Algorithmic Aspects of RNA Secondary Structures

Stéphane Vialette

CNRS & LIGM, Université Paris-Est Marne-la-Vallée, France

2016-1017

Plan

Computational biology

- Computational biology involves the development and application of data-analytical and theoretical methods, mathematical modeling and computational simulation techniques to the study of biological, behavioral, and social systems.
  The field is broadly defined and includes foundations in computer science, applied mathematics, animation, statistics, biochemistry, chemistry, biophysics, molecular biology, genetics, genomics, ecology, evolution, anatomy, neuroscience, and visualization.

- Computational biology is different from biological computation, which is a subfield of computer science and computer engineering using bioengineering and biology to build computers, but is similar to bioinformatics, which is an interdisciplinary science using computers to store and process biological data.

Computational biology – Subfields

- Computational biomodeling
- Computational genomics (Computational genetics)
- Computational neuroscience
- Computational pharmacology
- Computational evolutionary biology
- Cancer computational biology
- …

Computational biology – Major conferences

- Workshop on Algorithms in Bioinformatics (WABI)
- Asia Pacific Bioinformatics Conference (APBC)
- Intelligent Systems for Molecular Biology (ISMB)
- European Conference on Computational Biology (ECCB)
- Research in Computational Molecular Biology (RECOMB)

Computational biology – RECOMB satellites

- RECOMB Satellite Workshop on Massively Parallel Sequencing (RECOMB-SEQ)
- RECOMB Satellite Conference on Bioinformatics Education (RECOMB-BE)
- RECOMB Satellite Workshop on Computational Cancer Biology (RECOMB-CCB)
- RECOMB Satellite Workshop on Computational Methods in Genetics (RECOMB-Genetics)
- RECOMB Satellite Workshop on Comparative Genomics (RECOMB-CG)
- RECOMB Satellite Conference on Open Problems in Algorithmic Biology

Computational biology – Main journals

There are numerous journals dedicated to computational biology:

- Journal of Computational Biology
- PLOS Computational Biology
- Bionformatics
- IEEE/ACM Transactions on Computational Biology and Bioinformatics
- BMC Bioinformatics
- Journal of Bioinformatics and Computational Biology
- …

Société Française de Bioinformatique (SFBI)

Société Française de BioInformatique

Pour la promotion de la recherche pluridisciplinaire autour des Sciences du Vivant

Rechercher

Connexion utilisateur

Connexion/Inscription

Menu

- La SFBI
  - Présentation
  - Conseil
  - Statuts
  - Adhésion
  - Rapports Activité
- Emplois
- La bioinfo en France
  - Équipes Françaises
  - Formations
    - Formations universitaires
    - Formations permanentes
    - Ateliers, écoles thématiques à venir
  - Groupes de travail
  - Liens
- Colloques
- Prix SFBI à JOBIM
- Bourses de voyages SFBI
- Publications
  - Thèses
  - HDR
  - Ouvrages
- Restez informés!
  - Calendrier des événements
  - Bioinformations
  - La liste bioinfo
    - Archives
    - Inscription liste bioinfo
- Le site web
- Logos SFBI

Bienvenue sur le site web de la SFBI

La Société Française de BioInformatique (**SFBI**) a été créée en 2005 pour promouvoir la recherche pluri- et inter-disciplinaire en Bioinformatique, à l'interface entre la Biologie moléculaire, l'Informatique, les Mathématiques et la Physique.

La SFBI a vocation à rassembler les chercheurs, enseignants et ingénieurs francophones qui travaillent sur des questions relevant de la bioinformatique, pour des rencontres et des échanges scientifiques. Elle participe et assure la pérénité de l'organisation de **JOBIM**, conférence nationale annuelle de bioinformatique qui réunit de 300 à 500 personnes.

La SFBI assure la diffusion au sein de la communauté bioinformatique francophone (liste de diffusion modérée **bioinfo@sfbi.fr**) d'offres de recrutements, de stages et de thèses, et d'informations sur les formations en bioinformatique, sur les manifestations scientifiques nationales et internationales relevant de la bioinformatique.

Ce site se veut une vitrine de la recherche et de la formation en bioinformatique en France.

Pour tous renseignements : sfbi.contact@gmail.com

Brèves

prix SFBI décernés à Jobim 2016

Bravo à Olivier Chabrol qui a remporté le prix de la meilleur présentation orale décerné par la SFBI, et à Chrystelle Delord et Valentine Rech de Laval qui ont reçu les prix des deux meilleurs posters !

Voir tous les prix SFBI à JOBIM.

Lire la suite.....

