}^{*}

```
UUAGGCGGCCACAGC
GGUGGGGUUGCCUCC
CGUACCAUCCCGAA
CACGGAAGAUAAAGCC
CACCAGCGUCCGGG
GAGUACUGGAGUGCG
CGAGCCUCUGGGAAA
CCCGGUUCGCCGCCA
CC
```

Primary Structure



Secondary Structure



Tertiary Structure

MFE folding prediction: $\mathcal{O}(n^3)$

5s rRNA (PDBID: 1K73:B)



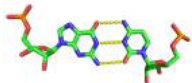
Crossing interactions

Excluded from the secondary structure:

- **Non-canonical base-pairs:**

Any base-pair **other than** {(A-U), (C-G), (G-U)}

OR interacting in a non-standard way (WC/WC-Cis) [Leontis Westhof, RNA 2001].

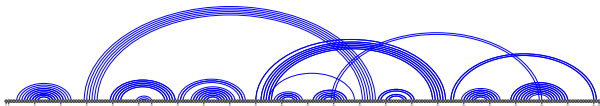


Canonical CG base-pair (WC/WC-Cis)



Non-canonical base-pair (Sugar/WC-Trans)

- **(Pseudo?)knots:** Crossing sets of nested stable base-pairs



Group I Ribozyme (PDBID: 1Y0Q:A)

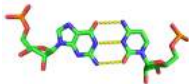
Crossing interactions

Excluded from the secondary structure:

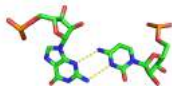
- **Non-canonical base-pairs:**

Any base-pair **other than** {(A-U), (C-G), (G-U)}

OR interacting in a non-standard way (WC/WC-Cis) [Leontis Westhof, RNA 2001].

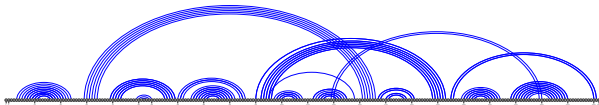


Canonical CG base-pair (WC/WC-Cis)



Non-canonical base-pair (Sugar/WC-Trans)

- **(Pseudo?)knots:** Crossing sets of nested stable base-pairs



Group I Ribozyme (PDBID: 1Y0Q:A)

Crossing interactions

Excluded from the secondary structure:

- **Non-canonical base-pairs:**

Any
OR
200



Crossing interactions do exist!

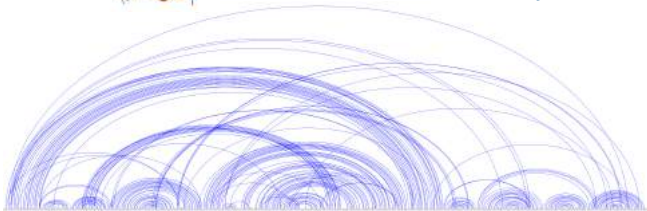
Example: Group II Intron (PDB ID: 3IGI)

But are **hard** to predict

[Lyngsoe-ICALP'04]

[Sheikh Backofen Ponty, CPM'12]

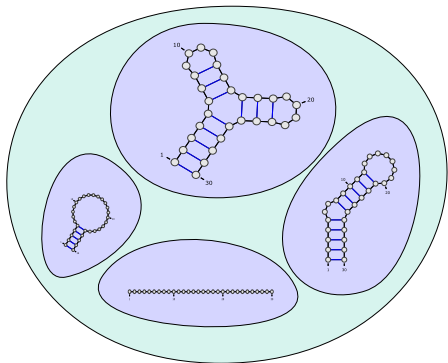
- (Ps



Thermodynamics vs Kinetics

Paradigms for RNA structure prediction

- **1978–1990s** Most probable structure = Minimal Free-Energy (MFE)
- **1990s–2010s** Functional structure(s) = Boltzmann ensemble (partition function)
- **2010s–????** Embracing the kinetics of RNA folding



$T \rightarrow \infty$

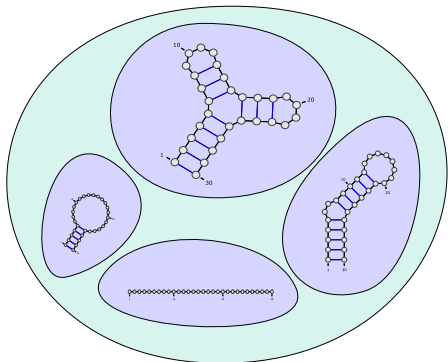
mRNA half-life: $\sim 7\text{h}$
(Mouse [Sharova2009])



Thermodynamics vs Kinetics

Paradigms for RNA structure prediction

- **1978–1990s** Most probable structure = Minimal Free-Energy (MFE)
- **1990s–2010s** Functional structure(s) = Boltzmann ensemble (partition function)
- **2010s–????** Embracing the kinetics of RNA folding



$$T \rightarrow \infty$$

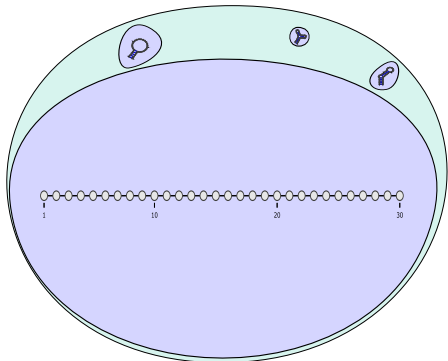
mRNA half-life: $\sim 7\text{h}$
(Mouse [Sharova2009])



Thermodynamics vs Kinetics

Paradigms for RNA structure prediction

- **1978–1990s** Most probable structure = Minimal Free-Energy (MFE)
- **1990s–2010s** Functional structure(s) = Boltzmann ensemble (partition function)
- **2010s–????** Embracing the kinetics of RNA folding



$T = 0h$

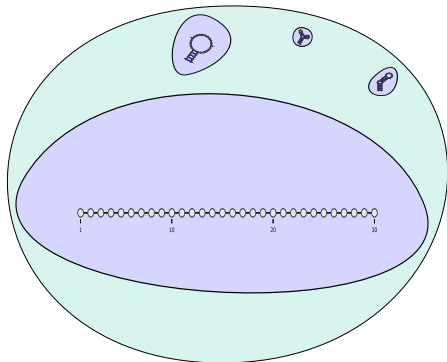
mRNA half-life: $\sim 7h$
(Mouse [Sharova2009])



Thermodynamics vs Kinetics

Paradigms for RNA structure prediction

- **1978–1990s** Most probable structure = Minimal Free-Energy (MFE)
- **1990s–2010s** Functional structure(s) = Boltzmann ensemble (partition function)
- **2010s–????** Embracing the kinetics of RNA folding



$T = 1h$

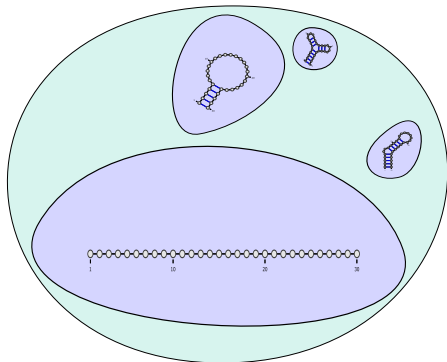
mRNA half-life: $\sim 7h$
(Mouse [Sharova2009])



Thermodynamics vs Kinetics

Paradigms for RNA structure prediction

- **1978–1990s** Most probable structure = Minimal Free-Energy (MFE)
- **1990s–2010s** Functional structure(s) = Boltzmann ensemble (partition function)
- **2010s–????** Embracing the kinetics of RNA folding



$T = 2h$

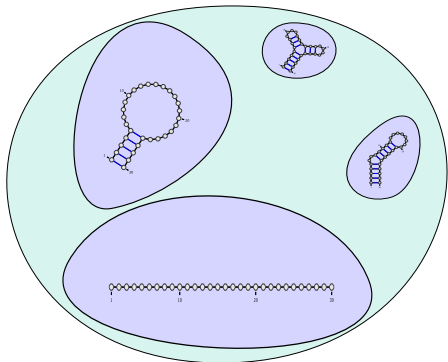
mRNA half-life: $\sim 7h$
(Mouse [Sharova2009])



Thermodynamics vs Kinetics

Paradigms for RNA structure prediction

- **1978–1990s** Most probable structure = Minimal Free-Energy (MFE)
- **1990s–2010s** Functional structure(s) = Boltzmann ensemble (partition function)
- **2010s–????** Embracing the kinetics of RNA folding



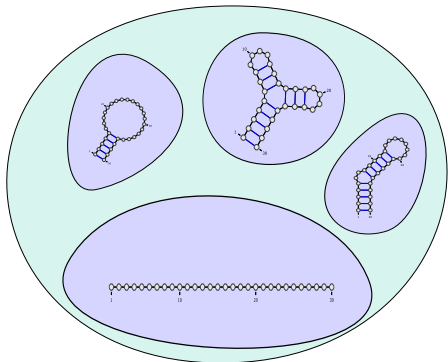
mRNA half-life: $\sim 7h$
(Mouse [Sharova2009])



Thermodynamics vs Kinetics

Paradigms for RNA structure prediction

- **1978–1990s** Most probable structure = Minimal Free-Energy (MFE)
- **1990s–2010s** Functional structure(s) = Boltzmann ensemble (partition function)
- **2010s–????** Embracing the kinetics of RNA folding



$T = 10h$

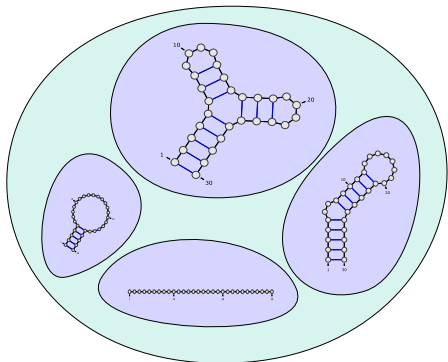
mRNA half-life: $\sim 7h$
(Mouse [Sharova2009])



Thermodynamics vs Kinetics

Paradigms for RNA structure prediction

- **1978–1990s** Most probable structure = Minimal Free-Energy (MFE)
- **1990s–2010s** Functional structure(s) = Boltzmann ensemble (partition function)
- **2010s–????** Embracing the kinetics of RNA folding



$T \rightarrow \infty$

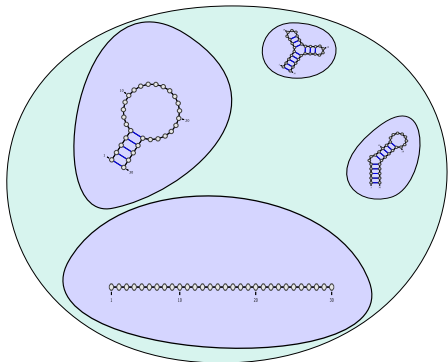
mRNA half-life: $\sim 7\text{h}$
(Mouse [Sharova2009])



Thermodynamics vs Kinetics

Paradigms for RNA structure prediction

- **1978–1990s** Most probable structure = Minimal Free-Energy (MFE)
- **1990s–2010s** Functional structure(s) = Boltzmann ensemble (partition function)
- **2010s–????** Embracing the kinetics of RNA folding

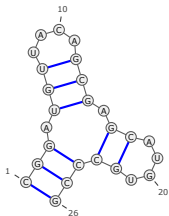
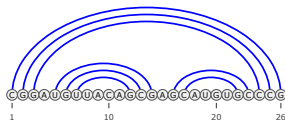


$T = 10h$

mRNA half-life: $\sim 7h$
(Mouse [Sharova2009])

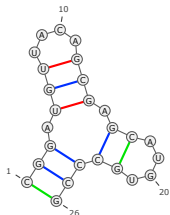
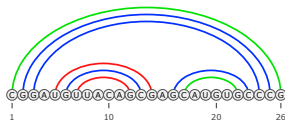


Problem statement



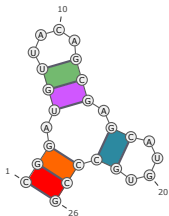
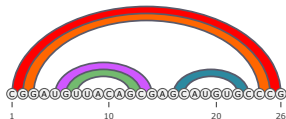
- **RNA structure S :** Non-crossing base-pairs for positions in sequence w
- **Motifs:** Sequence/structure features (e.g. Base-pairs, Stacks, Loops...)
- **Energy model:**
 - Motif** \rightarrow Free-energy contribution $\Delta(\cdot) \in \mathbb{R}^- \cup \{+\infty\}$
 - Free-Energy $E_w(S)$:** Sum over (independently contributing) motifs in S

Problem statement



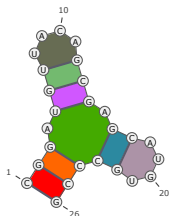
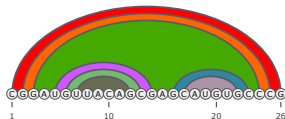
- **RNA structure S :** Non-crossing base-pairs for positions in sequence w
- **Motifs:** Sequence/structure features (e.g. Base-pairs, Stacks, Loops...)
- **Energy model:**
 - Motif \rightarrow Free-energy contribution $\Delta(\cdot) \in \mathbb{R}^- \cup \{+\infty\}$
 - Free-Energy $E_w(S)$: Sum over (independently contributing) motifs in S

Problem statement



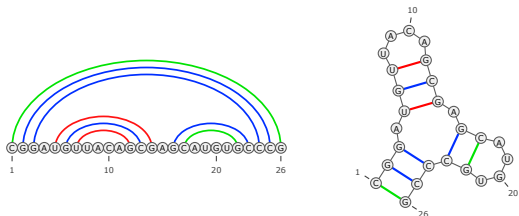
- **RNA structure S :** Non-crossing base-pairs for positions in sequence w
- **Motifs:** Sequence/structure features (e.g. Base-pairs, Stacks, Loops...)
- **Energy model:**
 - Motif \rightarrow Free-energy contribution $\Delta(\cdot) \in \mathbb{R}^- \cup \{+\infty\}$
 - Free-Energy $E_w(S)$: Sum over (independently contributing) motifs in S

Problem statement



- **RNA structure S :** Non-crossing base-pairs for positions in sequence w
- **Motifs:** Sequence/structure features (e.g. Base-pairs, Stacks, Loops. . .)
- **Energy model:**
 - Motif \rightarrow Free-energy contribution $\Delta(\cdot) \in \mathbb{R}^- \cup \{+\infty\}$
 - Free-Energy $E_w(S)$: Sum over (independently contributing) motifs in S

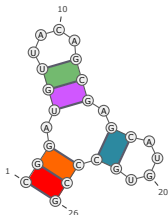
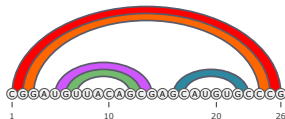
Problem statement



- **RNA structure S :** Non-crossing base-pairs for positions in sequence w
- **Motifs:** Sequence/structure features (e.g. Base-pairs, Stacks, Loops...)
- **Energy model:**
 - Motif** \rightarrow Free-energy contribution $\Delta(\cdot) \in \mathbb{R}^- \cup \{+\infty\}$
 - Free-Energy $E_w(S)$:** Sum over (independently contributing) motifs in S

$$E_S = 2 \cdot \Delta \left(\begin{array}{c} \text{U} \\ | \\ \text{G} \end{array} \right) + 4 \cdot \Delta \left(\begin{array}{c} \text{G} \\ | \\ \text{C} \end{array} \right) + 2 \cdot \Delta \left(\begin{array}{c} \text{C} \\ | \\ \text{G} \end{array} \right)$$

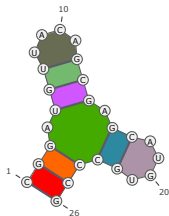
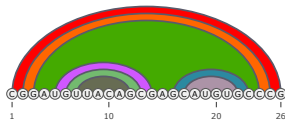
Problem statement



