Enumerating RNA Pseudoknots

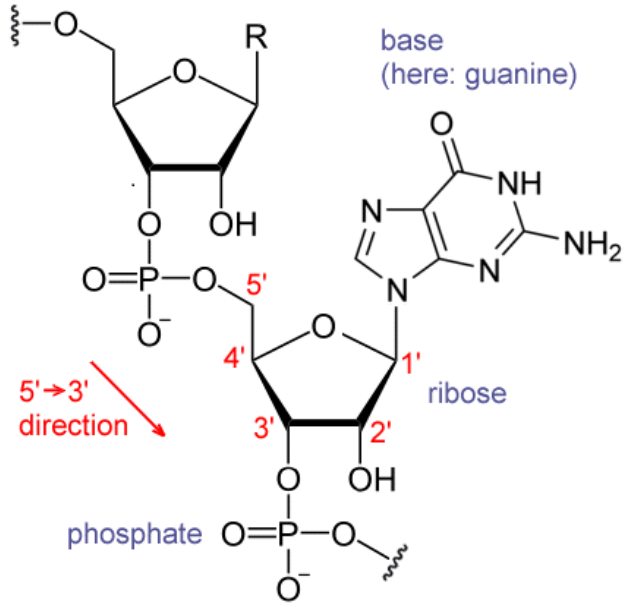
Markus E. Nebel (joint work with Frank Weinberg)

University of Kaiserslautern

June 16th 2011

Nucleotides and RNA structure

Ribonucleic acid (RNA) is a polymer built from so-called nucleotides (building blocks):



Four different types only differing in the base involved (adenine, cytosine, guanine and uracil, $\{a,c,g,u\}$).

Nucleotides and RNA structure

Polymer: By linking the phosphate (attached to the $5'$ carbon) of one nucleotide to the $3'$ carbon of another nucleotide.

$\rightsquigarrow$ *primary structure* modeled as a string over $\{a,c,g,u\}$ (in $5' \to 3'$ direction).

Nucleotides and RNA structure

Polymer: By linking the phosphate (attached to the $5'$ carbon) of one nucleotide to the $3'$ carbon of another nucleotide.

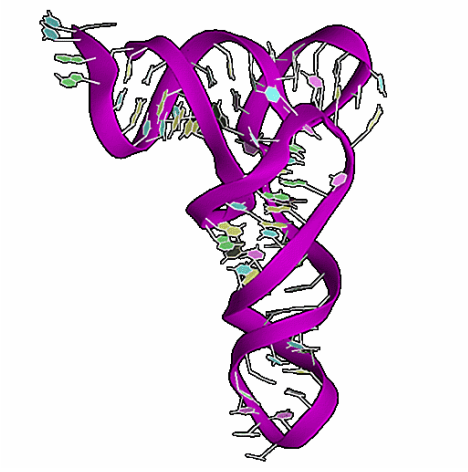
$\rightsquigarrow$ *primary structure* modeled as a string over $\{a,c,g,u\}$ (in $5' \to 3'$ direction).

Spatial structure: Bases are sort of *sticky*; c with g, a with u and g with u can form hydrogen bonds such that the linear molecule is twisted into a 3D conformation (*tertiary structure*) of minimal free energy which determines the molecule's function.

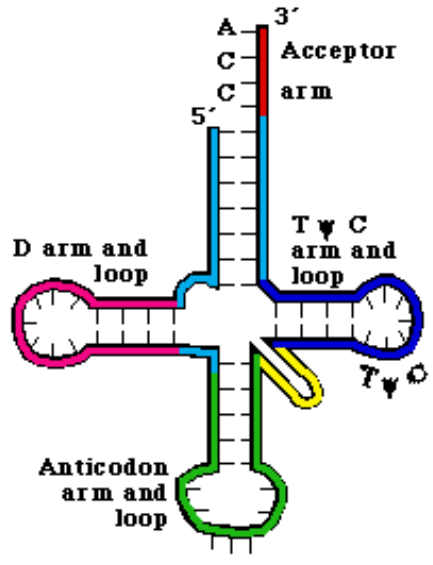
Discrete Models of RNA polymers

Secondary structure: Base pairs *leaving the plane* are prohibited.