prix SFBI de la meilleure présentation orale à Jobim 2015

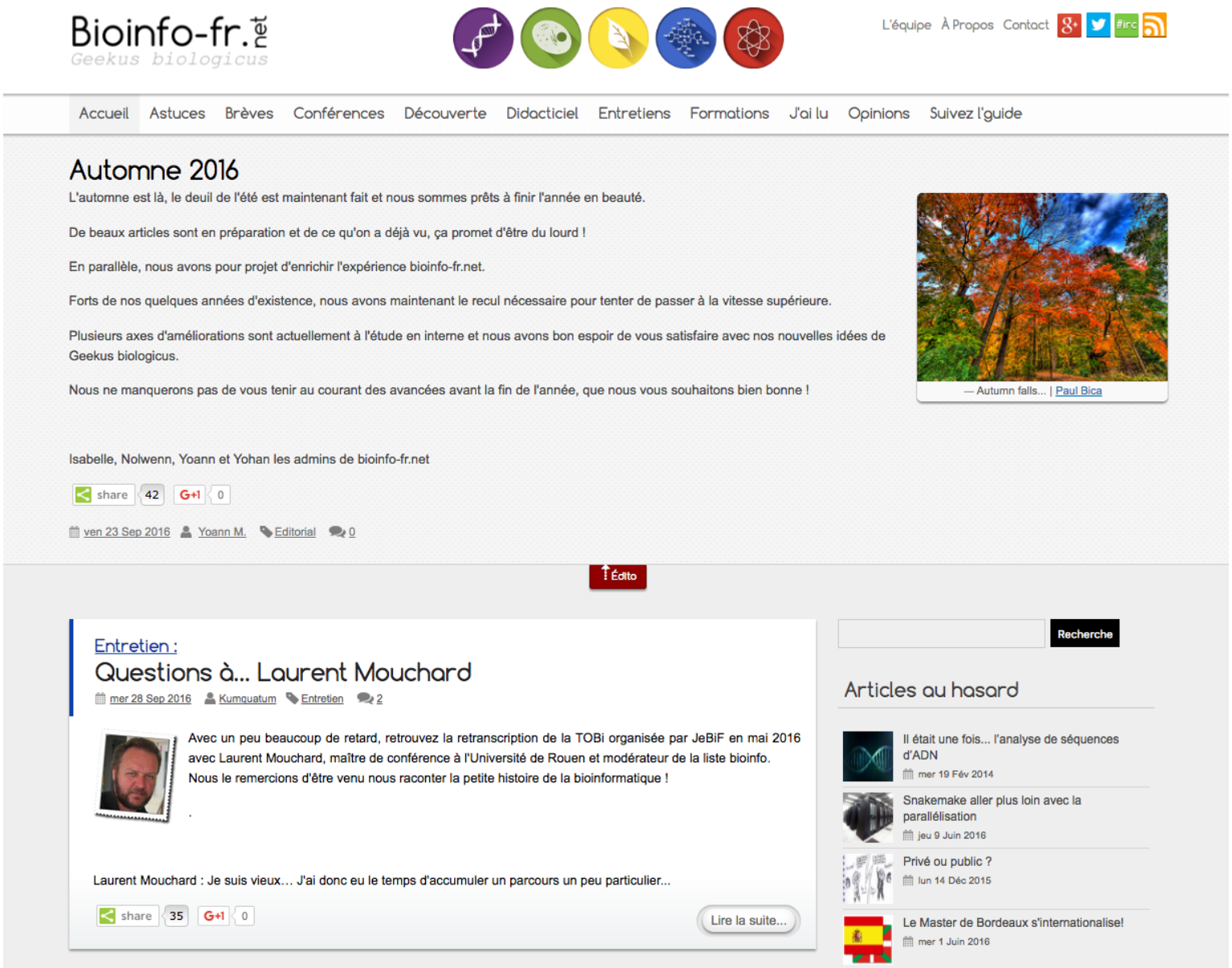
Bravo à **Isaure Chauvot de Beauchêne** qui a remporté ce premier prix décerné par la SFBI !

Plus d'infos sur sa présentation

Lire la suite.....

Association des Jeunes Bioinformaticiens de France (JeBiF)

http://bioinfo-fr.net/



GdR de Bioinformatique Moléculaire (GdR BIM – INS2I CNRS)

Carrières académiques

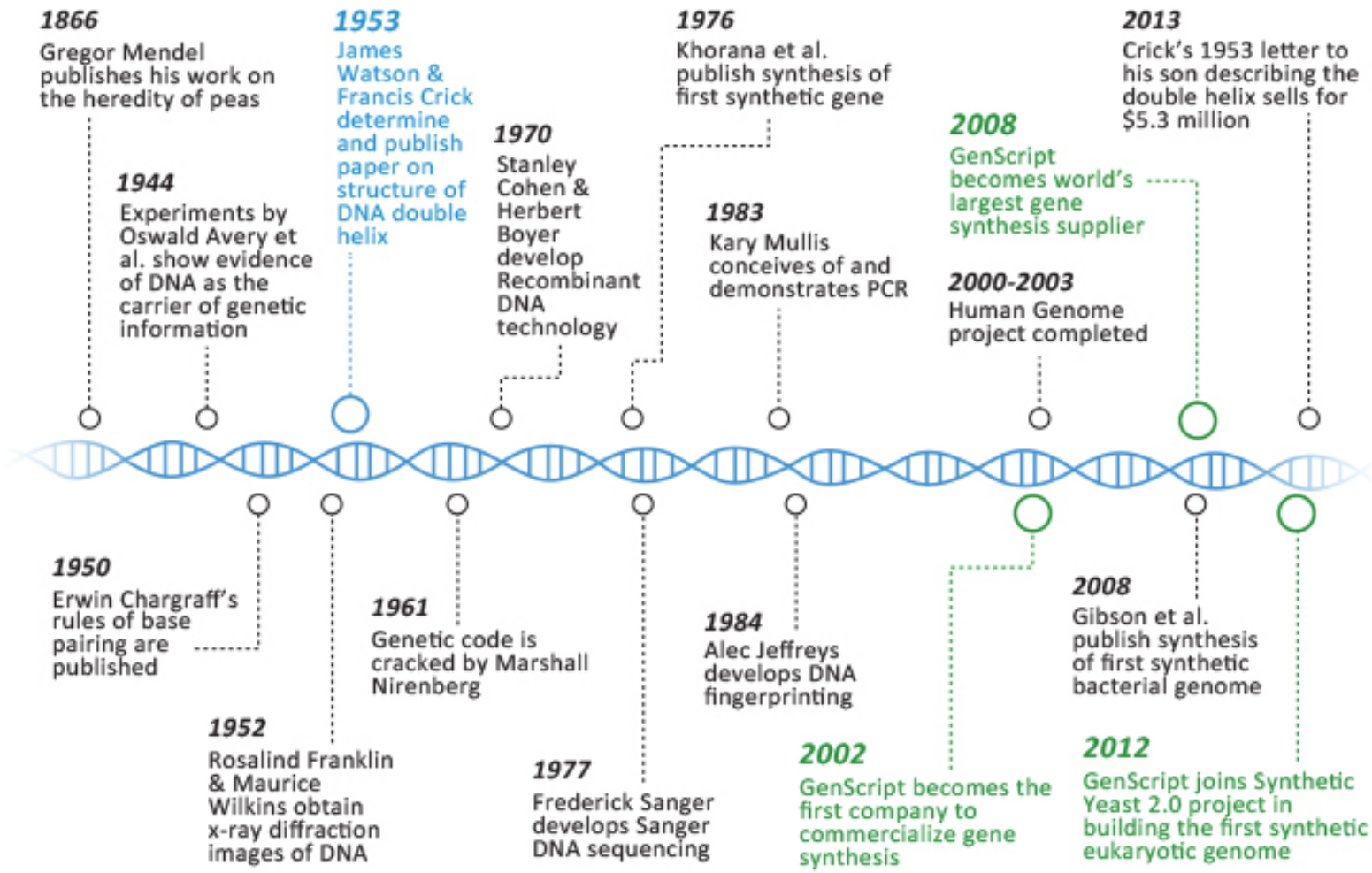
- Maître de Conférences / Professeur des Universités
- CNRS : Section 06, Section 07 et CID 51 (Modélisation, et analyse des données et des systèmes biologiques : approches informatiques, mathématiques et physiques)
- INRIA
- INRA
- CEA
- Institut Pasteur
- Institut Curie