- **RNA structure S :** Non-crossing base-pairs for positions in sequence w
- **Motifs:** Sequence/structure features (e.g. Base-pairs, Stacks, Loops. . .)
- **Energy model:**
 - Motif** \rightarrow Free-energy contribution $\Delta(\cdot) \in \mathbb{R}^- \cup \{+\infty\}$
 - Free-Energy $E_w(S)$:** Sum over (independently contributing) motifs in S

$$E_S = \Delta \left(\begin{array}{cc} \text{C} & \text{G} \\ \text{G} & \text{C} \end{array} \right) + \Delta \left(\begin{array}{cc} \text{G} & \text{G} \\ \text{C} & \text{C} \end{array} \right) + \Delta \left(\begin{array}{cc} \text{U} & \text{G} \\ \text{G} & \text{C} \end{array} \right) + \Delta \left(\begin{array}{cc} \text{U} & \text{G} \\ \text{G} & \text{C} \end{array} \right) + \Delta \left(\begin{array}{cc} \text{U} & \text{G} \\ \text{G} & \text{C} \end{array} \right)$$

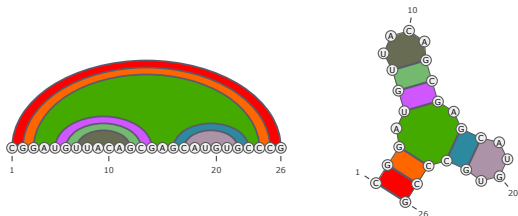
Problem statement



- **RNA structure S :** Non-crossing base-pairs for positions in sequence w
- **Motifs:** Sequence/structure features (e.g. Base-pairs, Stacks, Loops...)
- **Energy model:**
 - Motif** \rightarrow Free-energy contribution $\Delta(\cdot) \in \mathbb{R}^- \cup \{+\infty\}$
 - Free-Energy $E_w(S)$:** Sum over (independently contributing) motifs in S

$$\begin{aligned}
 E_S = & \Delta \left(\begin{array}{c} \text{C} \quad \text{G} \\ | \quad | \\ \text{G} \quad \text{C} \end{array} \right) + \Delta \left(\begin{array}{c} \text{G} \quad \text{G} \\ | \quad | \\ \text{C} \quad \text{C} \end{array} \right) + \Delta \left(\begin{array}{c} \text{U} \quad \text{G} \\ | \quad | \\ \text{G} \quad \text{C} \end{array} \right) + \Delta \left(\begin{array}{c} \text{U} \quad \text{G} \\ | \quad | \\ \text{G} \quad \text{C} \end{array} \right) + \Delta \left(\begin{array}{c} \text{U} \quad \text{G} \\ | \quad | \\ \text{G} \quad \text{C} \end{array} \right) \\
 & + \Delta \left(\begin{array}{c} \text{A} \quad \text{C} \quad \text{A} \\ / \quad | \quad \backslash \\ \text{U} \quad \quad \text{G} \end{array} \right) + \Delta \left(\begin{array}{c} \text{U} \quad \text{G} \quad \text{A} \\ / \quad | \quad \backslash \\ \text{G} \quad \quad \text{C} \end{array} \right) + \Delta \left(\begin{array}{c} \text{C} \quad \text{A} \\ / \quad \backslash \\ \text{U} \quad \text{G} \end{array} \right)
 \end{aligned}$$

Problem statement



- **RNA structure S :** Non-crossing base-pairs for positions in sequence w
- **Motifs:** Sequence/structure features (e.g. Base-pairs, Stacks, Loops. . .)
- **Energy model:**
 - Motif** \rightarrow Free-energy contribution $\Delta(\cdot) \in \mathbb{R}^- \cup \{+\infty\}$
 - Free-Energy $E_w(S)$:** Sum over (independently contributing) motifs in S

Definition (MFE-PREDICT(E) problem)

Input: RNA sequence $w \in \{A, C, G, U\}^*$

Output: Secondary struct. S^* with Minimal Free-Energy (MFE) $E_w(S^*)$

Problem solved **exactly** in $O(n^3)$ time.

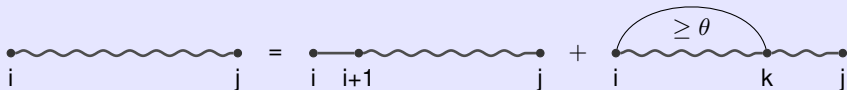
[Nussinov Jacobson, PNAS 1980] [Zuker Stiegler, NAR 1981]. . . .



Dynamic programming (DP) for RNA folding

Theorem ([Nussinov and Jacobson(1980)])

Max #base-pairs/min energy structure computed in $\mathcal{O}(n^3)/\mathcal{O}(n^2)$ time/memory



$E_{i,k}$: Free-energy contribution of base-pair (i, k) . $(-1 / +\infty$ or $\Delta G(s_i \stackrel{?}{=} s_k))$

$N_{i,j}$: Max #base-pairs over interval $[i, j]$

$$N_{i,t} = 0, \quad \forall t \in [i, i + \theta]$$

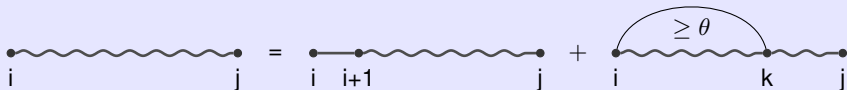
$$N_{i,j} = \min \begin{cases} N_{i+1,j} & \{i \text{ unpaired}\} \\ \min_{k=i+\theta+1}^j E_{i,k} + N_{i+1,k-1} + N_{k+1,j} & \{i \text{ paired to } k\} \end{cases}$$



Dynamic programming (DP) for RNA folding

Theorem ([Nussinov and Jacobson(1980)])

Max #base-pairs/min energy structure computed in $\mathcal{O}(n^3)/\mathcal{O}(n^2)$ time/memory



$E_{i,k}$: Free-energy contribution of base-pair (i, k) . $(-1 / +\infty$ or $\Delta G(s_i \stackrel{?}{=} s_k)$)

$C_{i,j}$: Number of secondary structures compatible with interval $[i, j]$

$$C_{i,t} = 1, \quad \forall t \in [i, i + \theta]$$

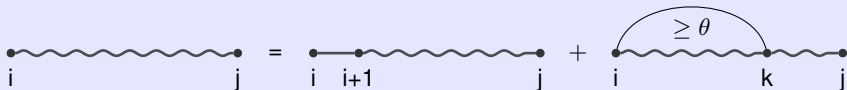
$$C_{i,j} = \sum \left\{ \begin{array}{ll} \sum_{k=i+\theta+1}^j \mathbb{1}_{\text{comp.}(i,k)} \times C_{i+1,j} & \{i \text{ unpaired}\} \\ \sum_{k=i+\theta+1}^j \mathbb{1}_{\text{comp.}(i,k)} \times C_{i+1,k-1} \times C_{k+1,j} & \{i \text{ paired to } k\} \end{array} \right.$$



Dynamic programming (DP) for RNA folding

Theorem ([Nussinov and Jacobson(1980)])

Max #base-pairs/min energy structure computed in $\mathcal{O}(n^3)/\mathcal{O}(n^2)$ time/memory



$E_{i,k}$: Free-energy contribution of base-pair (i, k) . $(-1 / +\infty$ or $\Delta G(s_i \stackrel{?}{=} s_k)$)

$Z_{i,j} = \sum_{\substack{S \text{ comp.} \\ \text{with } w_{[i,j]}}} e^{-\frac{E_w(S)}{RT}} =$ Partition function for compatible structs within $[i, j]$

$$Z_{i,t} = \mathbf{1}, \quad \forall t \in [i, i + \theta]$$

$$Z_{i,j} = \sum \left\{ \begin{array}{ll} \sum_{k=i+\theta+1}^j e^{-\frac{E_{i,k}}{RT}} \times Z_{i+1,j} & \{i \text{ unpaired}\} \\ \sum_{k=i+\theta+1}^j e^{-\frac{E_{i,k}}{RT}} \times Z_{i+1,k-1} \times Z_{k+1,j} & \{i \text{ paired to } k\} \end{array} \right.$$



Dynamic programming (DP) for RNA folding

Many extensions:

- Nearest-neighbor/Turner energy model [Zuker1981]
- Comparative folding [Sankoff1985]
- Equilibrium base-pairing probabilities [McCaskill1990]
- Moments of additive features [Miklos2005,Ponty2011]
- Δ kcal.mol⁻¹ suboptimal structures of MFE [Wuchty1999]
- Basic crossing structures [Rivas1999]. . .
- Exact sampling in Boltzmann distr. [Ding2003,Ponty2008]
- Moments of additive features [Miklos2005,Ponty2011]
- Maximum expected accuracy structure [Do2006]
- Distance-classified partitioning of Boltzmann ens. [E.Freyhult2007a]

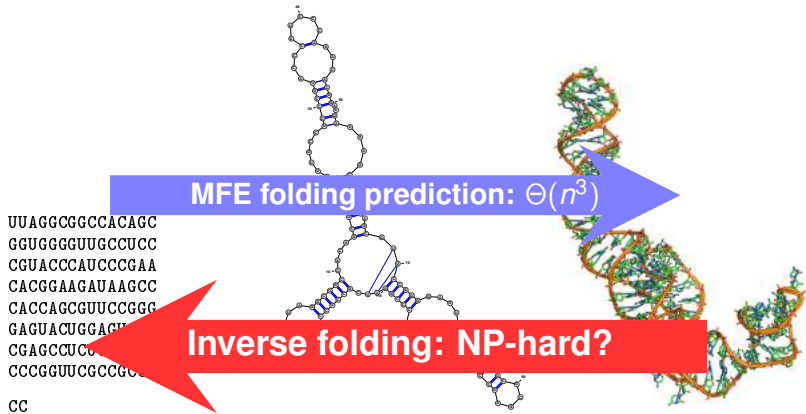
Made possible by:

- **Completeness/Unambiguity** of decomposition
∃ energy-preserving bijection between **derivations of DP scheme** and **search space**
- Objective function **additive** with respect to DP scheme



RNA inverse folding

RNA = Linear Polymer = Sequence in $\{A, C, G, U\}^*$



Primary Structure

Secondary Structure

Structure Tertiaire

5s rRNA (PDBID: 1K73:B)



RNA Inverse Folding

Definition (INVERSE-FOLDING(E) problem)

Input: Secondary structure S + Energy distance $\Delta > 0$.

Output: RNA sequence $w \in \Sigma^*$ such that:

$$\forall S' \in \mathcal{S}|w| \setminus \{S\} : E_{w,S'} \geq E_{w,S} + \Delta$$

or \emptyset if no such sequence exists.

Difficult problem: No **obvious** DP decomposition

- Existing algorithms: Heuristics or Exponential-time
- Complexity of problem unknown (despite [Schnall Levin *et al*, ICML'08])
Reason: Non locality, no theoretical frameworks, too many parameters...



RNA Inverse Folding

Definition (INVERSE-FOLDING(E) problem)

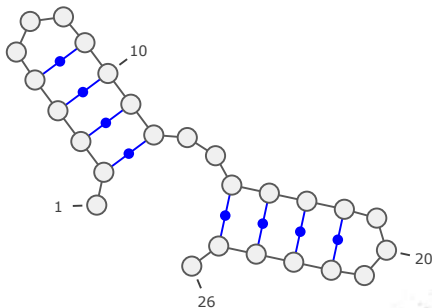
Input: Secondary structure S + Energy distance $\Delta > 0$.

Output: RNA sequence $w \in \Sigma^*$ such that:

$$\forall S' \in \mathcal{S}|w| \setminus \{S\} : E_{w,S'} \geq E_{w,S} + \Delta$$

or \emptyset if no such sequence exists.

Example:



RNA Inverse Folding

Definition (INVERSE-FOLDING(E) problem)

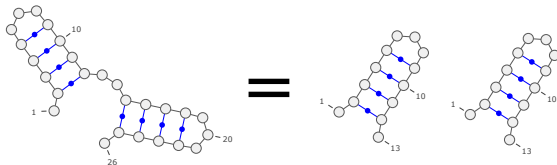
Input: Secondary structure S + Energy distance $\Delta > 0$.

Output: RNA sequence $w \in \Sigma^*$ such that:

$$\forall S' \in \mathcal{S}|w| \setminus \{S\} : E_{w,S'} \geq E_{w,S} + \Delta$$

or \emptyset if no such sequence exists.

Example:



RNA Inverse Folding

Definition (INVERSE-FOLDING(E) problem)

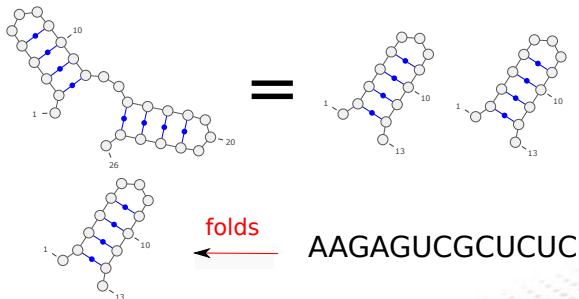
Input: Secondary structure S + Energy distance $\Delta > 0$.

Output: RNA sequence $w \in \Sigma^*$ such that:

$$\forall S' \in \mathcal{S}|w| \setminus \{S\} : E_{w,S'} \geq E_{w,S} + \Delta$$

or \emptyset if no such sequence exists.

Example:



RNA Inverse Folding

Definition (INVERSE-FOLDING(E) problem)

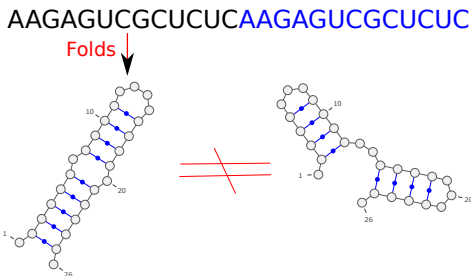
Input: Secondary structure S + Energy distance $\Delta > 0$.

Output: RNA sequence $w \in \Sigma^*$ such that:

$$\forall S' \in \mathcal{S}|w| \setminus \{S\} : E_{w,S'} \geq E_{w,S} + \Delta$$

or \emptyset if no such sequence exists.

Example:



Existing approaches for negative design

Based on local search...

- RNAInverse - TBI Vienna
- Info-RNA - Backofen@Freiburg
- RNA-SSD - Condon@UBC
- NUPack - Pierce@Caltech

... bio-inspired algorithms...

- RNAFBinv - Barash@Ben Gurion
- FRNAKenstein - Hein@Oxford
- AntaRNA - Backofen@Freiburg
- ERD - Ganjtabesh@Tehran

... exact approaches...

- RNAIFold - Clote@Boston College
- CO4 - Will@Leipzig

Typical issues:

- Naive initialization strategies
- Poor coverage of sequence space:
Local search remain *confined* near initial sequence
- GC-rich produced sequences

⇒ **Global sampling** [Levin *et al*, NAR 12]



Existing approaches for negative design

Based on local search...

- RNAInverse - TBI Vienna
- Info-RNA - Backofen@Freiburg
- RNA-SSD - Condon@UBC
- NUPack - Pierce@Caltech

... bio-inspired algorithms...

- RNAFBinv - Barash@Ben Gurion
- FRNAKenstein - Hein@Oxford
- AntaRNA - Backofen@Freiburg
- ERD - Ganjtabesh@Tehran

... exact approaches...