Example:



untwisting+planarity
→



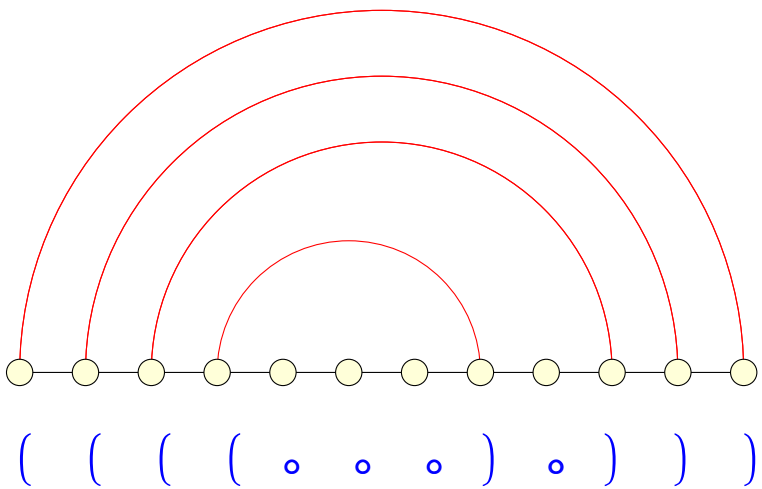
((((((((ooo((((oooooo))))oo((((((ooooooo))))))ooooooooo((((oooooo))))))))))ooo

Dot-bracket representation; obvious correspondence to tree structures.

Discrete Models of RNA polymers

Alternative representation: *Contact graph*

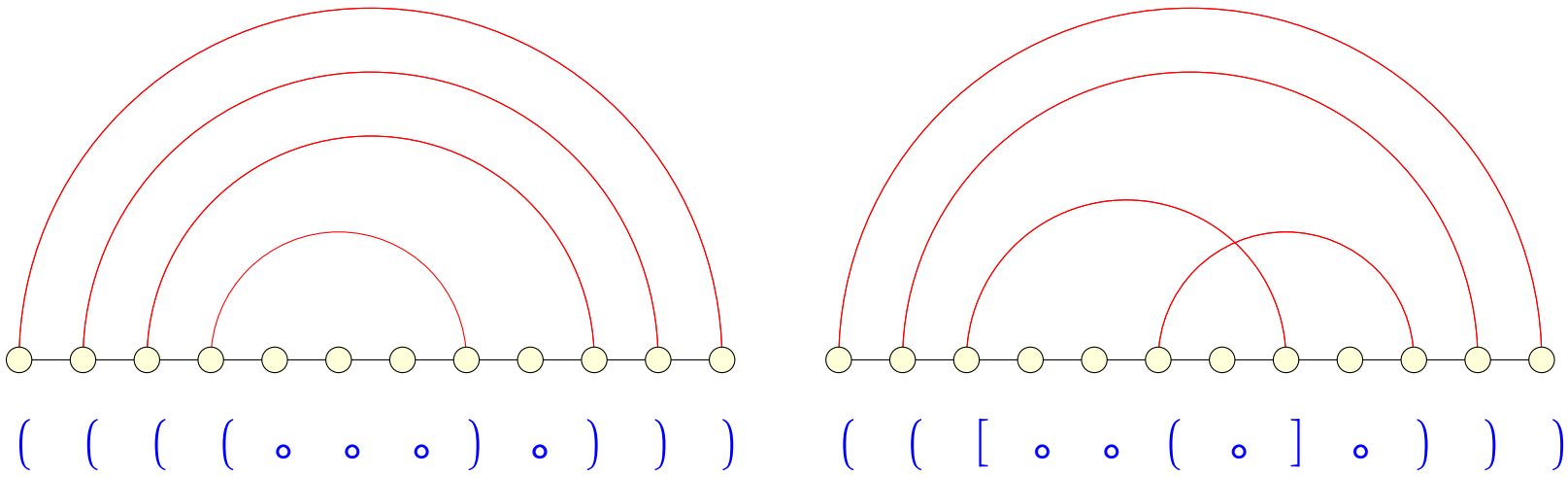
- Backbone (primary structure) linear chain of vertices (in natural order) each representing a nucleotide;
- hydrogen bonds are represented by arcs in the upper half-plane.