AlgoB :: Algorithmics for Bioinformatics :: Home

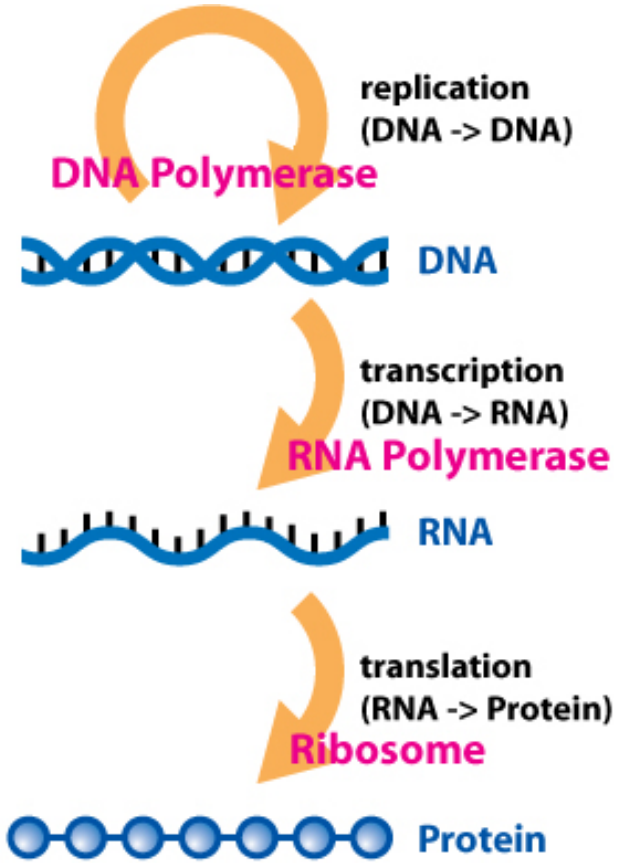
AlgoB :: Algorithmics for Bioinformatics :: Home

Plan

Timeline


1866
Gregor Mendel publishes his work on the heredity of peas
1944
Experiments by Oswald Avery et al. show evidence of DNA as the carrier of genetic information
1953
James Watson & Francis Crick determine and publish paper on structure of DNA double helix
1970
Stanley Cohen & Herbert Boyer develop Recombinant DNA technology
1976
Khorana et al. publish synthesis of first synthetic gene
1983
Kary Mullis conceives of and demonstrates PCR
2000-2003
Human Genome project completed
2008
GenScript becomes world's largest gene synthesis supplier
2013
Crick's 1953 letter to his son describing the double helix sells for $5.3 million
1950
Erwin Chargraff's rules of base pairing are published
1952
Rosalind Franklin & Maurice Wilkins obtain x-ray diffraction images of DNA
1961
Genetic code is cracked by Marshall Nirenberg
1977
Frederick Sanger develops Sanger DNA sequencing
1984
Alec Jeffreys develops DNA fingerprinting
2002
GenScript becomes the first company to commercialize gene synthesis
2008
Gibson et al. publish synthesis of first synthetic bacterial genome
2012
GenScript joins Synthetic Yeast 2.0 project in building the first synthetic eukaryotic genome