- RNAIFold - Clote@Boston College
- CO4 - Will@Leipzig

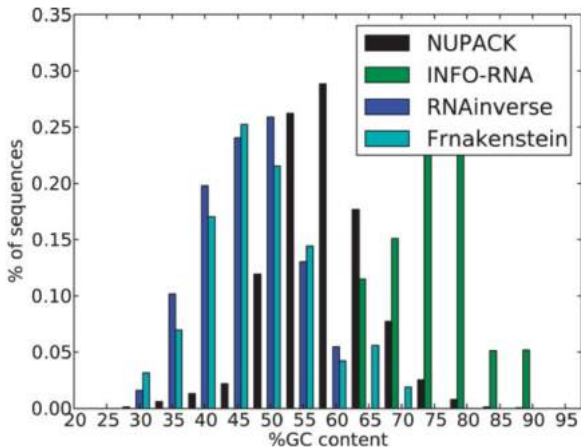
Typical issues:

- Naive initialization strategies
- Poor coverage of sequence space:
Local search remain *confined* near initial sequence
- GC-rich produced sequences

⇒ **Global sampling** [Levin *et al*, NAR 12]



The case for a control of GC-content



High GC-content suspected to induce **kinetic traps**

Global sampling [Levin *et al*, NAR 12]

Target structure S

- **Boltzmann distribution** based on **affinity** towards S
- **Random generation** from **Boltzmann Distribution**
- **Fold** sampled sequences and **compare** to target

Boltzmann factor:

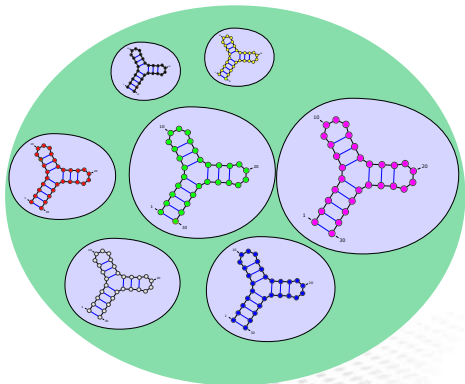
$$\mathcal{B}_w(S) := e^{-\frac{E_w(S)}{RT}}$$

Pseudo-Partition Function:

$$\mathcal{Z}(S) = \sum_{w \in \Sigma^*} \mathcal{B}_w(S)$$

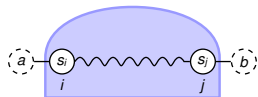
Boltzmann probability:

$$p(s) := \frac{\mathcal{B}_w(S)}{\mathcal{Z}}$$



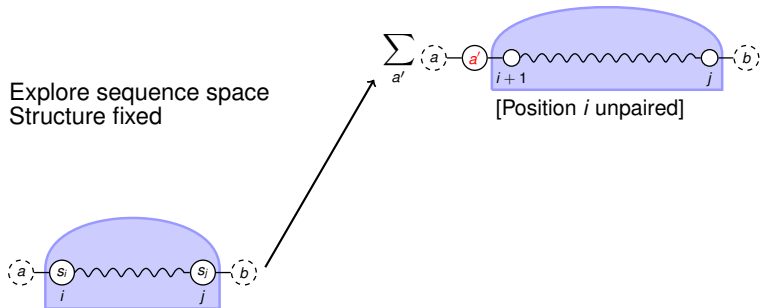
IncaRNAtion [Reinharz *et al*, Bioinformatics 2013]

Explore sequence space
Structure fixed



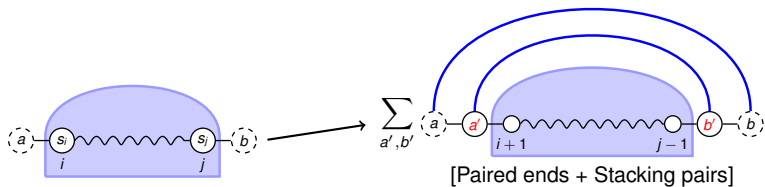
IncaRNation [Reinharz *et al*, Bioinformatics 2013]

Explore sequence space
Structure fixed



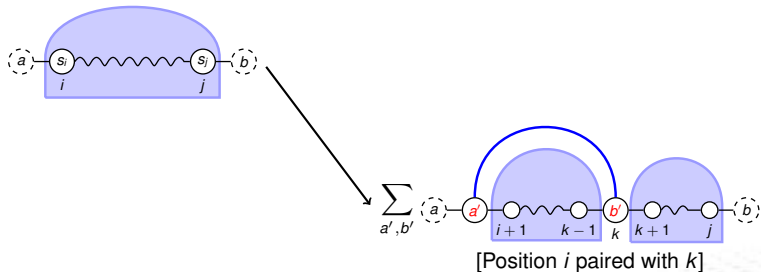
IncaRNation [Reinharz *et al*, Bioinformatics 2013]

Explore sequence space
Structure fixed



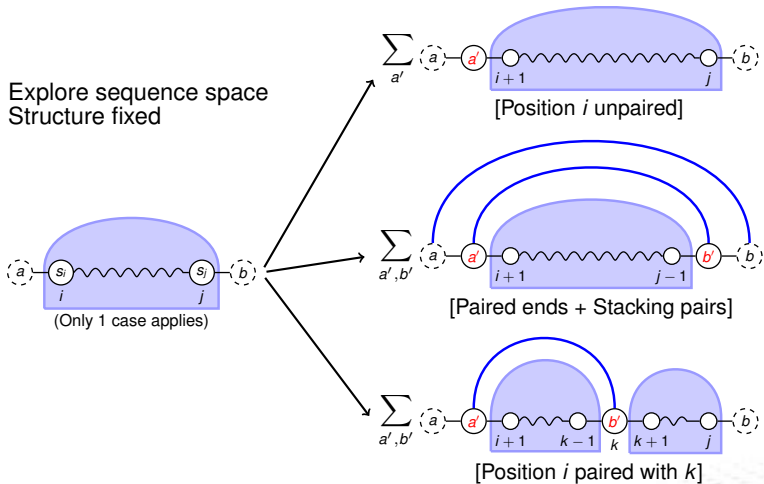
IncaRNation [Reinharz *et al*, Bioinformatics 2013]

Explore sequence space
Structure fixed



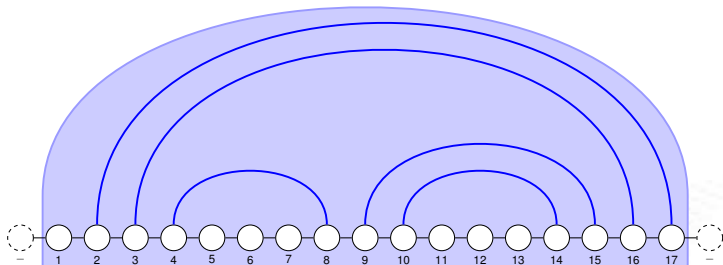
IncaRNAtion [Reinharz *et al*, Bioinformatics 2013]

Explore sequence space
Structure fixed



"Global" Stochastic Backtrack

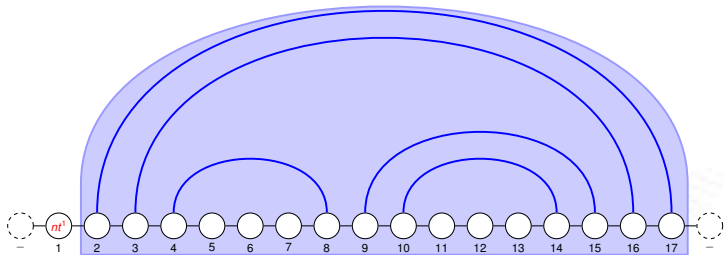
Sequence:



"Global" Stochastic Backtrack

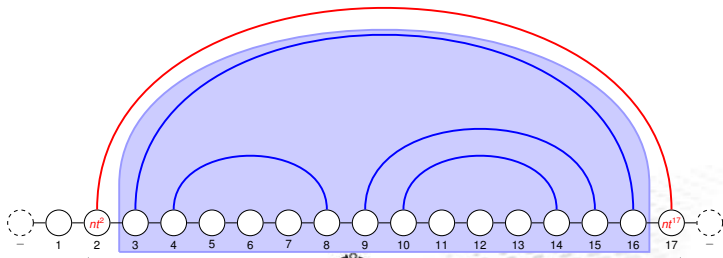
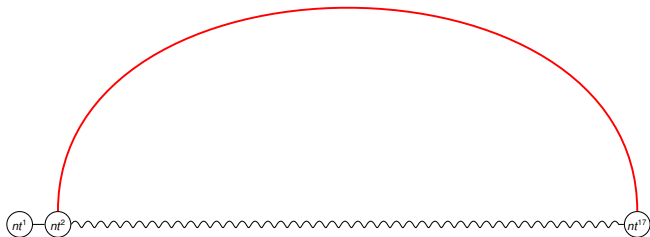
Sequence:

nt^1



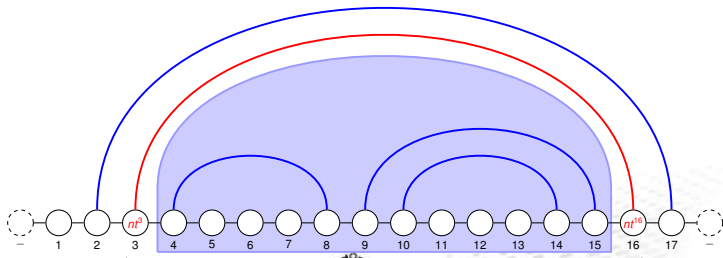
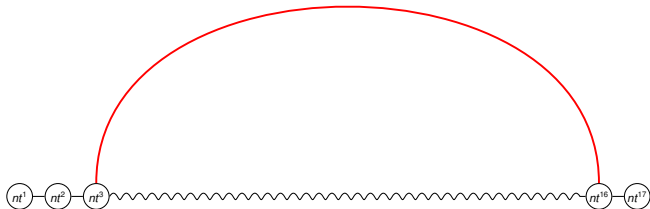
"Global" Stochastic Backtrack

Sequence:



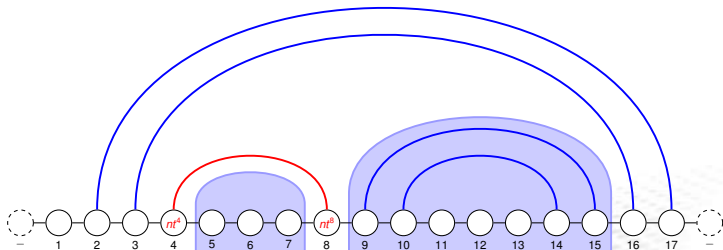
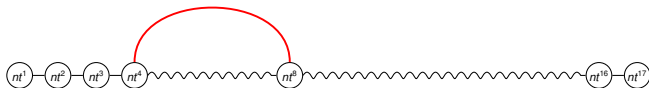
"Global" Stochastic Backtrack

Sequence:



"Global" Stochastic Backtrack

Sequence:

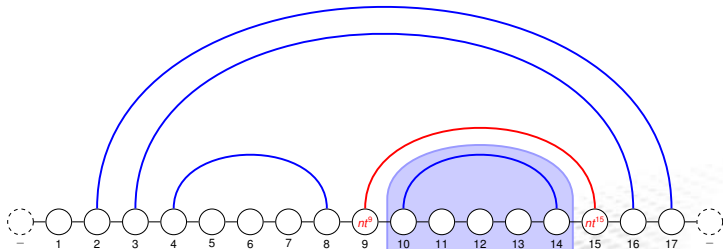
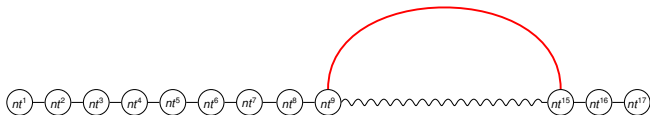


$$Z = \text{weight of } nt^4 \wedge nt^8$$



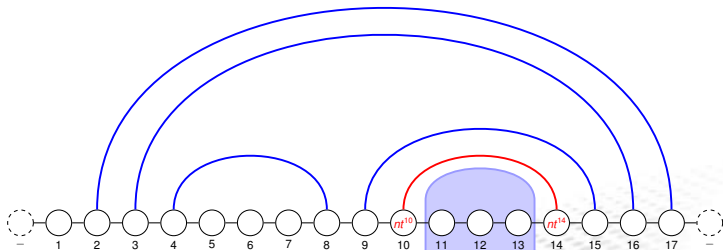
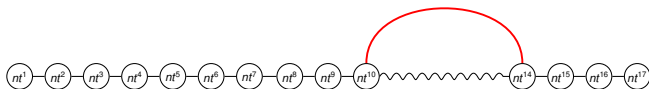
"Global" Stochastic Backtrack

Sequence:

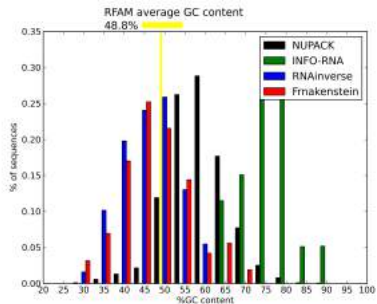
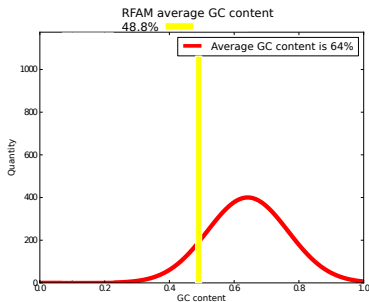


"Global" Stochastic Backtrack

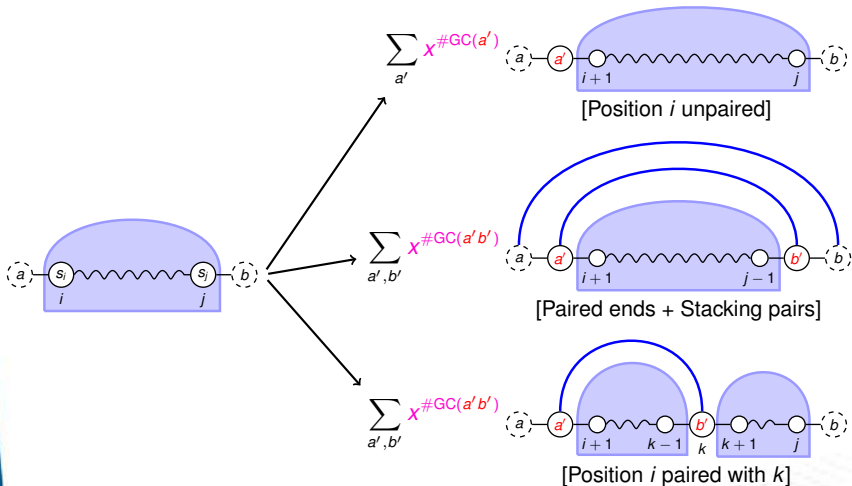
Sequence:



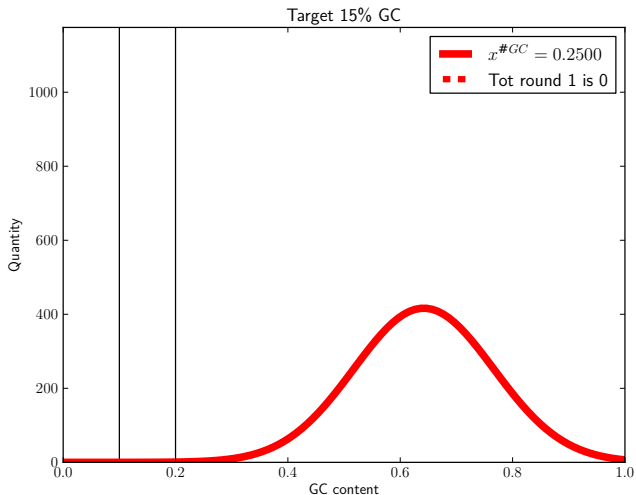
GC-content bias



Weighted DP Recursions



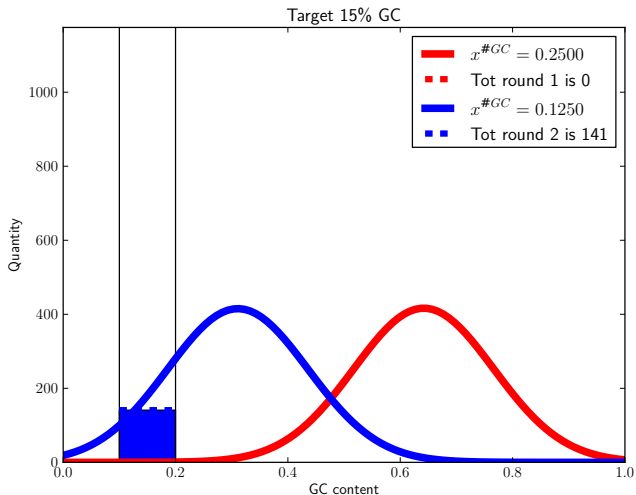
Incarnation NT distribution: Bisection scheme



[Waldispühl and Ponty, RECOMB, 2011]



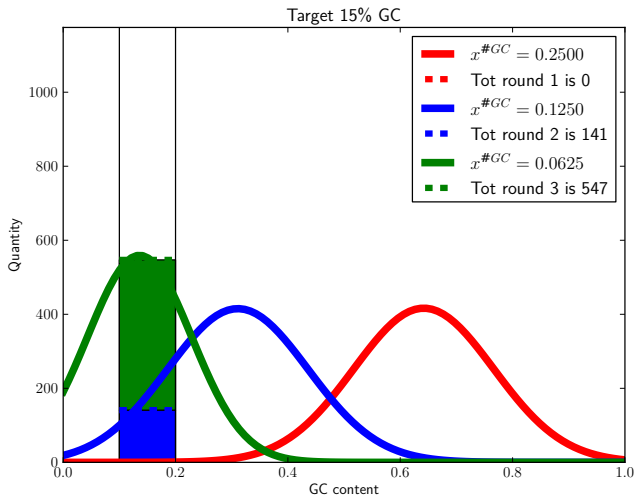
Incarnation NT distribution: Bisection scheme