Discrete Models of RNA polymers

Alternative representation: *Contact graph*

- Backbone (primary structure) linear chain of vertices (in natural order) each representing a nucleotide;
- hydrogen bonds are represented by arcs in the upper half-plane.



If arcs do cross (right example) $\rightsquigarrow$ **pseudoknot** (otherwise secondary structure).

Algorithmic challenges

Determine the structure from known sequence of bases (primary structure)

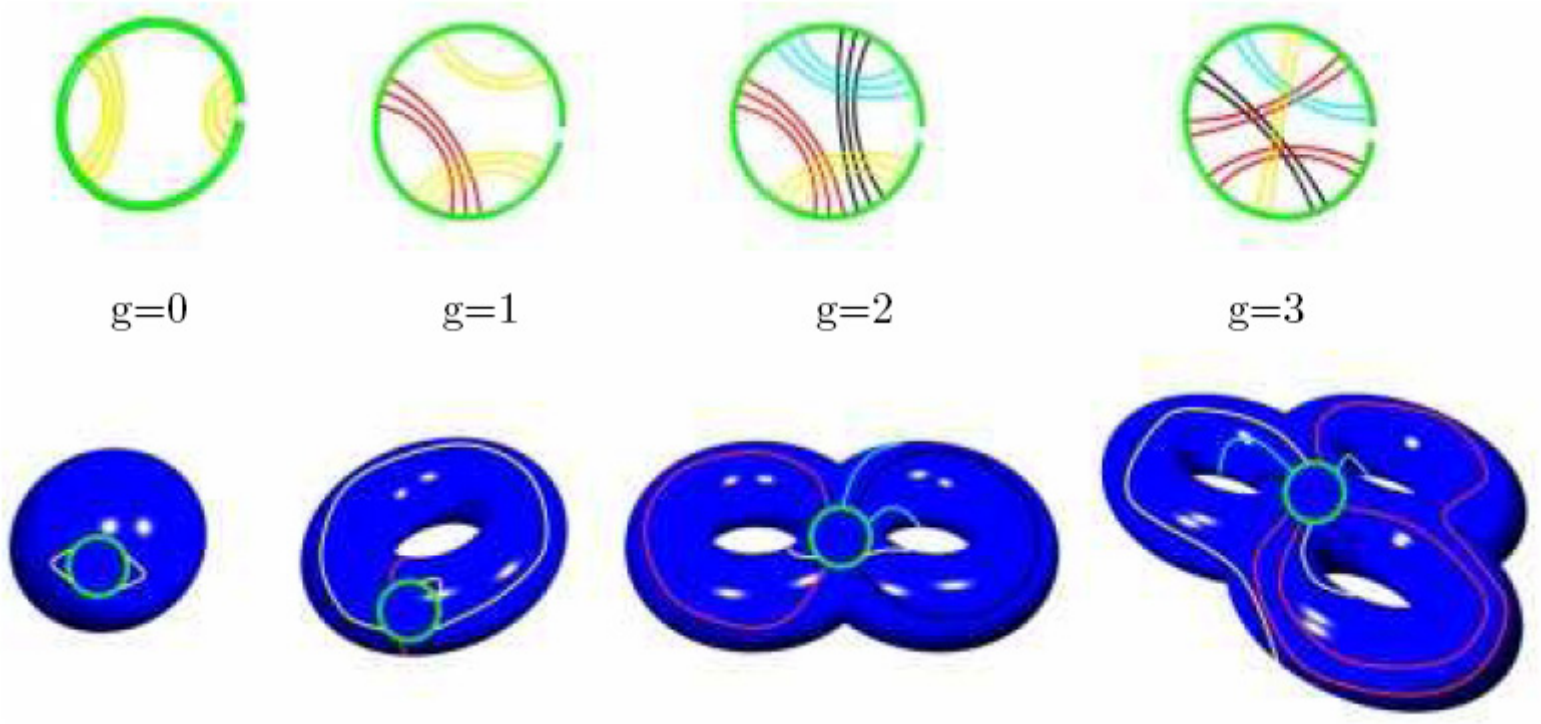
- by minimizing free energy (most prominent),
- by stochastic approaches,
- ...