Central dogma of molecular biology



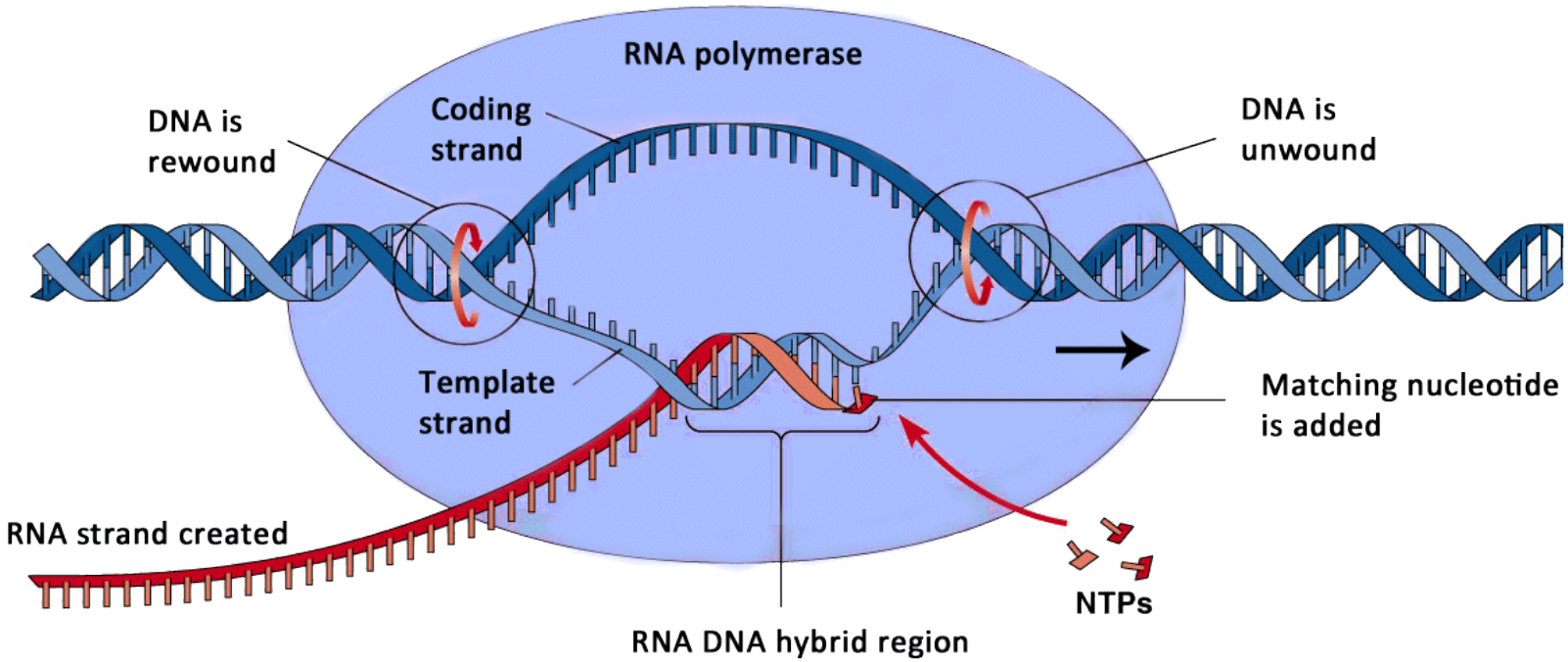
Central dogma of molecular biology

- The central dogma of molecular biology deals with the detailed residue-by-residue transfer of sequential information. It states that such information cannot be transferred back from protein to either protein or nucleic acid.
- This has also been described as *DNA makes RNA makes protein*. However, this simplification does not make it clear that the central dogma as stated by Crick does not preclude the reverse flow of information

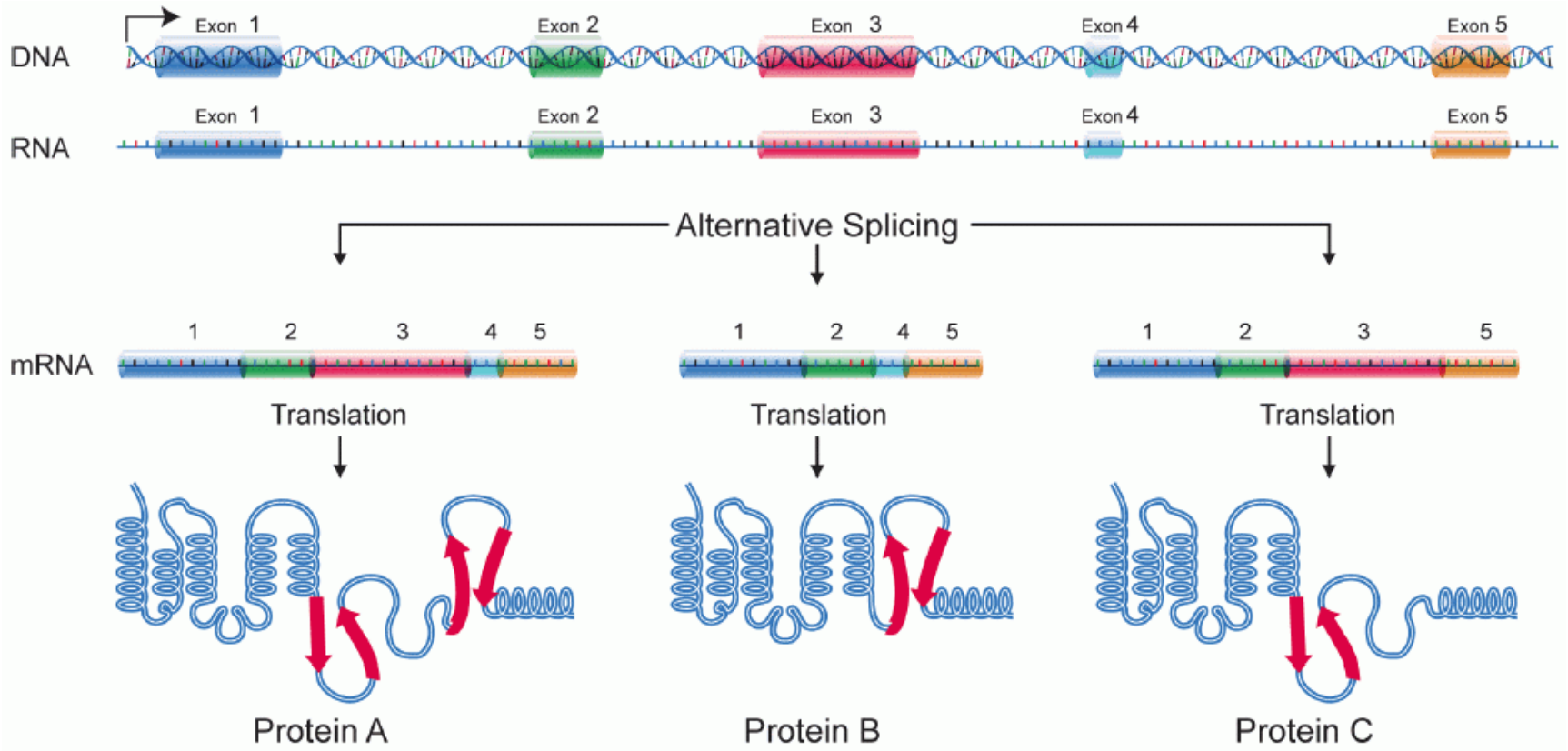
Transcription

- **Transcription** is the process by which the information contained in a section of DNA is transferred to a newly assembled piece of messenger RNA (mRNA).
- It is facilitated by RNA polymerase and transcription factors.
- In eukaryotic cells the primary transcript (pre-mRNA) must be processed further in order to ensure translation.
- This normally includes a $5'$ cap, a poly-A tail and splicing.
- Alternative splicing can also occur, which contributes to the diversity of proteins any single mRNA can produce.