[Waldispühl and Ponty, RECOMB, 2011]



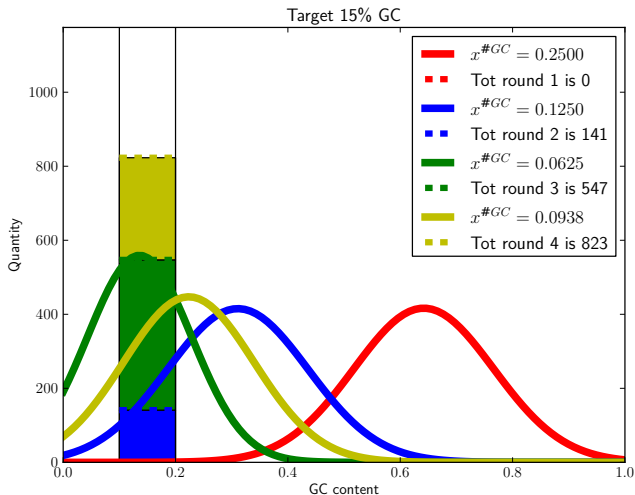
Incarnation NT distribution: Bisection scheme



[Waldispühl and Ponty, RECOMB, 2011]



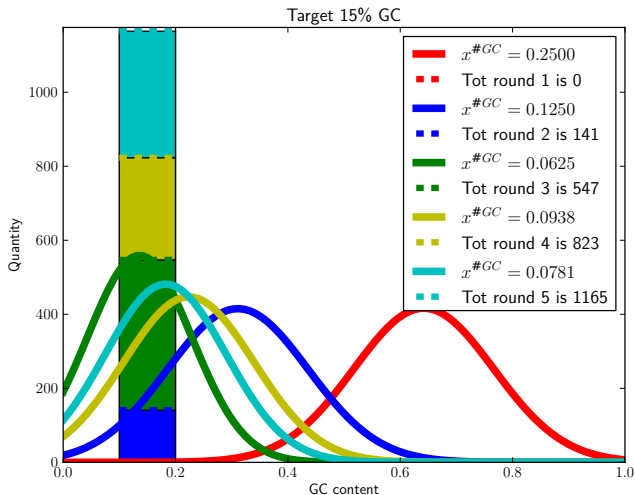
Incarnation NT distribution: Bisection scheme



[Waldispühl and Ponty, RECOMB, 2011]



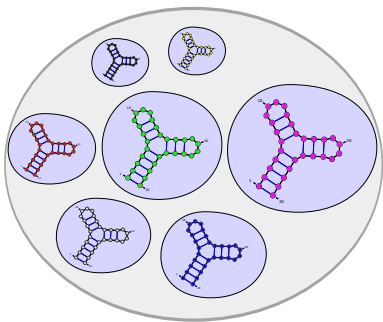
Incarnation NT distribution: Bisection scheme



[Waldispühl and Ponty, RECOMB, 2011]



Limits of the approach

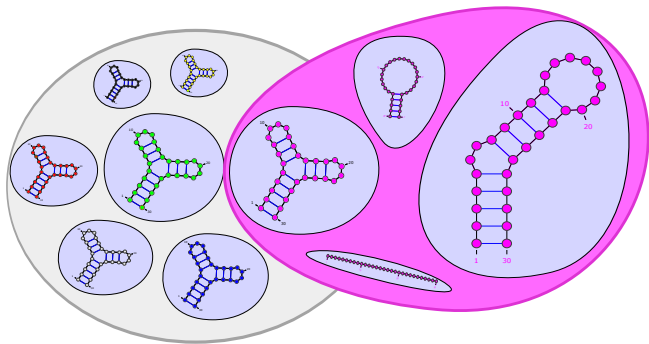


Heuristic: Strong affinity is neither sufficient, nor necessary, **but** . . .

- Strong **empirical** correlation affinity/success of design [Levin et al, NAR 2012]
- Linear time-complexity [Reinharz Ponty Waldispühl, ISMB/ECCB'13]
- **Composition** control [Bodini Ponty, AofA'10] [Reinharz et al, ISMB/ECCB'13]
- **Complementary** with local search approaches [Reinharz et al, ISMB/ECCB'13]



Limits of the approach

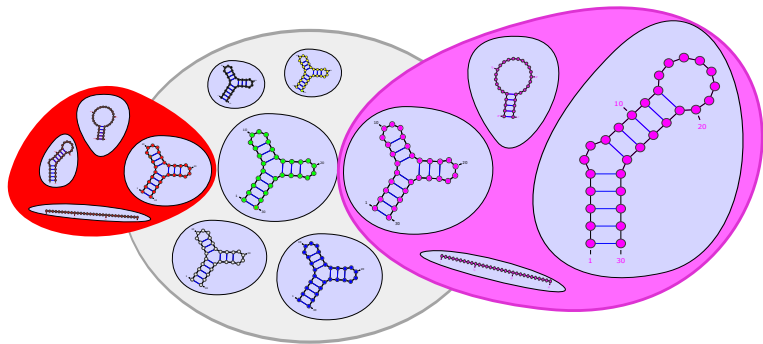


Heuristic: Strong affinity is **neither sufficient**, nor necessary, **but** ...

- Strong **empirical** correlation affinity/success of design [Levin et al, NAR 2012]
- Linear time-complexity [Reinharz Ponty Waldispühl, ISMB/ECCB'13]
- **Composition** control [Bodini Ponty, AofA'10] [Reinharz et al, ISMB/ECCB'13]
- **Complementary** with local search approaches [Reinharz et al, ISMB/ECCB'13]



Limits of the approach

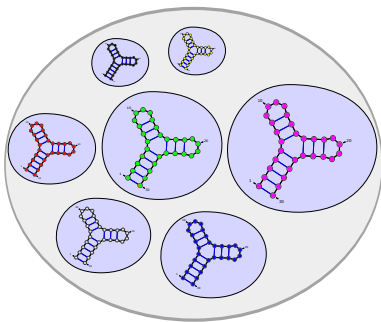


Heuristic: Strong affinity is **neither sufficient, nor necessary, but ...**

- Strong **empirical** correlation affinity/success of design [Levin et al, NAR 2012]
- Linear time-complexity [Reinharz Ponty Waldispühl, ISMB/ECCB'13]
- **Composition** control [Bodini Ponty, AofA'10] [Reinharz et al, ISMB/ECCB'13]
- **Complementary** with local search approaches [Reinharz et al, ISMB/ECCB'13]



Limits of the approach

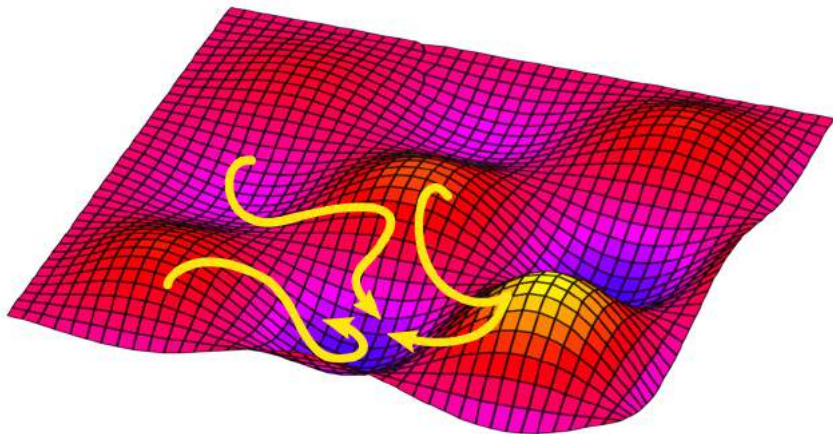


Heuristic: Strong affinity is **neither sufficient, nor necessary**, but ...

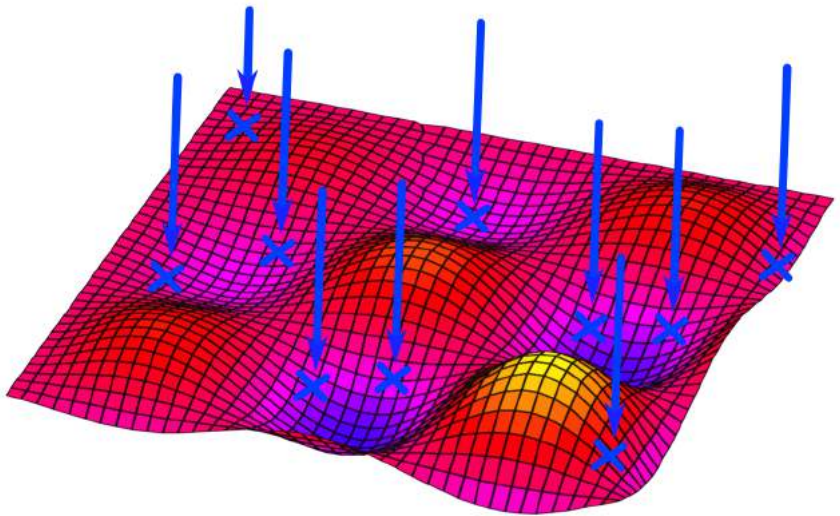
- Strong **empirical** correlation affinity/success of design [Levin et al, NAR 2012]
- **Linear** time-complexity [Reinharz Ponty Waldispühl, ISMB/ECCB'13]
- **Composition** control [Bodini Ponty, AofA'10] [Reinharz et al, ISMB/ECCB'13]
- **Complementary** with local search approaches [Reinharz et al, ISMB/ECCB'13]



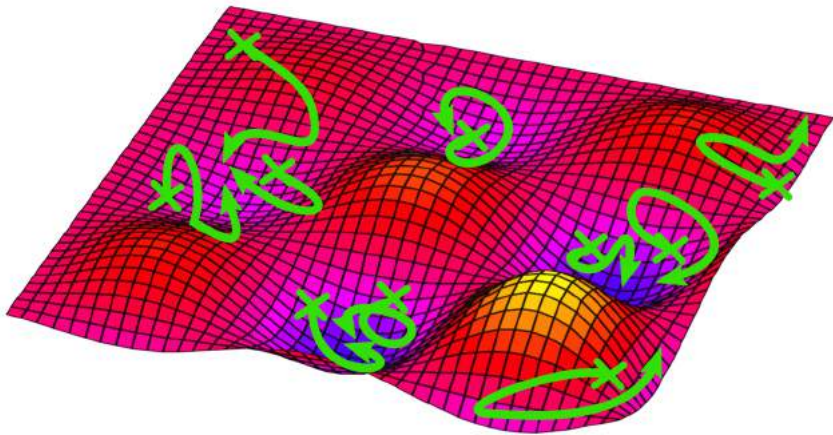
Local vs Global vs "Glocal"



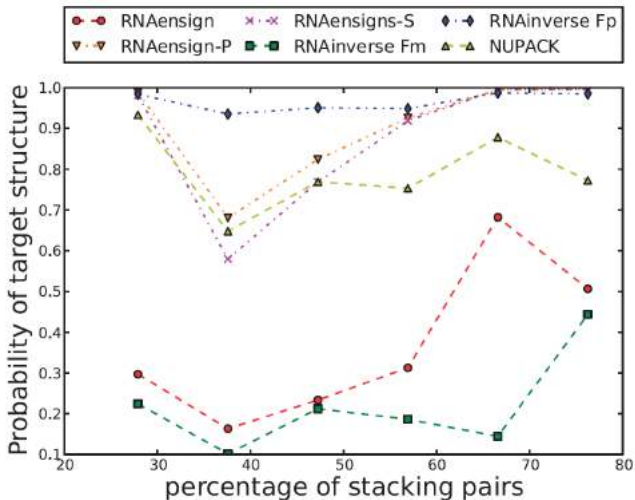
Local vs Global vs "Glocal"



Local vs Global vs "Glocal"



The success of glocal strategies



Sampling + Optimize creates highly probable design sequences



II. Constrained design

Avoiding/forcing motifs

Existing approaches for negative design

Based on local search...

- RNAInverse - TBI Vienna
- Info-RNA -
Backofen@Freiburg
- RNA-SSD - Condon@UBC
- NUPack - Pierce@Caltech

... bio-inspired algorithms...

- RNAFBinv - Barash@Ben Gurion
- FRNAKenstein - Hein@Oxford
- AntaRNA - Backofen@Freiburg

... exact approaches...

- RNAIFold - Clote@Boston College
- CO4 - Will@Leipzig

Few algorithms support avoided/mandatory motifs...

... none guarantees *reasonable* runtime.

Typical reasons:

- Deep local minima (Rugged landscape)
- Mandatory motifs \Rightarrow Late deadends (Branch and Bound)
- Forbidden motifs \Rightarrow Search space disconnection (Local Search)



Existing approaches for negative design

Based on local search...

- RNAInverse - TBI Vienna
- Info-RNA -
Backofen@Freiburg
- RNA-SSD - Condon@UBC
- NUPack - Pierce@Caltech

... bio-inspired algorithms...

- RNAFBinv - Barash@Ben Gurion
- FRNAKenstein - Hein@Oxford
- AntaRNA - Backofen@Freiburg

... exact approaches...

- RNAIFold - Clote@Boston College
- CO4 - Will@Leipzig

Few algorithms support avoided/mandatory motifs...

... none guarantees *reasonable* runtime.

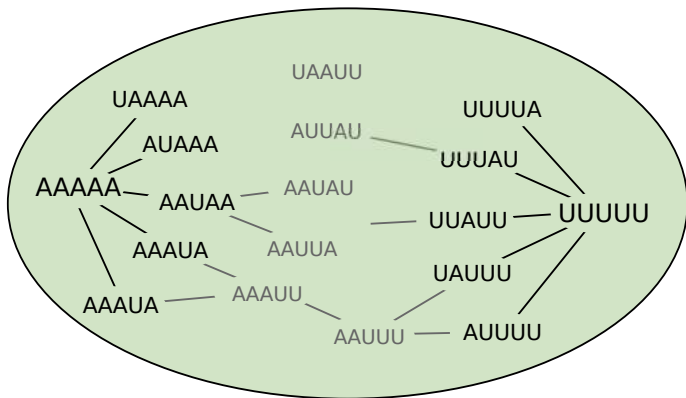
Typical reasons:

- Deep local minima (Rugged landscape)
- Mandatory motifs \Rightarrow Late deadends (Branch and Bound)
- Forbidden motifs \Rightarrow Search space disconnection (Local Search)



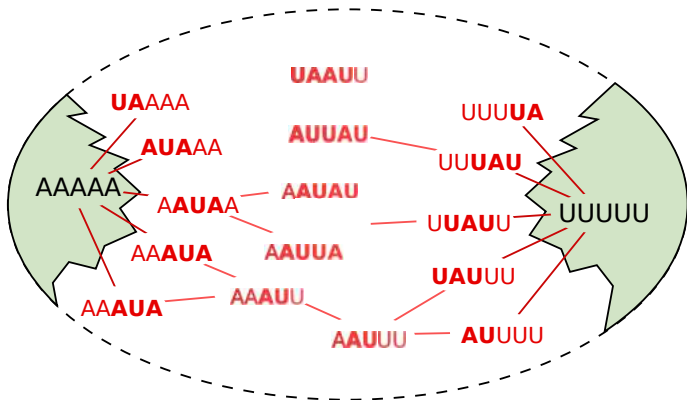
Problem with local approaches: An example

Simplified vocabulary {A, U}



Problem with local approaches: An example

Simplified vocabulary $\{A, U\}$ + **Forbidden motifs** $\mathcal{F} = \{AU, UA\}$



$\Rightarrow \mathcal{F}$ may **disconnect** search space (holds for **any** move set!)