Forbidding pseudoknots this task is computationally feasible (running time in $\mathcal{O}(n^3)$ based on dynamic programming), allowing pseudoknots leads to an $\mathcal{NP}$-complete problem (LYNGSØ AND PEDERSEN 2000) even for a rather simple model of free energy (nearest neighbor model).

Resort: Limit the types of legal pseudoknots (*zoo* of examples).

Algorithmic challenges

Mathematically driven approaches (BON ET AL. 2008): Restrict topological genus of structures:



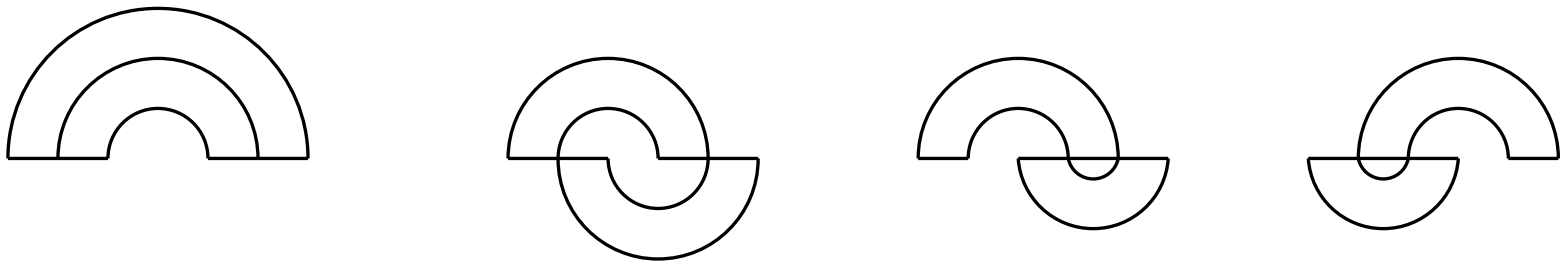
Prediction algorithms work on corresponding subclasses (e.g. N. ET AL. 2011).

Algorithmic challenges

Algorithmically driven approaches: Restrict structures according to decomposition used by (dynamic programming) algorithm:

Most prominent and successful subclass: RIVAS AND EDDY (1999)

- Without pseudoknots: Prediction by optimizing the conformation of connected subsequences which are elongated step by step.
- Here: Subproblems are given by regions with 1 gap; optimal (partial) conformations can be combined according to the following patterns:

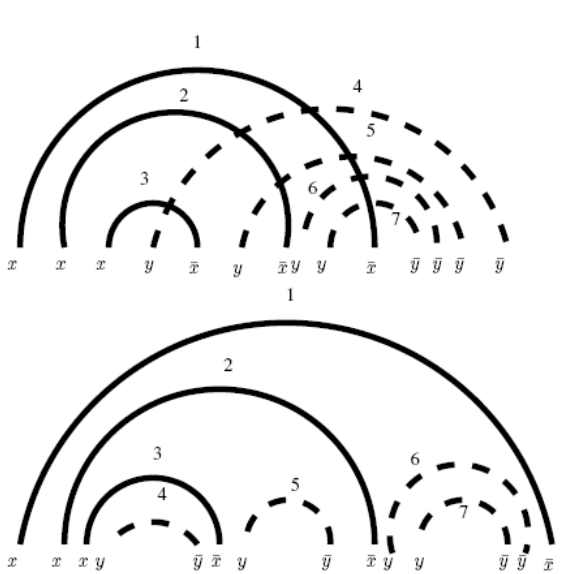


- Runtime $\mathcal{O}(n^6)$, memory $\mathcal{O}(n^4)$.