Transcription



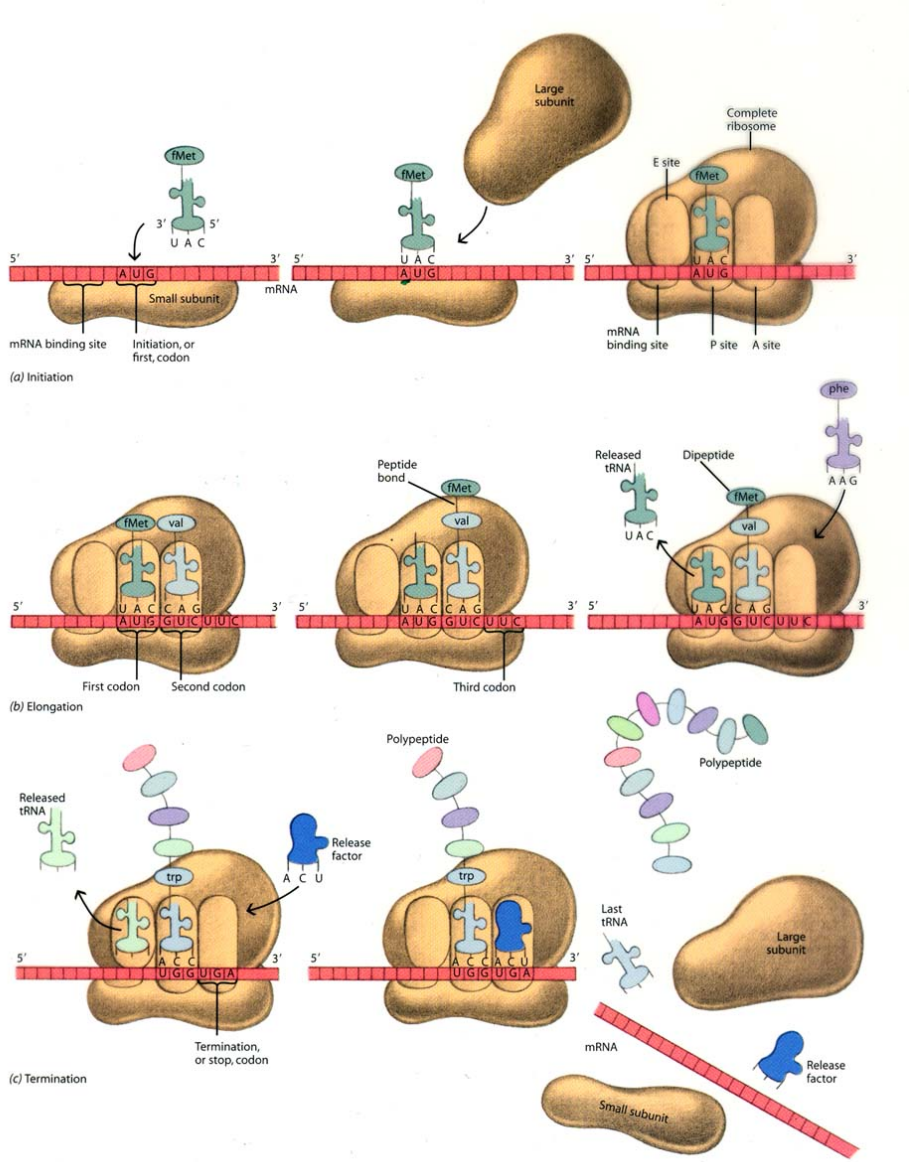
Alternative splicing



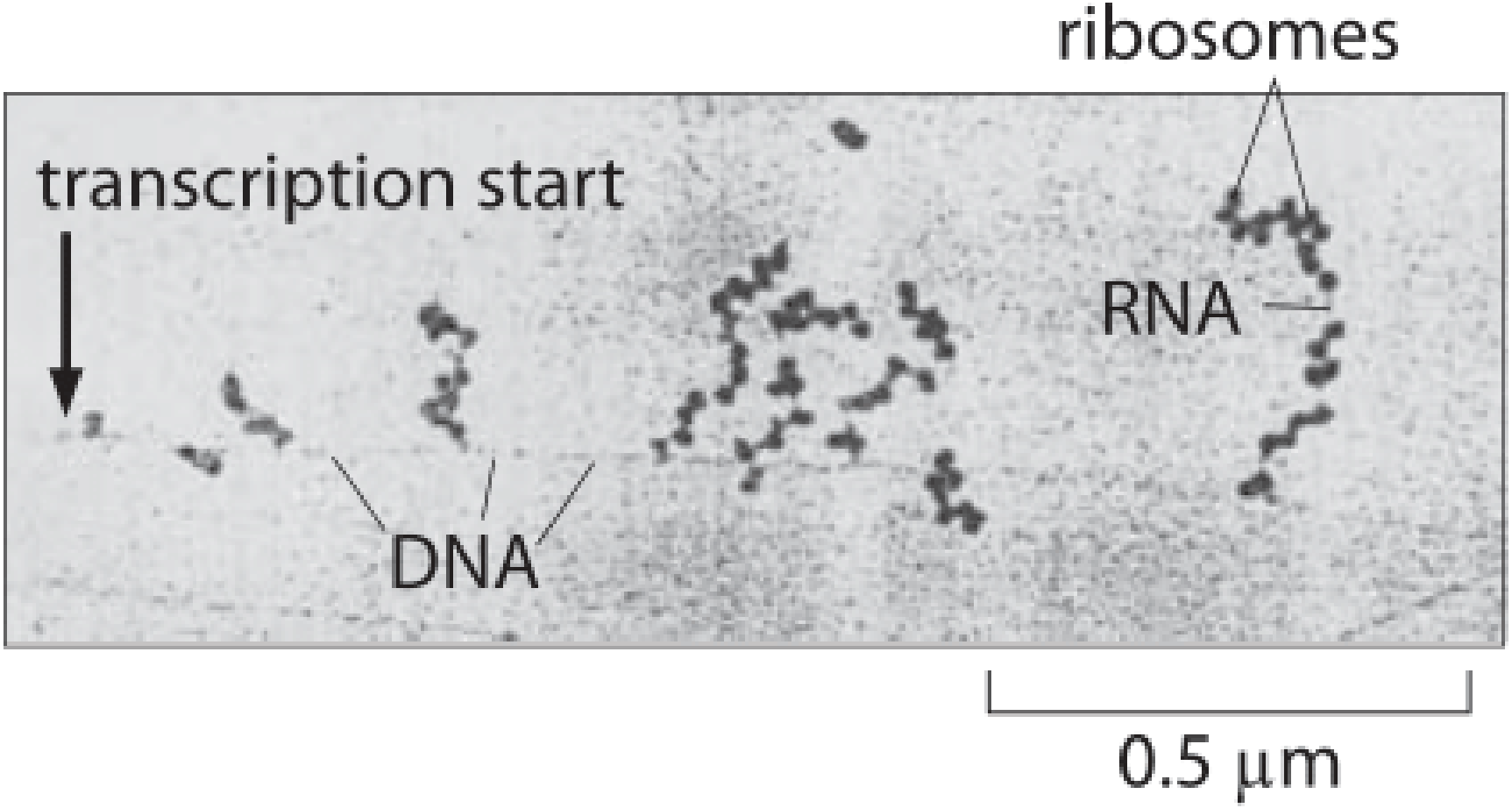
Translation

- Eventually, the mature mRNA finds its way to a ribosome, where it is **translated**.
- In prokaryotic cells, which have no nuclear compartment, the process of transcription and translation may be linked together. In eukaryotic cells, the site of transcription (the cell nucleus) is usually separated from in the site of translation (the cytoplasm), so the mRNA must be transported out of the nucleus into the cytoplasm, where it can be bound by ribosomes
- The mRNA is read by the ribosome as triplet codons, usually beginning with an AUG (adenine−uracil−guanine), or initiator methionine codon downstream of the ribosome binding site.
- Translation ends with a *UAA*, *UGA*, or *UAG* stop codon.

Translation