Idea

Use formal language constructs to constrain global sampling

Forced motifs
Avoided motifs \rightarrow Regular language $\mathcal{L}_C \in \text{Reg}$

Structure compatibility
+ Positional constraints \rightarrow **Weighted** Context-Free Lang $\mathcal{L}_S \in \text{CFL}$
+ Energy Model

Folklore theorem (constructive): $\text{Reg} \cap (\mathbf{W})\text{CFL} \subseteq (\mathbf{W})\text{CFL}$

Build weighted context-free grammar \mathcal{G} for $\mathcal{L}_C \cap \mathcal{L}_S$
+ Random generation

\Rightarrow **Global sampling under constraints**



Idea

Use formal language constructs to constrain global sampling

Forced motifs
Avoided motifs \rightarrow Regular language $\mathcal{L}_C \in \text{Reg}$

Structure compatibility
+ Positional constraints \rightarrow **Weighted** Context-Free Lang $\mathcal{L}_S \in \text{CFL}$
+ Energy Model

Folklore theorem (constructive): $\text{Reg} \cap (\mathbf{W})\text{CFL} \subseteq (\mathbf{W})\text{CFL}$

Build weighted context-free grammar \mathcal{G} for $\mathcal{L}_C \cap \mathcal{L}_S$
+ Random generation

\Rightarrow Global sampling under constraints



Idea

Use formal language constructs to constrain global sampling

Forced motifs
Avoided motifs \rightarrow Regular language $\mathcal{L}_C \in \text{Reg}$

Structure compatibility
+ Positional constraints \rightarrow **Weighted** Context-Free Lang $\mathcal{L}_S \in \text{CFL}$
+ Energy Model

Folklore theorem (constructive): $\text{Reg} \cap (\mathbf{W})\text{CFL} \subseteq (\mathbf{W})\text{CFL}$

Build weighted context-free grammar \mathcal{G} for $\mathcal{L}_C \cap \mathcal{L}_S$
+ Random generation

\Rightarrow Global sampling under constraints



Idea

Use formal language constructs to constrain global sampling

Forced motifs
Avoided motifs \rightarrow Regular language $\mathcal{L}_C \in \text{Reg}$

Structure compatibility
+ Positional constraints \rightarrow **Weighted** Context-Free Lang $\mathcal{L}_S \in \text{CFL}$
+ Energy Model

Folklore theorem (constructive): $\text{Reg} \cap (\mathbf{W})\text{CFL} \subseteq (\mathbf{W})\text{CFL}$

**Build weighted context-free grammar \mathcal{G} for $\mathcal{L}_C \cap \mathcal{L}_S$
+ Random generation**

\Rightarrow **Global sampling under constraints**



Building the Finite State Automaton

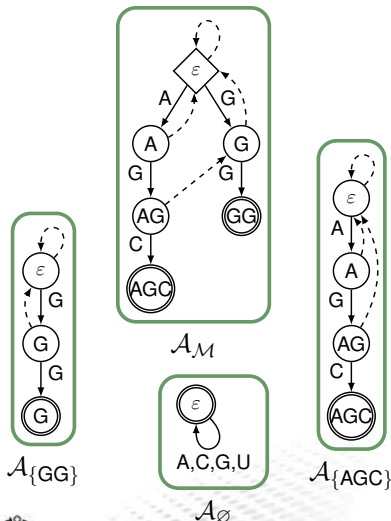
To force multiple words, **keep track** of generated words:

- Create disjunctive automata for each $\mathcal{M}' \subseteq \mathcal{M}$
- Reroute accepting states
- Accepting state = no **forced word** remaining (ϵ in \mathcal{A}_\emptyset)
- Forbidden words can be added to sub-automata

#States:

$$O\left(2^{|\mathcal{M}|} \cdot (\sum_i |f_i| + \sum_j |m_j|)\right)$$

Example: $\mathcal{M} = \{AGC, GG\}$



Building the Finite State Automaton

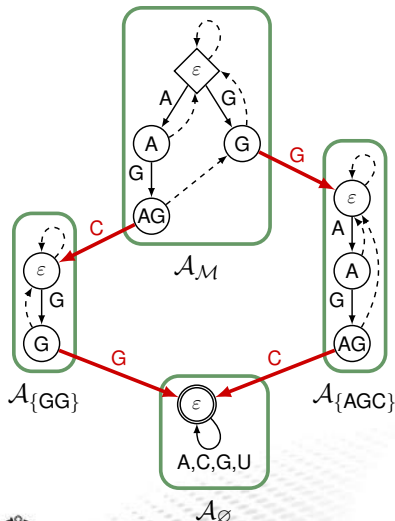
To force multiple words, **keep track** of generated words:

- Create disjunctive automata for each $\mathcal{M}' \subseteq \mathcal{M}$
- **Reroute** accepting states
- Accepting state = no **forced word** remaining (ϵ in \mathcal{A}_\emptyset)
- Forbidden words can be added to sub-automata

#States:

$$O\left(2^{|\mathcal{M}|} \cdot (\sum_i |f_i| + \sum_j |m_j|)\right)$$

Example: $\mathcal{M} = \{AGC, GG\}$



Building the Finite State Automaton

To force multiple words, **keep track** of generated words:

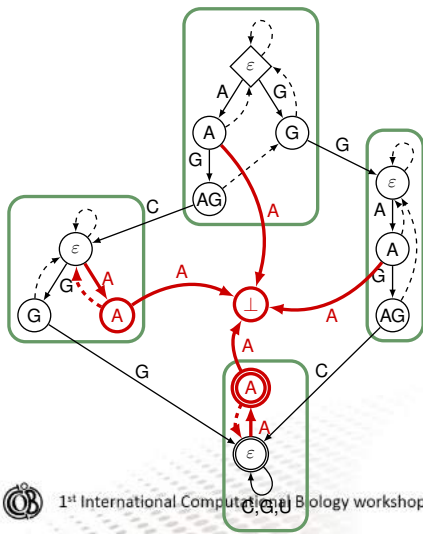
- Create disjunctive automata for each $M' \subseteq M$
- **Reroute** accepting states
- Accepting state = no **forced word** remaining (ϵ in \mathcal{A}_\emptyset)
- Forbidden words can be added to sub-automata

#States:

$$O\left(2^{|\mathcal{M}|} \cdot (\sum_i |f_i| + \sum_j |m_j|)\right)$$

Example:

$$\mathcal{M} = \{AGC, GG\}; \mathcal{F} = \{AA\}$$



Building the grammar

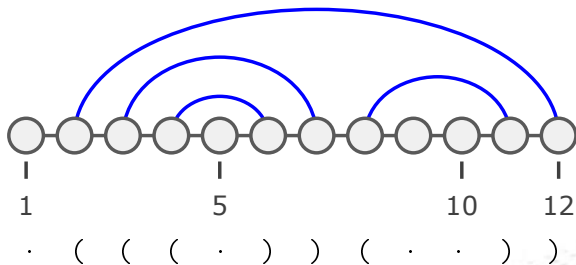
Input: Secondary Structure S + Positional constraints

A Create Parse Tree for secondary structure

B Translate Parse Tree into single-word grammar

C Expand grammar to instantiate compatible base/base-pairs

D Restrict to bases/base-pairs allowed at each position



Building the grammar

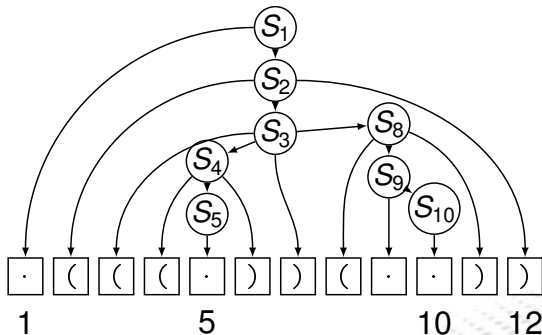
Input: Secondary Structure S + Positional constraints

A Create Parse Tree for secondary structure

B Translate Parse Tree into single-word grammar

C Expand grammar to instantiate compatible base/base-pairs

D Restrict to bases/base-pairs allowed at each position



Building the grammar

Input: Secondary Structure S + Positional constraints

A Create Parse Tree for secondary structure

B Translate Parse Tree into single-word grammar

C Expand grammar to instantiate compatible base/base-pairs

D Restrict to bases/base-pairs allowed at each position

$$\begin{array}{cccc} S_1 \rightarrow .S_2 & S_2 \rightarrow (S_3) & S_3 \rightarrow (S_4)S_8 & S_4 \rightarrow (S_5) \\ S_5 \rightarrow . & S_8 \rightarrow (S_9) & S_9 \rightarrow .S_{10} & S_{10} \rightarrow . \end{array}$$



Building the grammar

Input: Secondary Structure S + Positional constraints

A Create Parse Tree for secondary structure

B Translate Parse Tree into single-word grammar

C Expand grammar to instantiate compatible base/base-pairs

D Restrict to bases/base-pairs allowed at each position

$$V_1 \rightarrow AV_2 \mid CV_2 \mid GV_2 \mid UV_2$$

$$V_2 \rightarrow AV_3U \mid CV_3G \mid GV_3C \mid GV_3U \mid UV_3A \mid UV_3G$$

$$V_3 \rightarrow AV_4UV_8 \mid CV_4GV_8 \mid GV_4CV_8 \mid GV_4UV_8 \mid UV_4AV_8 \mid UV_4GV_8$$

$$V_4 \rightarrow AV_5U \mid CV_5G \mid GV_5C \mid GV_5U \mid UV_5A \mid UV_5G$$

$$V_5 \rightarrow A \mid C \mid G \mid U$$

$$V_8 \rightarrow AV_9U \mid CV_9G \mid GV_9C \mid GV_9U \mid UV_9A \mid UV_9G$$

$$V_9 \rightarrow AV_{10} \mid CV_{10} \mid GV_{10} \mid UV_{10}$$

$$V_{10} \rightarrow A \mid C \mid G \mid U$$



Building the grammar

Input: Secondary Structure S + **Positional constraints**

A Create Parse Tree for secondary structure

B Translate Parse Tree into single-word grammar

C Expand grammar to instantiate compatible base/base-pairs

D Restrict to bases/base-pairs allowed at each position

$$V_1 \rightarrow AV_2 \mid CV_2 \mid GV_2 \mid UV_2$$

$$V_2 \rightarrow AV_3U \mid \underline{CV_3G} \mid \underline{GV_3C} \mid \underline{GV_3U} \mid UV_3A \mid \underline{UV_3G}$$

$$V_3 \rightarrow AV_4UV_8 \mid \underline{CV_4GV_8} \mid \underline{GV_4CV_8} \mid \underline{GV_4UV_8} \mid UV_4AV_8 \mid \underline{UV_4GV_8}$$

$$V_4 \rightarrow AV_5U \mid \underline{CV_5G} \mid \underline{GV_5C} \mid \underline{GV_5U} \mid UV_5A \mid \underline{UV_5G}$$

$$V_5 \rightarrow A \mid C \mid G \mid U$$

$$V_8 \rightarrow AV_9U \mid \underline{CV_9G} \mid \underline{GV_9C} \mid \underline{GV_9U} \mid UV_9A \mid \underline{UV_9G}$$

$$V_9 \rightarrow AV_{10} \mid CV_{10} \mid GV_{10} \mid UV_{10}$$

$$V_{10} \rightarrow A \mid C \mid G \mid U$$



Random generation

Combine CFG and aut. \rightarrow CFG (Multiplying #Rules by $|Q|^3$)

GenGenS [Ponty Termier Denise, Bioinformatics 2006]:

- Precomputes #words for each non-terminal
- Random Generation w.r.t. **weighted distribution**

Energy models:

- **Uniform distribution**
- **Nussinov energy model**
- **Stacking-pairs model (Turner 2004)**

Based on refined, yet similar, grammar

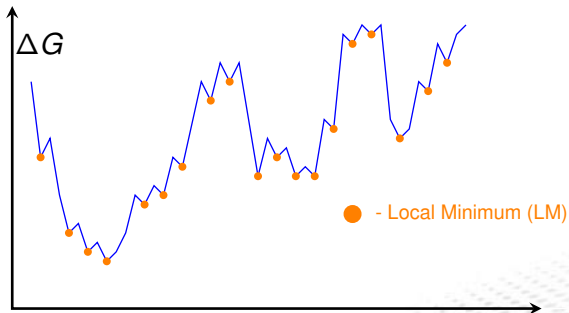
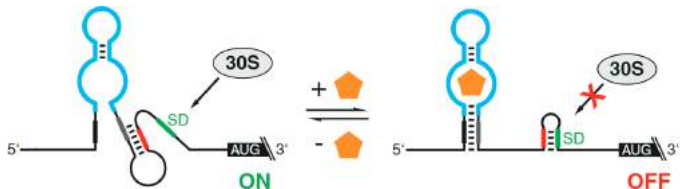
Overall complexity: $|S| \cdot 2^{3|\mathcal{M}|} \cdot (\sum_i |f_i| + \sum_j |m_j|)^3$

- **Linear** on $|S|$
- **Exponential** on $|\mathcal{M}|$, but **NP-Hard** problem



III. Positive design for multiple structures

Motivation: Kinetics and riboswitches



Design objectives

Positive structural design

Optimize affinity of designed sequences towards target structure
Or simply ensure their compatibility with one or **several structures**

Examples: Most stable sequence for given fold. . .

Negative structural design

Limit affinity of designed sequences towards **alternative structures**

Examples: Lowest free-energy, High Boltzmann probability/Low entropy. . .

Additional constraints:

- **Forbid** motif list to appear **anywhere** in design
- **Force** motif list to appear **each at least once**
- **Limit** available alternatives at certain positions
- **Control** overall composition (GC-content)



Design objectives

Positive structural design

Optimize affinity of designed sequences towards target structure
Or simply ensure their compatibility with one or **several structures**

Examples: Most stable sequence for given fold. . .

Negative structural design

Limit affinity of designed sequences towards **alternative structures**

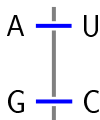
Examples: Lowest free-energy, High Boltzmann probability/Low entropy. . .

Additional constraints:

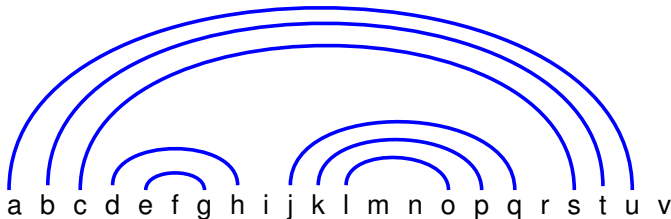
- **Forbid** motif list to appear **anywhere** in design
- **Force** motif list to appear **each at least once**
- **Limit** available alternatives at certain positions
- **Control** overall composition (GC-content)



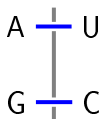
Counting compatible RNAs: Watson-Crick + Single structure



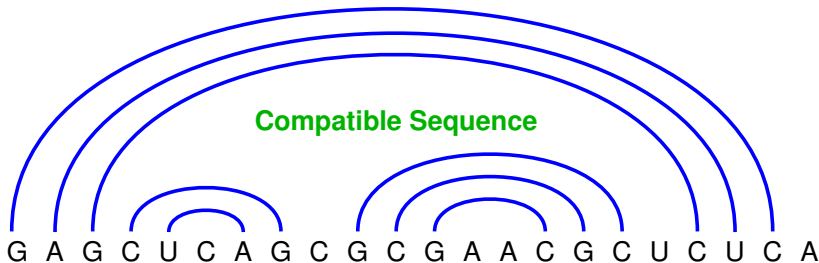
Compatible Base Pairs = Only **Watson-Crick** base pairs



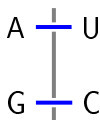
Counting compatible RNAs: Watson-Crick + Single structure



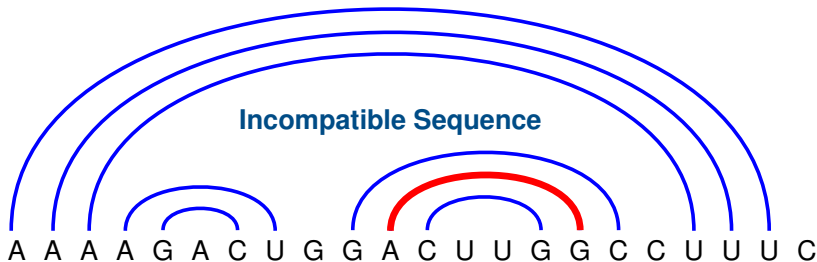
Compatible Base Pairs = Only **Watson-Crick** base pairs



Counting compatible RNAs: Watson-Crick + Single structure



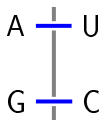
Compatible Base Pairs = Only **Watson-Crick** base pairs



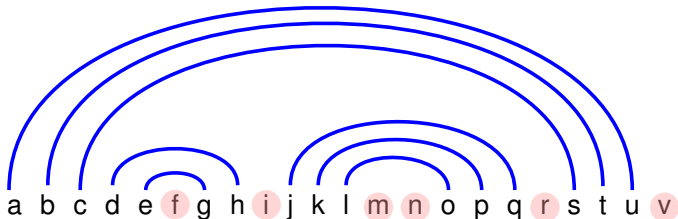
Question: How many **Compatible** sequences?