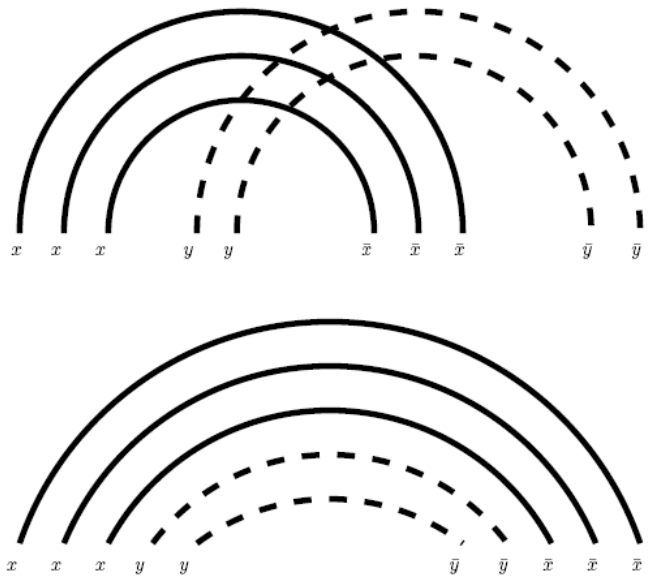
Mathematical challenges

Question: Quantify the portion of all possible pseudoknots ($\sim \sqrt{2} \cdot 2^n \cdot \left(\frac{n}{e}\right)^n$) covered by R&E (and other classes).

Approach: (SAULE, RÉGNIER, STEYAERT, DENISE 2010) bijective combinatorics



(a) Simple pseudoknot



(b) H-type pseudoknot

Remark: In the before mentioned investigation unpaired nucleotides (symbols $\circ$) were neglected!!

Mathematical challenges

However, SAULE ET AL. were not able to apply this approach to all known classes especially not to the R&E class; the corresponding enumeration problem was again left open (now for almost 12 years).

Our solution:

- Get unified algebraic descriptions of the different classes first (not part of this talk),
- translate them into a special kind of grammar (here: find appropriate grammar),
- which are used for enumeration purposes (by means of generating functions).

Stay tuned for details!

Enumeration by multiple context-free grammars

Basic concepts and definitions

A context-free grammar

- is a mechanism to generate sets of strings (language $\mathcal{L}$),
- which allows to derive a system of equations for the generating function $\sum_{w \in \mathcal{L}} z^{|w|}$.

Such a grammar is given by

- two disjoint alphabets I and T of intermediate and terminal symbols respectively, and
- a distinguished intermediate symbol S called axiom, and
- a set of rules/productions P specifying how intermediate symbols can be replaced by strings over $I \cup T$.

Enumeration by multiple context-free grammars

Basic concepts and definitions

A context-free grammar

- is a mechanism to generate sets of strings (language $\mathcal{L}$),
- which allows to derive a system of equations for the generating function $\sum_{w \in \mathcal{L}} z^{|w|}$.

Such a grammar is given by

- two disjoint alphabets I and T of intermediate and terminal symbols respectively, and
- a distinguished intermediate symbol S called axiom, and
- a set of rules/productions P specifying how intermediate symbols can be replaced by strings over $I \cup T$.

Basic concepts and definitions

Example

- CFG $G = (I,T,P,S) = \big(\{S\}, \{(,)\}, \{S \to (S)\,S, S \to \epsilon\}, S\big)$
- $S \Rightarrow (S)\,S \Rightarrow (S)\,(S)\,S \Rightarrow \underbrace{((S)\,S)\,(S)\,S}_{\text{sentential form}} \Rightarrow (())\,()$
- $S \to (S)\,S, S \to \epsilon \quad \Longrightarrow \quad S(z) = z^2 S(z)^2 + 1$

Bad news: Representation of pseudoknot classes cannot be generated (without bijection) by context-free grammar, i.e. use of CFGs leads to bijective combinatorics.