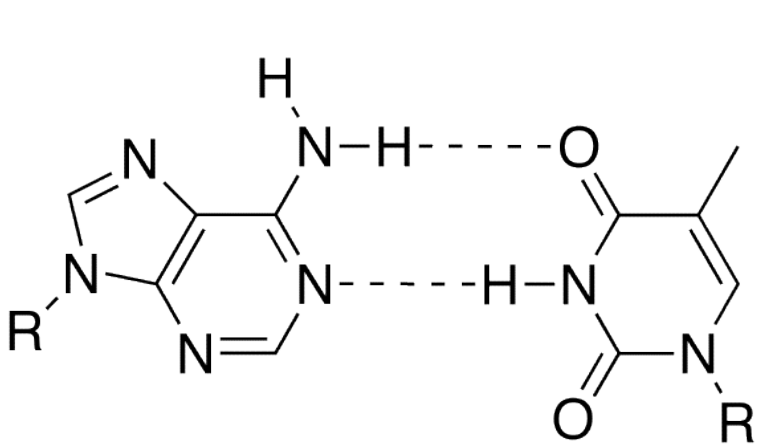
Simultaneous translation and transcription



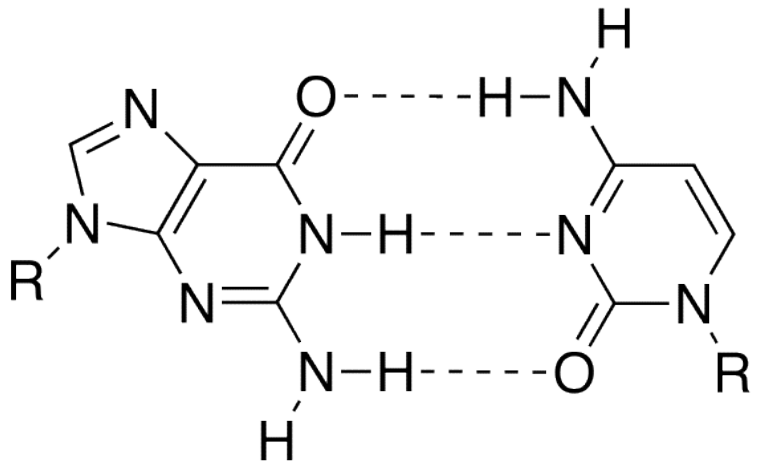
Base pairing

- In molecular biology, two nucleotides on opposite complementary DNA or RNA strands that are connected via hydrogen bonds are called a base pair (often abbreviated bp).
- In the canonical Watson-Crick base pairing, adenine (A) forms a base pair with thymine (T), and guanine (G) forms one with cytosine (C) in DNA.
- In RNA, thymine is replaced by uracil (U).
- Alternate hydrogen bonding patterns, such as the wobble base pair and Hoogsteen base pair, also occur—particularly in RNA—giving rise to complex and functional tertiary structures.
- Importantly, pairing is the mechanism by which codons on messenger RNA molecules are recognized by anticodons on transfer RNA during protein translation