Counting compatible RNAs: Watson-Crick + Single structure

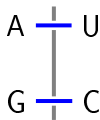


Compatible Base Pairs = Only **Watson-Crick** base pairs

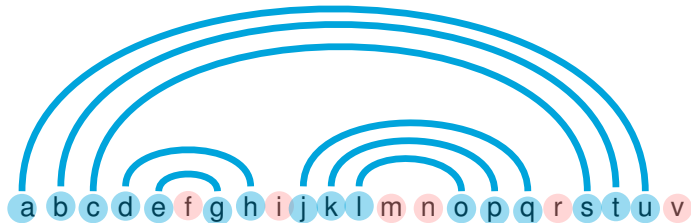


Question: How many **Compatible** sequences?

Counting compatible RNAs: Watson-Crick + Single structure



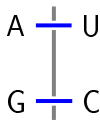
Compatible Base Pairs = Only **Watson-Crick** base pairs



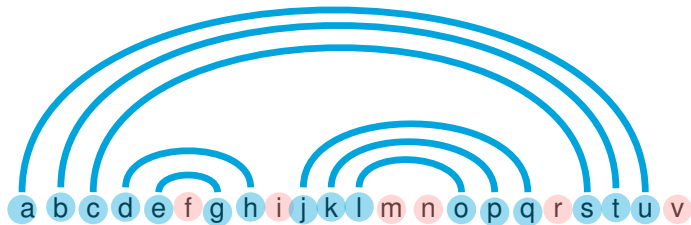
Question: How many **Compatible** sequences?



Counting compatible RNAs: Watson-Crick + Single structure



Compatible Base Pairs = Only **Watson-Crick** base pairs

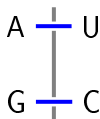


Question: How many **Compatible** sequences?

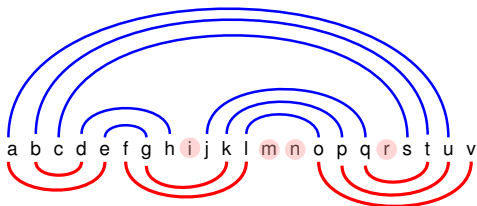
Answer: $4^{\#BPs} \times 4^{\#Unpaired} \rightarrow 268\,435\,456$



Counting compatible RNAs: Watson-Crick + Two structures



Compatible Base Pairs = Only **Watson-Crick** base pairs



Dependency graph:

Cycles + Paths

i m n r

g-e-a-u h j-q

k-p d-b-t

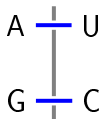
f-l-o-v c-s

Question: How many **Compatible** sequences?

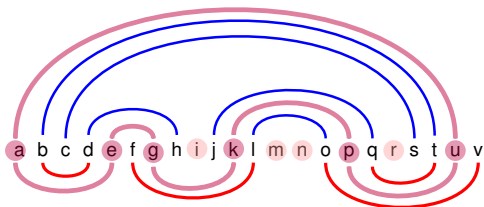
Answer: $\neq \emptyset!$ (both base-pairs and dependency graphs **bipartite**)



Counting compatible RNAs: Watson-Crick + Two structures



Compatible Base Pairs = Only **Watson-Crick** base pairs



Dependency graph:

Cycles + Paths

i m n r

g — e — a — u h j — q

k — p d — b — t

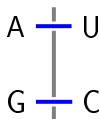
f — l — o — v c — s

Question: How many **Compatible** sequences?

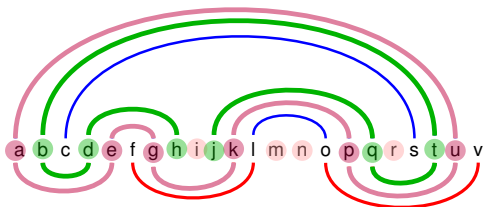
Answer: $\neq \emptyset!$ (both base-pairs and dependency graphs **bipartite**)



Counting compatible RNAs: Watson-Crick + Two structures



Compatible Base Pairs = Only **Watson-Crick** base pairs



Dependency graph:

Cycles + Paths

i m n r

g - e - a - u h j - q

k - p d - b - t

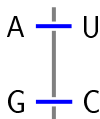
f - l - o - v c - s

Question: How many **Compatible** sequences?

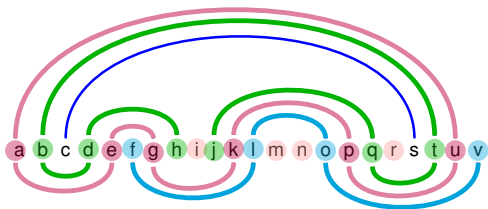
Answer: $\neq \emptyset!$ (both base-pairs and dependency graphs **bipartite**)



Counting compatible RNAs: Watson-Crick + Two structures



Compatible Base Pairs = Only **Watson-Crick** base pairs



Dependency graph:

Cycles + Paths

i m n r

g — e — a — u h j — q

k — p d — b — t

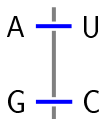
f — l — o — v c — s

Question: How many **Compatible** sequences?

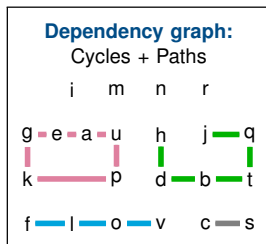
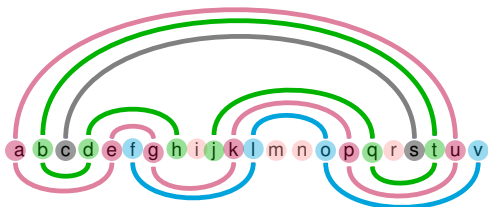
Answer: $\neq \emptyset!$ (both base-pairs and dependency graphs **bipartite**)



Counting compatible RNAs: Watson-Crick + Two structures



Compatible Base Pairs = Only **Watson-Crick** base pairs



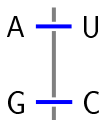
Question: How many **Compatible** sequences?

Answer: $\neq \emptyset!$ (both base-pairs and dependency graphs **bipartite**)

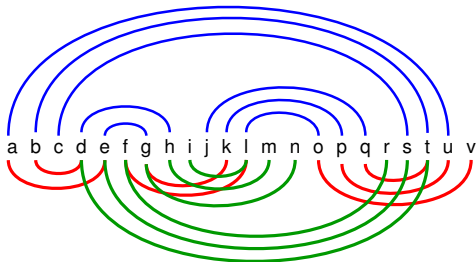
$$4^{\#\text{CCs}} \rightarrow 65\,536$$



Counting compatible RNAs: Watson-Crick + > 2 structs

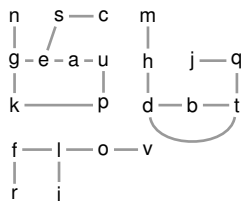


Compatible Base Pairs = Only **Watson-Crick** base pairs



Dependency graph:

Cycles, Paths, Trees...

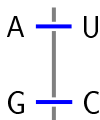


Question: How many **Compatible** sequences?

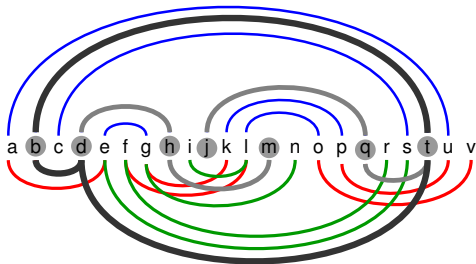
Answer: Non-bipartite $\rightarrow \emptyset$; Bipartite $\rightarrow 4^{\#CCs} = 64$



Counting compatible RNAs: Watson-Crick + > 2 structs

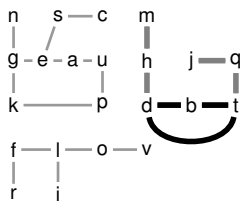


Compatible Base Pairs = Only **Watson-Crick** base pairs



Dependency graph:

Cycles, Paths, Trees...

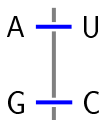


Question: How many **Compatible** sequences?

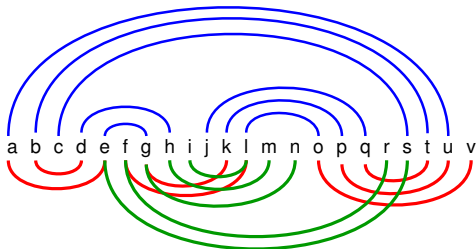
Answer: Non-bipartite $\rightarrow \emptyset$; Bipartite $\rightarrow 4^{\#CCs} = 64$



Counting compatible RNAs: Watson-Crick + > 2 structs

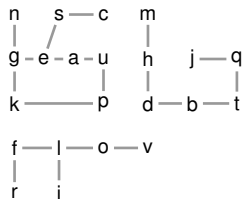


Compatible Base Pairs = Only **Watson-Crick** base pairs



Dependency graph:

Cycles, Paths, Trees...

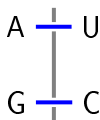


Question: How many **Compatible** sequences?

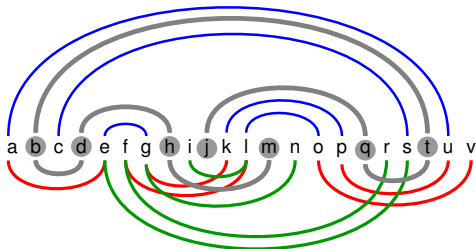
Answer: Non-bipartite $\rightarrow \emptyset$; Bipartite $\rightarrow 4^{\#OCs} = 64$



Counting compatible RNAs: Watson-Crick + > 2 structs

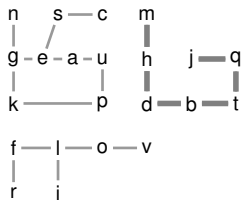


Compatible Base Pairs = Only **Watson-Crick** base pairs



Dependency graph:

Cycles, Paths, Trees...

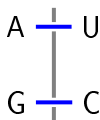


Question: How many **Compatible** sequences?

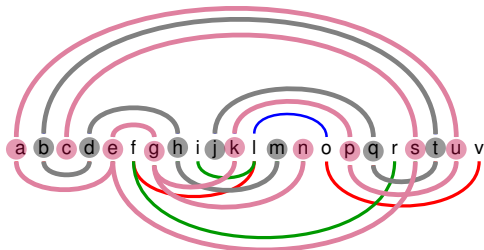
Answer: Non-bipartite $\rightarrow \emptyset$; Bipartite $\rightarrow 4^{\#OCs} = 64$



Counting compatible RNAs: Watson-Crick + > 2 structs

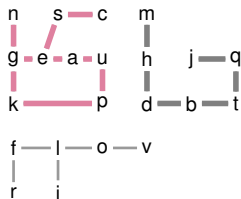


Compatible Base Pairs = Only **Watson-Crick** base pairs



Dependency graph:

Cycles, Paths, Trees...

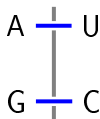


Question: How many **Compatible** sequences?

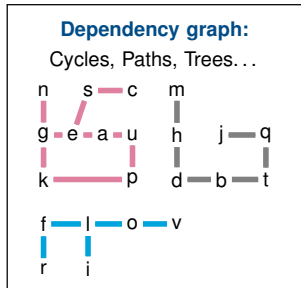
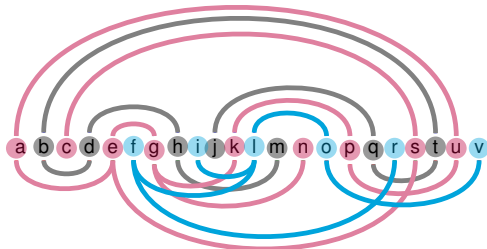
Answer: Non-bipartite $\rightarrow \emptyset$; Bipartite $\rightarrow 4^{\#OCs} = 64$



Counting compatible RNAs: Watson-Crick + > 2 structs



Compatible Base Pairs = Only **Watson-Crick** base pairs

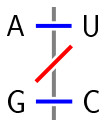


Question: How many **Compatible** sequences?

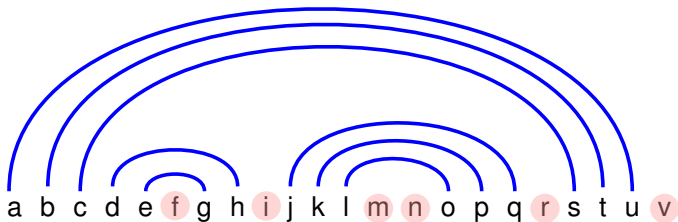
Answer: Non-bipartite $\rightarrow \emptyset$; Bipartite $\rightarrow 4^{\#\text{CCs}} = 64$



Counting compatible RNAs: WC/Wobble + Single struct.



Compatible Base Pairs = Include **Wobble** base pairs

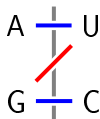


Question: How many **Compatible** sequences?

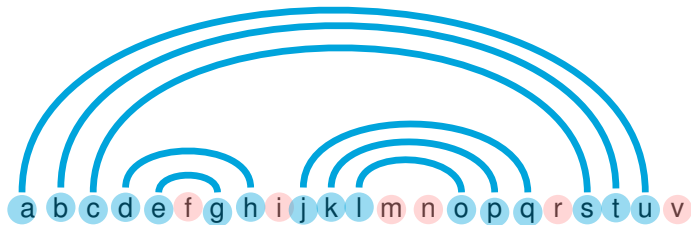
Answer: $4^{\#Unpaired} \times 6^{\#BPs} \rightarrow 6879707136$



Counting compatible RNAs: WC/Wobble + Single struct.



Compatible Base Pairs = Include **Wobble** base pairs

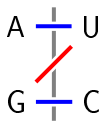


Question: How many **Compatible** sequences?

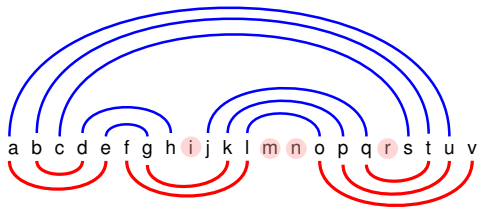
Answer: $4^{\#Unpaired} \times 6^{\#BPs} \rightarrow 6\ 879\ 707\ 136$



Counting compatible RNAs: WC/Wobble + Two structures



Compatible Base Pairs = Include **Wobble** base pairs



Dependency graph:

Cycles + Paths

i m n r

g-e-a-u h j-q

k-p d-b-t

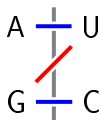
f-l-o-v c-s

Question: How many **Compatible** sequences?

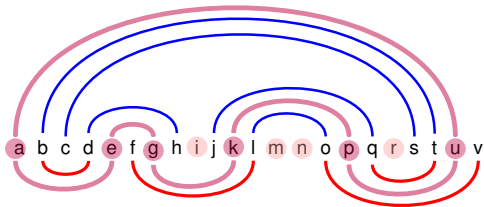
Answer: $\neq \emptyset!$ (base-pairs and dependency graphs **always bipartite**)



Counting compatible RNAs: WC/Wobble + Two structures



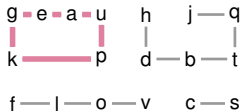
Compatible Base Pairs = Include **Wobble** base pairs



Dependency graph:

Cycles + Paths

i m n r

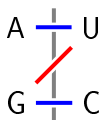


Question: How many **Compatible** sequences?