Basic concepts and definitions

Example

- CFG $G = (I, T, P, S) = \big(\{S\}, \{(,)\}, \{S \to (S)S, S \to \epsilon\}, S\big)$
- $S \Rightarrow (S)S \Rightarrow (S)(S)S \Rightarrow \underbrace{((S)S)(S)S}_{\text{sentential form}} \Rightarrow (())()$
- $S \to (S)S, S \to \epsilon \quad \rightsquigarrow \quad S(z) = z^2 S(z)^2 + 1$

Bad news: Representation of pseudoknot classes cannot be generated (without bijection) by context-free grammar, i.e. use of CFGs leads to bijective combinatorics.

Basic concepts and definitions

Example

- CFG $G = (I,T,P,S) = \big(\{S\}, \{(,)\}, \{S \to (S)S, S \to \epsilon\}, S\big)$
- $S \Rightarrow (S)S \Rightarrow (S)(S)S \Rightarrow \underbrace{((S)S)(S)S}_{\text{sentential form}} \Rightarrow (())()$
- $S \to (S)S, S \to \epsilon \quad \equiv \quad S(z) = z^2 S(z)^2 + 1$

Bad news: Representation of pseudoknot classes cannot be generated (without bijection) by context-free grammar, i.e. use of CFGs leads to bijective combinatorics.

Basic concepts and definitions

Example

- CFG $G = (I,T,P,S) = \big(\{S\}, \{(,)\}, \{S \to (S)S, S \to \epsilon\}, S\big)$
- $S \Rightarrow (S)S \Rightarrow (S)(S)S \Rightarrow \underbrace{((S)S)(S)S}_{\text{sentential form}} \Rightarrow (())()$
- $S \to (S)S, S \to \epsilon \quad \equiv \quad S(z) = z^2 S(z)^2 + 1$

Bad news: Representation of pseudoknot classes cannot be generated (without bijection) by context-free grammar, i.e. use of CFGs leads to bijective combinatorics.

Basic concepts and definitions

Example

- CFG $G = (I,T,P,S) = \big(\{S\}, \{(,)\}, \{S \to (S)S, S \to \epsilon\}, S\big)$
- $S \Rightarrow (S)S \Rightarrow (S)(S)S \Rightarrow \underbrace{((S)S)(S)S}_{\text{sentential form}} \Rightarrow (())()$
- $S \to (S)S, S \to \epsilon \quad \equiv \quad S(z) = z^2 S(z)^2 + 1$

Bad news: Representation of pseudoknot classes cannot be generated (without bijection) by context-free grammar, i.e. use of CFGs leads to bijective combinatorics.

Basic concepts and definitions

Example

- CFG $G = (I,T,P,S) = \big(\{S\}, \{(,)\}, \{S \to (S)S, S \to \epsilon\}, S\big)$
- $S \Rightarrow (S)S \Rightarrow (S)(S)S \Rightarrow \underbrace{((S)S)(S)S}_{\text{sentential form}} \Rightarrow (())()$
- $S \to (S)S, S \to \epsilon \quad \equiv \quad S(z) = z^2 S(z)^2 + 1$

Bad news: Representation of pseudoknot classes cannot be generated (without bijection) by context-free grammar, i.e. use of CFGs leads to bijective combinatorics.

Basic concepts and definitions

Multiple context-free grammars (MCFGs): Intermediate symbols now have multiple components

- which are replaced in a coupled way,
- while appearing detached in the sentential forms.

Example: $\ldots\alpha A_1 \beta A_2 \gamma \ldots \Rightarrow \alpha[\beta]\gamma$ for $A_1, A_2 \to [,] \in P$.

Goal: Find unambiguous (one tree per word) MCFGs which generate the (language of the) R&E class (without bijection).

Key ideas:

- Represent pair of corresponding brackets $[\cdots]$ by two dimensional intermediate $A_1 \cdots A_2$ (plus rule $A_1, A_2 \to [,]$).
- To enforce unambiguity, different intermediates are used depending on which operation introduces the brackets.
- Find set of rules to generate the different ways *gapped structures* may be combined.

Basic concepts and definitions

Multiple context-free grammars (MCFGs): Intermediate symbols now have multiple components

- which are replaced in a coupled way,
- while appearing detached in the sentential forms.