Base pairing



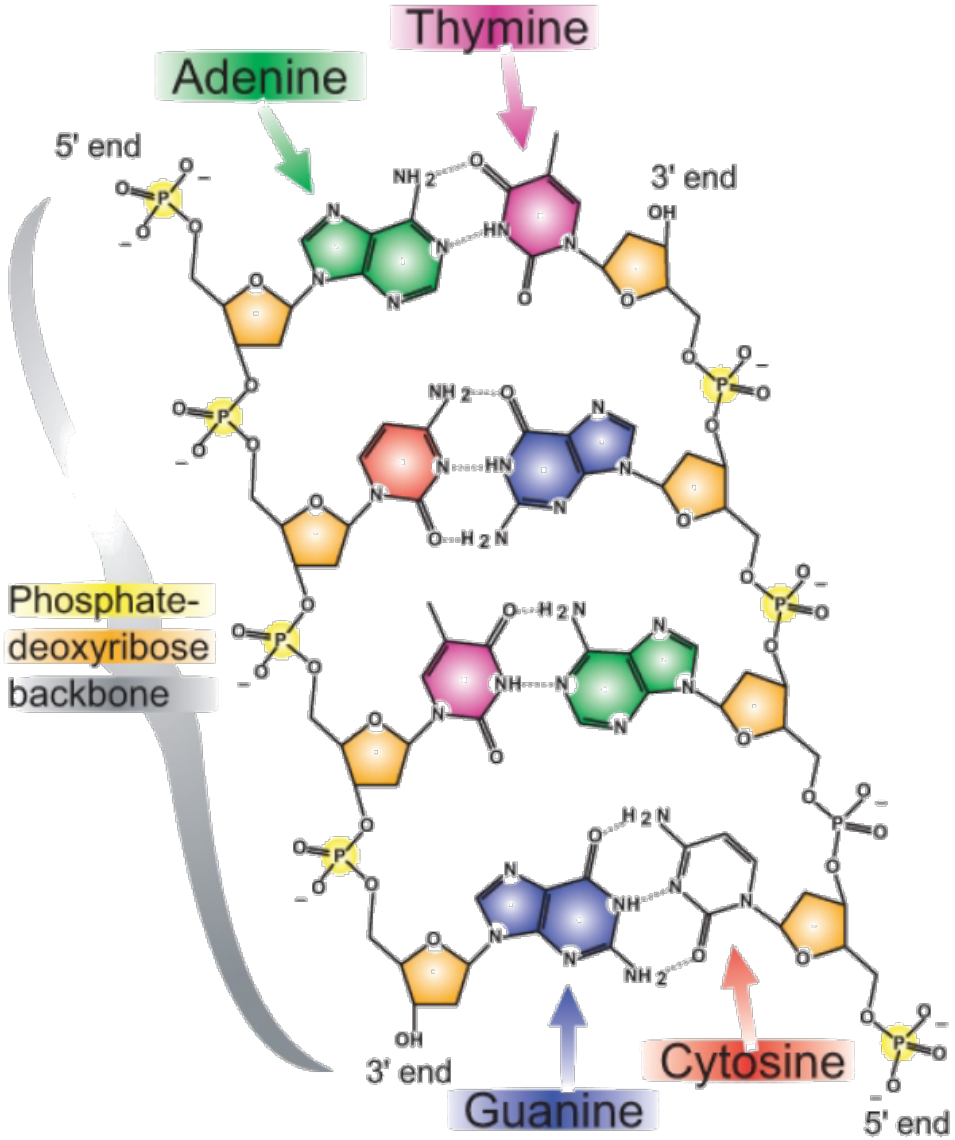
Adenine Thymine



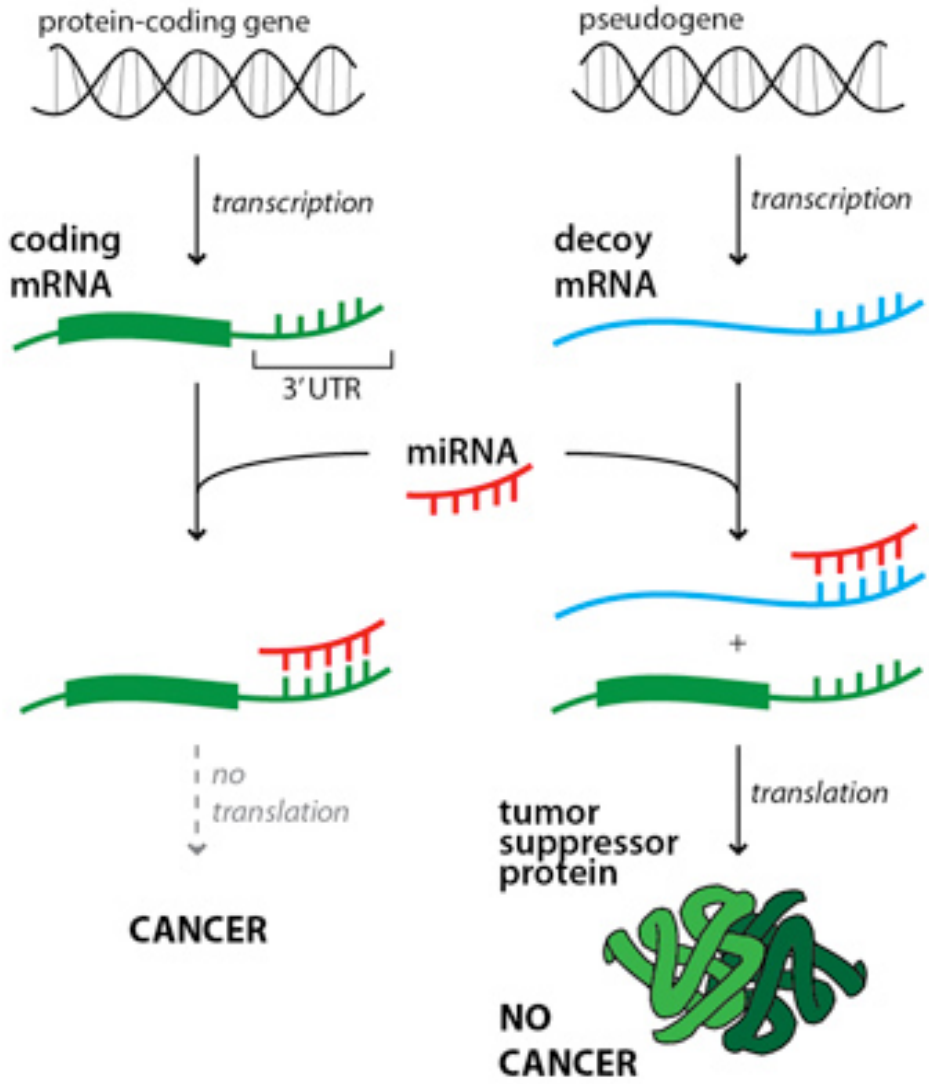
Guanine Cytosine

Left, an *AT* base pair demonstrating two intermolecular hydrogen bonds; Right, a *GC* base pair demonstrating three intermolecular hydrogen bonds.