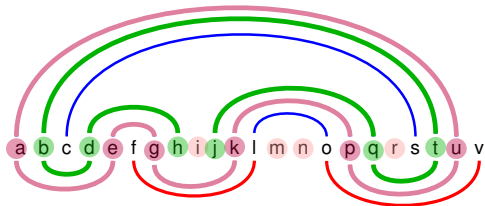
Answer: $\neq \emptyset!$ (base-pairs and dependency graphs **always bipartite**)



Counting compatible RNAs: WC/Wobble + Two structures



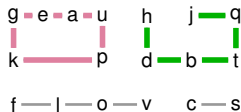
Compatible Base Pairs = Include **Wobble** base pairs



Dependency graph:

Cycles + Paths

i m n r

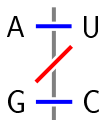


Question: How many **Compatible** sequences?

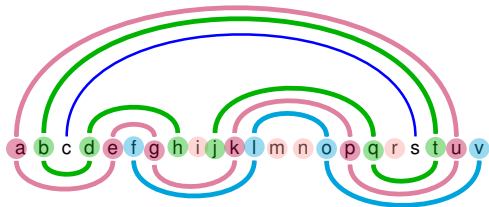
Answer: $\neq \emptyset!$ (base-pairs and dependency graphs **always bipartite**)



Counting compatible RNAs: WC/Wobble + Two structures



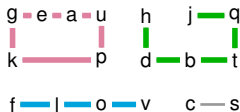
Compatible Base Pairs = Include **Wobble** base pairs



Dependency graph:

Cycles + Paths

i m n r

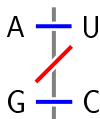


Question: How many **Compatible** sequences?

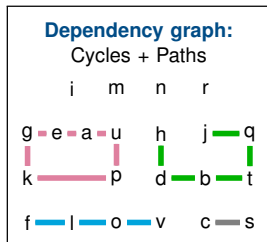
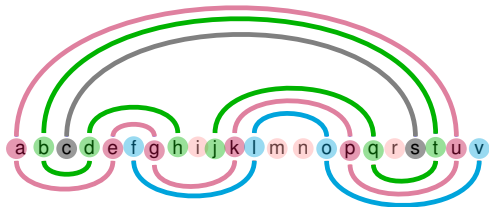
Answer: $\neq \emptyset!$ (base-pairs and dependency graphs **always bipartite**)



Counting compatible RNAs: WC/Wobble + Two structures



Compatible Base Pairs = Include **Wobble** base pairs



Question: How many **Compatible** sequences?

Answer: $\neq \emptyset!$ (base-pairs and dependency graphs **always bipartite**)

$$\#Designs(G) = \prod_{c \in CC(G)} \#Designs(cc)$$



Counting compatible designs for paths and cycles

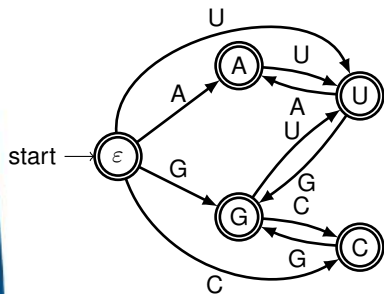
Theorem (#Compatible designs for paths and cycles)

The numbers of compatible designs for **paths** and **cycles** of length n are:

$$p(n) = 2 \mathcal{F}_{n+2} \quad \text{and} \quad c(n) = 2 \mathcal{F}_n + 4 \mathcal{F}_{n-1}$$

where \mathcal{F}_n : n^{th} **Fibonacci number**, $\mathcal{F}_0 = 0$, $\mathcal{F}_1 = 1$ and $\mathcal{F}_n = \mathcal{F}_{n-1} + \mathcal{F}_{n-2}$.

For paths: A simple DFA generates compatible sequences



Remark: $A \leftrightarrow C/G \leftrightarrow U$ symmetry



Counting compatible designs for paths and cycles

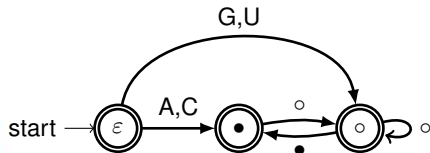
Theorem (#Compatible designs for paths and cycles)

The numbers of compatible designs for **paths** and **cycles** of length n are:

$$p(n) = 2 \mathcal{F}_{n+2} \quad \text{and} \quad c(n) = 2 \mathcal{F}_n + 4 \mathcal{F}_{n-1}$$

where \mathcal{F}_n : n^{th} **Fibonacci number**, $\mathcal{F}_0 = 0$, $\mathcal{F}_1 = 1$ and $\mathcal{F}_n = \mathcal{F}_{n-1} + \mathcal{F}_{n-2}$.

For paths: A simple DFA generates compatible sequences



Remark: $A \leftrightarrow C/G \leftrightarrow U$ symmetry

Counting compatible designs for paths and cycles

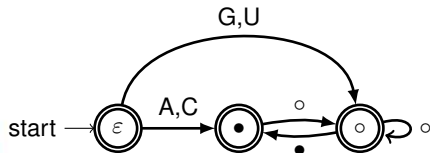
Theorem (#Compatible designs for paths and cycles)

The numbers of compatible designs for **paths** and **cycles** of length n are:

$$p(n) = 2 \mathcal{F}_{n+2} \quad \text{and} \quad c(n) = 2 \mathcal{F}_n + 4 \mathcal{F}_{n-1}$$

where \mathcal{F}_n : n^{th} **Fibonacci number**, $\mathcal{F}_0 = 0$, $\mathcal{F}_1 = 1$ and $\mathcal{F}_n = \mathcal{F}_{n-1} + \mathcal{F}_{n-2}$.

For paths: A simple DFA generates compatible sequences



Remark: $A \leftrightarrow C/G \leftrightarrow U$ symmetry

$$m_{\bullet}(n) = m_{\circ}(n-1)$$

Counting compatible designs for paths and cycles

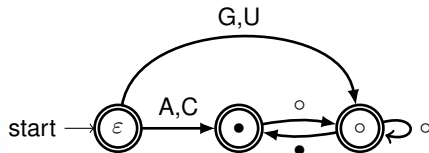
Theorem (#Compatible designs for paths and cycles)

The numbers of compatible designs for **paths** and **cycles** of length n are:

$$p(n) = 2 \mathcal{F}_{n+2} \quad \text{and} \quad c(n) = 2 \mathcal{F}_n + 4 \mathcal{F}_{n-1}$$

where \mathcal{F}_n : n^{th} **Fibonacci number**, $\mathcal{F}_0 = 0$, $\mathcal{F}_1 = 1$ and $\mathcal{F}_n = \mathcal{F}_{n-1} + \mathcal{F}_{n-2}$.

For paths: A simple DFA generates compatible sequences



Remark: $A \leftrightarrow C/G \leftrightarrow U$ symmetry

$$\begin{aligned} m_{\bullet}(n) &= m_{\circ}(n-1) \\ m_{\circ}(n) &= m_{\circ}(n-1) + m_{\bullet}(n-1) \\ &= m_{\circ}(n-1) + m_{\circ}(n-2) \\ &= \mathcal{F}(n+2) \end{aligned}$$



Counting compatible designs for paths and cycles

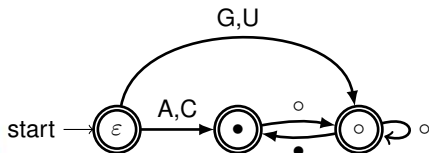
Theorem (#Compatible designs for paths and cycles)

The numbers of compatible designs for *paths* and *cycles* of length n are:

$$p(n) = 2 \mathcal{F}_{n+2} \quad \text{and} \quad c(n) = 2 \mathcal{F}_n + 4 \mathcal{F}_{n-1}$$

where \mathcal{F}_n : n^{th} **Fibonacci number**, $\mathcal{F}_0 = 0$, $\mathcal{F}_1 = 1$ and $\mathcal{F}_n = \mathcal{F}_{n-1} + \mathcal{F}_{n-2}$.

For paths: A simple DFA generates compatible sequences



Remark: $A \leftrightarrow C/G \leftrightarrow U$ symmetry

$$\begin{aligned} m_{\bullet}(n) &= m_{\circ}(n-1) \\ m_{\circ}(n) &= m_{\circ}(n-1) + m_{\bullet}(n-1) \\ &= m_{\circ}(n-1) + m_{\circ}(n-2) \\ &= \mathcal{F}(n+2) \end{aligned}$$

(Since $m_{\circ}(0) = 1$ and $m_{\circ}(1) = 2$)



Counting compatible designs for paths and cycles

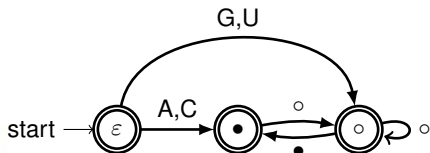
Theorem (#Compatible designs for paths and cycles)

The numbers of compatible designs for *paths* and *cycles* of length n are:

$$p(n) = 2 \mathcal{F}_{n+2} \quad \text{and} \quad c(n) = 2 \mathcal{F}_n + 4 \mathcal{F}_{n-1}$$

where \mathcal{F}_n : n^{th} **Fibonacci number**, $\mathcal{F}_0 = 0$, $\mathcal{F}_1 = 1$ and $\mathcal{F}_n = \mathcal{F}_{n-1} + \mathcal{F}_{n-2}$.

For paths: A simple DFA generates compatible sequences



Remark: A \leftrightarrow C/G \leftrightarrow U symmetry

$$\begin{aligned} m_{\bullet}(n) &= m_{\circ}(n-1) \\ m_{\circ}(n) &= m_{\circ}(n-1) + m_{\bullet}(n-1) \\ &= m_{\circ}(n-1) + m_{\circ}(n-2) \\ &= \mathcal{F}(n+2) \end{aligned}$$

(Since $m_{\circ}(0) = 1$ and $m_{\circ}(1) = 2$)

$$p(n) := m_{\epsilon}(n) = 2 m_{\bullet}(n-1) + 2 m_{\circ}(n-1) = 2(\mathcal{F}(n) + \mathcal{F}(n+1)) = 2\mathcal{F}(n+2)$$



Counting compatible designs for paths and cycles

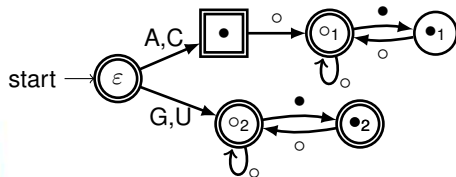
Theorem (#Compatible designs for paths and cycles)

The numbers of compatible designs for **paths** and **cycles** of length n are:

$$p(n) = 2 \mathcal{F}_{n+2} \quad \text{and} \quad c(n) = 2 \mathcal{F}_n + 4 \mathcal{F}_{n-1}$$

where \mathcal{F}_n : n^{th} **Fibonacci number**, $\mathcal{F}_0 = 0$, $\mathcal{F}_1 = 1$ and $\mathcal{F}_n = \mathcal{F}_{n-1} + \mathcal{F}_{n-2}$.

For cycle: A barely more involved DFA generates compatible sequences



Remark: $A \leftrightarrow C/G \leftrightarrow U$ symmetry

$$m_{o_2}(n) = \mathcal{F}(n+2)$$

$$m_{o_1}(n) = \mathcal{F}(n+1)$$

(Since $m_{o_1}(0) = 1$ and $m_{o_1}(1) = 1$)

$$\begin{aligned} c(n) &:= m_{\epsilon}(n) = 2 m_{o_1}(n-2) + 2 m_{o_2}(n-1) \\ &= 2(\mathcal{F}(n-1) + \mathcal{F}(n+1)) = 2 \mathcal{F}(n) + 4 \mathcal{F}(n-1) \end{aligned}$$



Counting compatible designs for paths and cycles

Theorem (#Compatible designs for paths and cycles)

The numbers of compatible designs for **paths** and **cycles** of length n are:

$$p(n) = 2 \mathcal{F}_{n+2} \quad \text{and} \quad c(n) = 2 \mathcal{F}_n + 4 \mathcal{F}_{n-1}$$

where \mathcal{F}_n : n^{th} **Fibonacci number**, $\mathcal{F}_0 = 0$, $\mathcal{F}_1 = 1$ and $\mathcal{F}_n = \mathcal{F}_{n-1} + \mathcal{F}_{n-2}$.

Theorem (#Compatible designs for general 2-structures graphs)

G : dependency graph associated with 2 RNA structures (max deg=2).

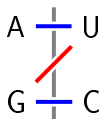
The number #Designs(G) of compatible designs for G is given by

$$\#\text{Designs}(G) = \prod_{p \in \mathcal{P}(G)} 2 \mathcal{F}_{|p|+2} \times \prod_{c \in \mathcal{C}(G)} (2 \mathcal{F}_{|c|} + 4 \mathcal{F}_{|c|-1})$$

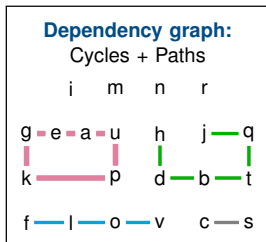
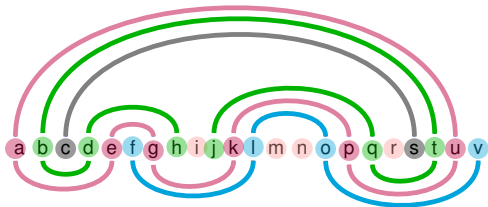
where G decomposes into **paths** $\mathcal{P}(G)$ and **cycles** $\mathcal{C}(G)$.



Counting compatible sequences: WC/Wobble + Two structures



Compatible Base Pairs = Include **Wobble** base pairs



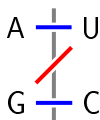
Question: How many **Compatible** sequences?

Answer: $\neq \emptyset!$ (base-pairs and dependency graphs **always bipartite**)

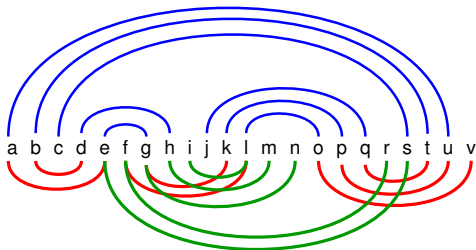
$$\# \text{Designs}(G) = \prod_{c \in CC(G)} \# \text{Designs}(cc) = 2\,322\,432$$



Counting compatible sequences: Watson-Crick + > 2 structures



Compatible Base Pairs = Include **Wobble** base pairs



Dependency graph:

Cycles, Paths, Trees...

n s—c m

g—e—a—u h j—q

| | | |
k—p d—b—t

f—l—o—v

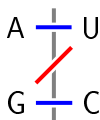
r i

Question: How many **Compatible** sequences?

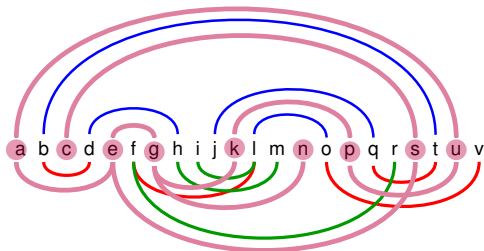
Answer: Non-bipartite $\rightarrow \emptyset$; Bipartite \rightarrow



Counting compatible sequences: Watson-Crick + > 2 structures

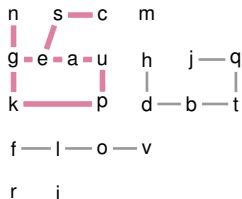


Compatible Base Pairs = Include **Wobble** base pairs



Dependency graph:

Cycles, Paths, Trees...

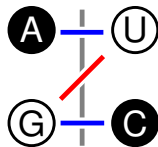
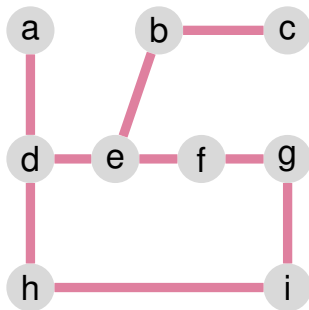


Question: How many **Compatible** sequences?