Example: $\ldots\alpha A_1 \beta A_2 \gamma \ldots \Rightarrow \alpha[\beta]\gamma$ for $A_1, A_2 \to [,] \in P$.

Goal: Find unambiguous (one tree per word) MCFGs which generate the (language of the) R&E class (without bijection).

Key ideas:

- Represent pair of corresponding brackets $[\cdots]$ by two dimensional intermediate $A_1 \cdots A_2$ (plus rule $A_1, A_2 \to [,]$).
- To enforce unambiguity, different intermediates are used depending on which operation introduces the brackets.
- Find set of rules to generate the different ways *gapped structures* may be combined.

Basic concepts and definitions

Multiple context-free grammars (MCFGs): Intermediate symbols now have multiple components

- which are replaced in a coupled way,
- while appearing detached in the sentential forms.

Example: $\ldots\alpha A_1 \beta A_2 \gamma \ldots \Rightarrow \alpha[\beta]\gamma$ for $A_1, A_2 \to [,] \in P$.

Goal: Find unambiguous (one tree per word) MCFGs which generate the (language of the) R&E class (without bijection).

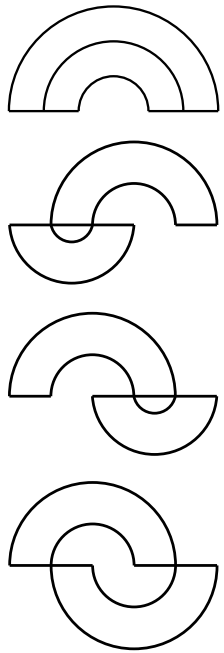
Key ideas:

- Represent pair of corresponding brackets $[\cdots]$ by two dimensional intermediate $A_1 \cdots A_2$ (plus rule $A_1, A_2 \to [,]$).
- To enforce unambiguity, different intermediates are used depending on which operation introduces the brackets.
- Find set of rules to generate the different ways *gapped structures* may be combined.

Grammar for the R&E class

Modulo ambiguity, the following grammar allows to *simulate* the recursive decomposition of R&E:

$$
\begin{array}{rcl}
S & \to & T_1\, T_2 \\
T_1, T_2 & \to & [S,]S \\
 & & T_1\, T_1', T_2'\, T_2 \\
 & & T_1\, T_1'\, T_2, T_2' \\
 & & T_1, T_1'\, T_2\, T_2' \\
 & & T_1\, T_1', T_2\, T_2'
\end{array}
$$



T_1', T_2' a distinguished copy of T_1, T_2.

Note that for enumeration purposes we do not need to distinguish between different kinds of brackets.

Grammar for the R&E class

$$
\begin{aligned}
S &\rightarrow \epsilon + \circ S + [S]S + K_1[SK_2]S, \\
K_1, K_2 &\rightarrow K_1 I_1, I_2 K_2 + I_1 K_1 I_2, K_2 + D_1, I_1 D_2 I_2 + L_1 K_1, L_2 K_2 + [S,]S, \\
I_1, I_2 &\rightarrow I_1 K_1 I_2, K_2 + D_1, I_1 D_2 I_2 + L_1 K_1, L_2 K_2 + [S,]S, \\
D_1, D_2 &\rightarrow K_1 I_1, I_2 K_2 + D_1, I_1 D_2 I_2 + L_1 K_1, L_2 K_2 + [S,]S, \\
L_1, L_2 &\rightarrow K_1 I_1, I_2 K_2 + I_1 K_1 I_2, K_2 + D_1, I_1 D_2 I_2 + [S,]S,
\end{aligned}
$$

automatically derived from an algebraic description of the pseudoknot class.

A general enumeration lemma

What do we gain? A lot when considering the ring of formal power series.

Advantage: Going forth and back from grammar to series to grammar allows to

- *reorder* the terminal symbols in the words generated (by commutativity of the series) without affecting their numbers (similar to bijective combinatorics) and the size of the language generated,
- find a plain context-free grammar which generates the corresponding (reordered) language of exactly the same sizes,
- use well-known enumeration techniques for context-free languages.

A general enumeration lemma

What do we gain? A lot when considering the ring of formal power series.