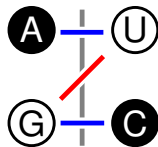
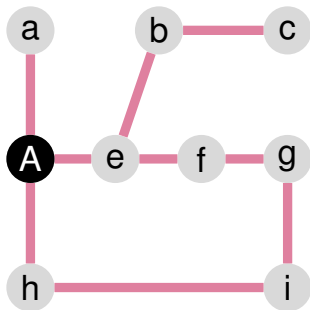
Answer: Non-bipartite $\rightarrow \emptyset$; Bipartite $\rightarrow \prod_{cc \in CC(G)} 2 \times \#IS(cc)$



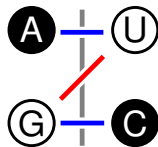
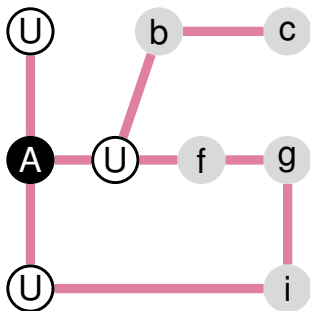
Bijection between Independent Sets and Valid Designs



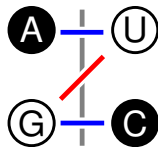
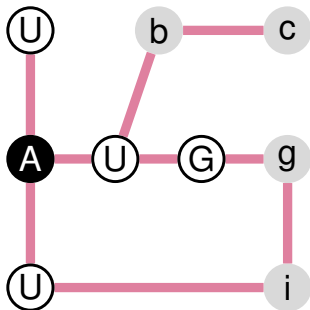
Bijection between Independent Sets and Valid Designs



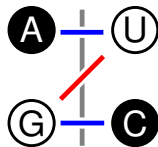
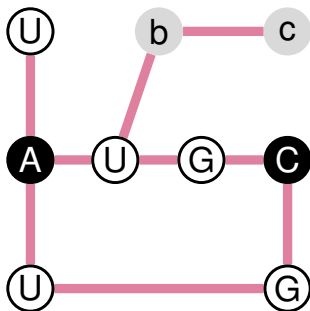
Bijection between Independent Sets and Valid Designs



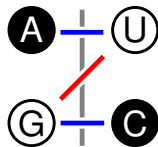
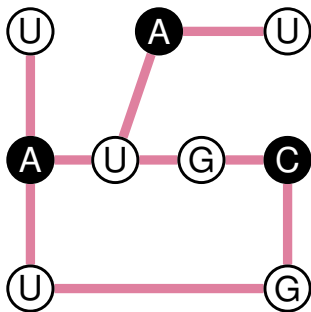
Bijection between Independent Sets and Valid Designs



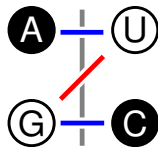
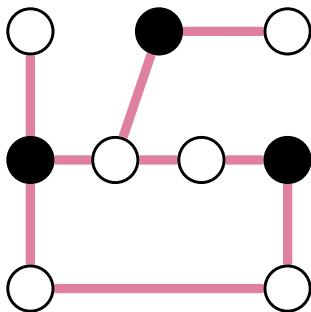
Bijection between Independent Sets and Valid Designs



Bijection between Independent Sets and Valid Designs



Bijection between Independent Sets and Valid Designs

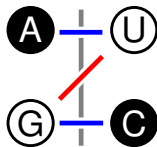
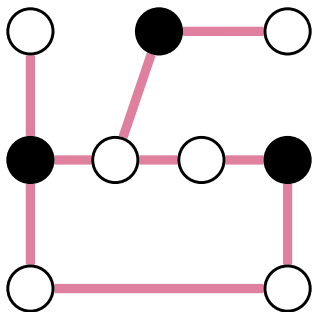


Remark: No adjacent **black letters** in compatible designs

Up to trivial symmetry* (e.g. top-left position $\in \{G, A\}$):

$$\text{Designs}^*(cc) \subseteq \text{IndependentSets}(cc)$$

Bijection between Independent Sets and Valid Designs



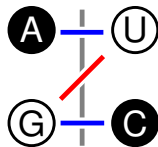
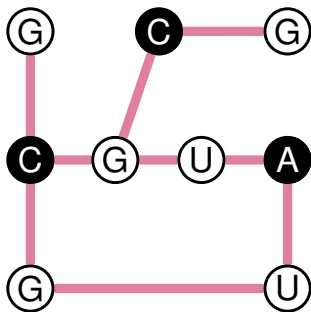
Remark: No adjacent **black letters** in compatible designs

Up to trivial symmetry* (e.g. top-left position $\in \{G, A\}$):

$$\text{Designs}^*(cc) \subseteq \text{IndependentSets}(cc)$$

Also, IS (black) + \nearrow vert. $\in \{G, A\} \Rightarrow$ **Unique** compatible design

Bijection between Independent Sets and Valid Designs



Remark: No adjacent **black letters** in compatible designs

Up to trivial symmetry* (e.g. top-left position $\in \{G, A\}$):

$$\text{Designs}^*(cc) \subseteq \text{IndependentSets}(cc)$$

Also, IS (black) + ↖ vert. $\in \{G, A\} \Rightarrow$ **Unique** compatible design

\Rightarrow Bijection between $\text{Designs}^*(cc)$ and $\text{IndependentSets}(cc)$.



Valid designs and independent sets

Theorem (#Valid design for bipartite connected dependency graphs)

Let G be a **bipartite connected** dependency graph, one has:

$$\#Designs(G) = 2 \times \#Designs^*(G) = 2 \times \#IS(G)$$

For a **bipartite** dependency graph G we get:

$$\#Designs(G) = \prod_{cc \in CC(G)} 2 \times \#IS(cc) = 2^{|CC(G)|} \times \#IS(G)$$

But $\#IS(G)$ is $\#P$ -hard on bipartite graphs [Bubbley&Dyer'01]
(+ Any G is a dependency graph)

Algorithm $\mathcal{A} \in P$ for $\#Designs(G) \rightarrow$ Algorithm $\mathcal{A}' \in P$ for $\#BIS \dots$

Theorem

$\#Designs$ is $\#P$ -hard.

No polynomial algorithm for $\#Designs(G)$ unless $\#P = FP$ ($\Rightarrow P = NP$)



Valid designs and independent sets

Theorem (#Valid design for bipartite connected dependency graphs)

Let G be a **bipartite connected** dependency graph, one has:

$$\#Designs(G) = 2 \times \#Designs^*(G) = 2 \times \#IS(G)$$

For a **bipartite** dependency graph G we get:

$$\#Designs(G) = \prod_{cc \in CC(G)} 2 \times \#IS(cc) = 2^{|CC(G)|} \times \#IS(G)$$

But $\#IS(G)$ is #P-hard on bipartite graphs [Bubbley&Dyer'01]
(+ Any G is a dependency graph)

Algorithm $\mathcal{A} \in P$ for $\#Designs(G) \rightarrow$ Algorithm $\mathcal{A}' \in P$ for $\#BIS \dots$

Theorem

$\#Designs$ is #P-hard.

No polynomial algorithm for $\#Designs(G)$ unless $\#P = FP \Leftrightarrow P = NP$



Valid designs and independent sets

Theorem (#Valid design for bipartite connected dependency graphs)

Let G be a **bipartite connected** dependency graph, one has:

$$\#Designs(G) = 2 \times \#Designs^*(G) = 2 \times \#IS(G)$$

For a **bipartite** dependency graph G we get:

$$\#Designs(G) = \prod_{cc \in CC(G)} 2 \times \#IS(cc) = 2^{|CC(G)|} \times \#IS(G)$$

But $\#IS(G)$ is #P-hard on bipartite graphs [Bubbley&Dyer'01]
(+ Any G is a dependency graph)

Algorithm $\mathcal{A} \in P$ for $\#Designs(G) \rightarrow$ Algorithm $\mathcal{A}' \in P$ for $\#BIS \dots$

Theorem

#Designs is #P-hard.

No polynomial algorithm for $\#Designs(G)$ unless $\#P = FP \Leftrightarrow P = NP$



Valid designs and independent sets

Theorem (#Valid design for bipartite connected dependency graphs)

Let G be a **bipartite connected** dependency graph, one has:

$$\#Designs(G) = 2 \times \#Designs^*(G) = 2 \times \#IS(G)$$

For a **bipartite** dependency graph G we get:

$$\#Designs(G) = \prod_{cc \in CC(G)} 2 \times \#IS(cc) = 2^{|CC(G)|} \times \#IS(G)$$

But $\#IS(G)$ is #P-hard on bipartite graphs [Bubbly&Dyer'01]
(+ Any G is a dependency graph)

Algorithm $\mathcal{A} \in P$ for $\#Designs(G) \rightarrow$ Algorithm $\mathcal{A}' \in P$ for $\#BIS \dots$

Theorem

#Designs is #P-hard.

No polynomial algorithm for $\#Designs(G)$ unless $\#P = FP (\Rightarrow P = NP)$



Consequences

Corollary (#Approximability for ≤ 5 structures) [Weitz'06]

For any G built from ≤ 5 pseudoknotted structures, #Design(G) can be approximated within **any ratio** in **polynomial time** (PTAS)

Corollary (#BIS hardness for > 5 struct.) [Cai, Galanis, Goldberg, Jerrum, McQuillan'16]

Beyond 5 **pseudoknotted** structures, approximating #Design becomes **as hard as** approximating #BIS without any constraint.

Why pseudoknotted? Because any bipartite graph of max degree Δ can be **decomposed** into Δ matchings **in polynomial time** (Vizing's theorem).

Lastly, connection between **counting** and **sampling** [Jerrum, Valiant, Vazirani'86].

Conjecture (#BIS hardness of sampling)

Generating comp. sequences **(almost) uniformly** for general input is **#BIS-hard**.



Consequences

Corollary (#Approximability for ≤ 5 structures) [Weitz'06]

For any G built from ≤ 5 pseudoknotted structures, #Design(G) can be approximated within **any ratio** in **polynomial time** (PTAS)

Corollary (#BIS hardness for > 5 struct.) [Cai, Galanis, Goldberg, Jerrum, McQuillan'16]

Beyond 5 **pseudoknotted** structures, approximating #Design becomes **as hard as** approximating #BIS without any constraint.

Why pseudoknotted? Because any bipartite graph of max degree Δ can be **decomposed** into Δ matchings **in polynomial time** (Vizing's theorem).

Lastly, connection between **counting** and **sampling** [Jerrum, Valiant, Vazirani'86].

Conjecture (#BIS hardness of sampling)

Generating comp. sequences **(almost) uniformly** for general input is **#BIS-hard**.



Consequences

Corollary (#Approximability for ≤ 5 structures) [Weitz'06]

For any G built from ≤ 5 pseudoknotted structures, #Design(G) can be approximated within **any ratio** in **polynomial time** (PTAS)

Corollary (#BIS hardness for > 5 struct.) [Cai, Galanis, Goldberg, Jerrum, McQuillan'16]

Beyond 5 **pseudoknotted** structures, approximating #Design becomes **as hard as** approximating #BIS without any constraint.

Why pseudoknotted? Because any bipartite graph of max degree Δ can be **decomposed** into Δ matchings **in polynomial time** (Vizing's theorem).

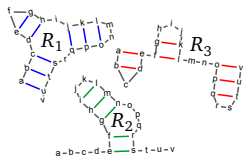
Lastly, connection between **counting** and **sampling** [Jerrum, Valiant, Vazirani'86].

Conjecture (#BIS hardness of sampling)

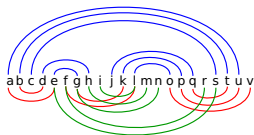
Generating comp. sequences **(almost) uniformly** for general input is **#BIS-hard**.



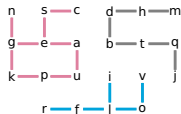
Perspectives: FPT and Boltzmann sampling algorithms



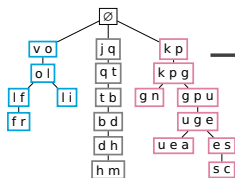
i) Input Structures



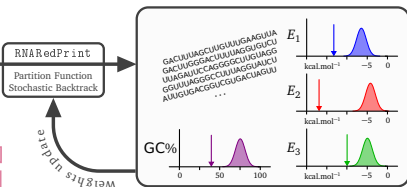
ii) Merged Base-Pairs



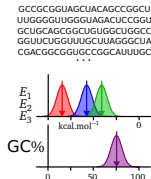
iii) Compatibility Graph



iv) Tree Decomposition



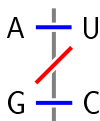
v) Weight Optimization (Adaptive Sampling)



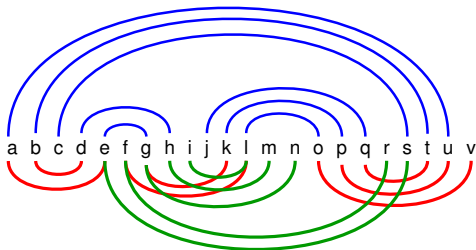
vi) Final Designs

- **FPT algorithm** for counting based on **tree decomposition**
- Multidimensional Boltzmann sampling to control energies, GC...

Counting compatible sequences: Watson-Crick + > 2 structures



Compatible Base Pairs = Include **Wobble** base pairs



Dependency graph:

Cycles, Paths, Trees...

n s—c m

g—e—u h j—q

k—p d—b—t

f—l—o—v

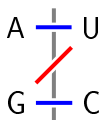
r i

Question: How many **Compatible** sequences?

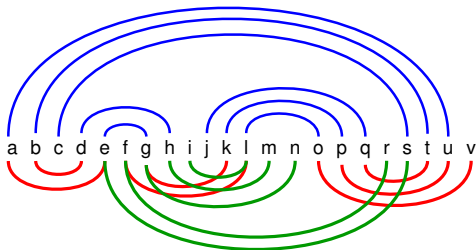
Answer: Bipartite →



Counting compatible sequences: Watson-Crick + > 2 structures



Compatible Base Pairs = Include **Wobble** base pairs



Dependency graph:

Cycles, Paths, Trees...

n s—c m

g—e—u h j—q

k—p d—b—t

f—l—o—v

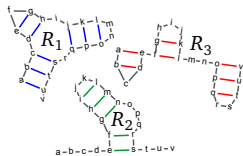
r i

Question: How many **Compatible** sequences?

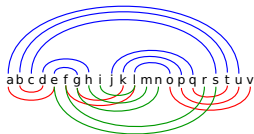
Answer: Bipartite $\rightarrow \prod_{cc \in CC(G)} 2 \times \#IS(cc) = 496\,672$



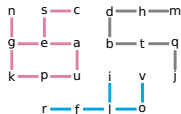
Perspectives: FPT and Boltzmann sampling algorithms



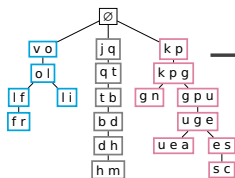
i) Input Structures



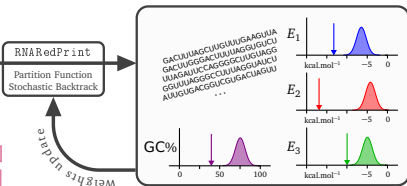
ii) Merged Base-Pairs



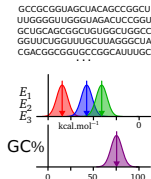
iii) Compatibility Graph



iv) Tree Decomposition



v) Weight Optimization (Adaptive Sampling)



vi) Final Designs

- **FPT algorithm** for counting based on **tree decomposition**
- **Multidimensional Boltzmann sampling** to control energies, GC...

Conclusions

- **RNA is cool!**
- **RNA design** is one of the current challenge of RNA bioinformatics with far-reaching consequences for drug design, synthetic biology. . .
- Practical use-cases require **expressive and modular constraints**
- Future methods: **kinetics, interactions, multiple structures, pseudoknots. . .**
- **RNA inverse folding** is the combinatorial core of design. It remains **largely unsolved**, and opens **new lines of research** in Comp. Sci.

Collaborators

University McGill

Vladimir Reinharz
Jérôme Waldispühl



MIT

Bonnie Berger
Srinivas Devadas
Alex Levin
Mieszko Lis
Charles O'Donnell



LRI – Univ. Paris Sud

Alain Denise
Vincent Le Gallic



Wuhan University

Yi Zhang
Yu Zhou



LIGM – Marne la Vallée

Stéphane Vialette



LIX – Ecole Polytechnique

Alice Héliou
Mireille Regnier



Simon Fraser University

Jozef Hales
Jan Manuch (UBC)
Ladislav Stacho



Cédric Chauve
Julien Courtiel

TBI Vienna

Ronnie Lorenz
Andrea Tanzer





Thanks!



Poster submission & Registration open soon...
(+ ISCB travel fellowships for students)

References I



R. Nussinov and A.B. Jacobson.

Fast algorithm for predicting the secondary structure of single-stranded RNA.
Proc Natl Acad Sci U S A, 77:6903–13, 1980.