Advantage: Going forth and back from grammar to series to grammar allows to

- *reorder* the terminal symbols in the words generated (by commutativity of the series) without affecting their numbers (similar to bijective combinatorics) and the size of the language generated,
- find a plain context-free grammar which generates the corresponding (reordered) language of exactly the same sizes,
- use well-known enumeration techniques for context-free languages.

A general enumeration lemma

What do we gain? A lot when considering the ring of formal power series.

Advantage: Going forth and back from grammar to series to grammar allows to

- *reorder* the terminal symbols in the words generated (by commutativity of the series) without affecting their numbers (similar to bijective combinatorics) and the size of the language generated,
- find a plain context-free grammar which generates the corresponding (reordered) language of exactly the same sizes,
- use well-known enumeration techniques for context-free languages.

This heuristic explanation opens the way to prove the following lemma.

A general enumeration lemma

Lemma

Let $G = (I, d, T, P, \vec{X}^{<1>})$, $I = \{\vec{X}^{<1>}, \ldots, \vec{X}^{<k>}\}$, *an unambiguous* MCFG *without* ε*-rules*[1] *and* $\mathcal{SE}$ *the system of equations where for each* $\vec{X}^{<i>} \in I$ *the following variable and corresponding equation is introduced:*

$$X^{(i)} = \sum_{\vec{\alpha}:\vec{X}^{<i>} \to \vec{\alpha} \in P} \prod_{1 \leqslant k \leqslant d(X^{<i>})} h(\alpha_k),$$

where h *is the substitution that maps* $a \in T$ *to variable* z, *the first component* $X_1^{<j>}$ *of any intermediate symbol to* $X^{(j)}$ *and its other components to* 1 *(replacing concatenation of symbols by multiplication). If* $X^{(1)}(z)$ *denotes the generating function obtained from solving* $\mathcal{SE}$ *for* $X^{(1)}$, *choosing the unique solution compatible with initial conditions, then* $[z^n]X^{(1)}(z) = |\mathcal{L}(G) \cap T^n|$ *holds.*

[1] For MCFGs an ε-rule is given by a rule like $\vec{A} \to (\varepsilon, \ldots, \varepsilon)^T$.

Results

Table: The asymptotical number of pseudoknot structures of size n.

Class	Asymp.	size = arcs		size = all		Run time
		α	ω	α	ω	algo.
L&P	$\alpha\omega^n$	$\frac{1}{2}$	4	$\frac{1}{4}$	3	$\mathcal{O}(n^5)$
C&C	$\frac{\alpha}{2\sqrt{\pi n^3}}\omega^n$	1.6651	5.857	4.0599	3.2864	$\mathcal{O}(n^6)$
R&G	$\frac{\alpha}{2\sqrt{\pi n^3}}\omega^n$	0.1651	6.576	2.7058	3.5129	$\mathcal{O}(n^4)$
D&P	$\frac{\alpha}{2\sqrt{\pi n^3}}\omega^n$	0.7535	7.315	1.7082	3.7046	$\mathcal{O}(n^5)$
A&U	$\frac{\alpha}{2\sqrt{\pi n^3}}\omega^n$	0.6575	7.547	1.4813	3.7472	$\mathcal{O}(n^6)$
N&R	$\frac{\alpha}{2\sqrt{\pi n^3}}\omega^n$	0.6429	8.284	1.4222	3.8782	$\mathcal{O}(n^6)$
R&E	$\frac{\alpha}{2\sqrt{\pi n^3}}\omega^n$	0.0176	15.792	0.0348	4.9739	$\mathcal{O}(n^6)$

Conclusion

Making use of multiple context-free grammars, formal power series and generating functions, we have

- introduced a new enumeration lemma applicable to objects with mildly context sensitive encodings,
- making the search for a bijection superfluous.

Those general findings provided an elegant and surprisingly simple solution to the problem of enumerating the R&E class.

Our ideas can be used to handle special kinds of couplings within recursive decompositions. Accordingly, we also made used of our approach to the analysis of algorithms e.g. to analyze the size of the intersection of two random trees (as a proof of concept).

Conclusion

Thanks for your attention!