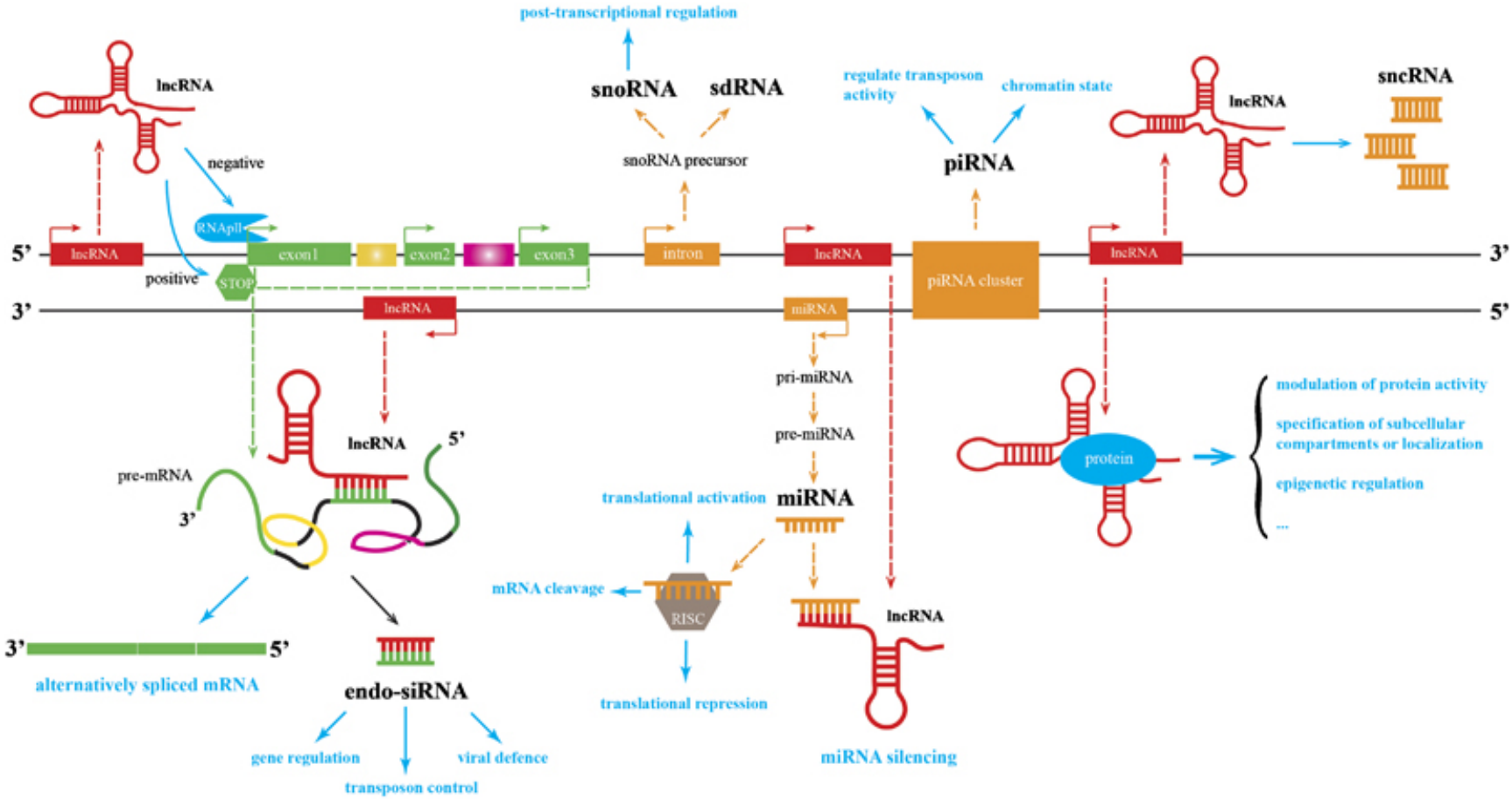
Base pairing



miRNA

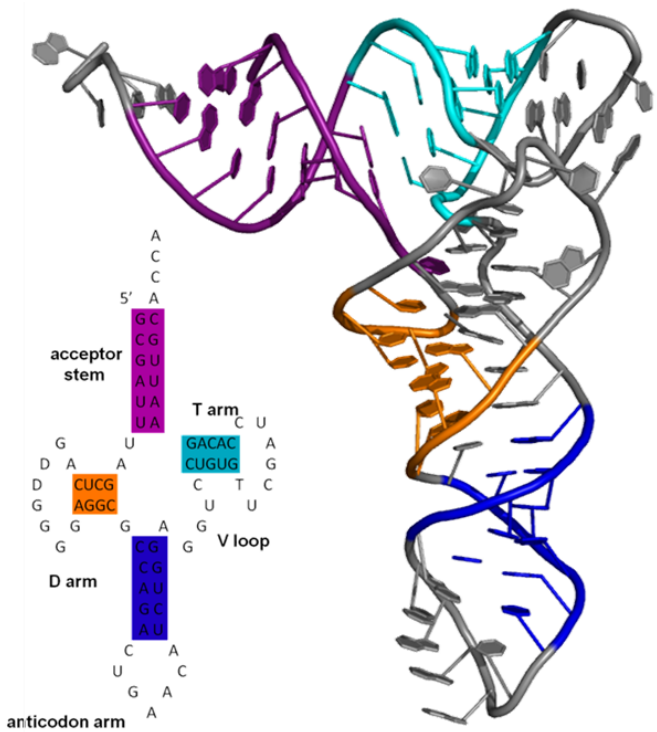

protein-coding gene
transcription
coding mRNA
3′UTR
miRNA
no translation
CANCER
pseudogene
transcription
decoy mRNA
+
translation
tumor suppressor protein
NO CANCER

Non-coding RNA

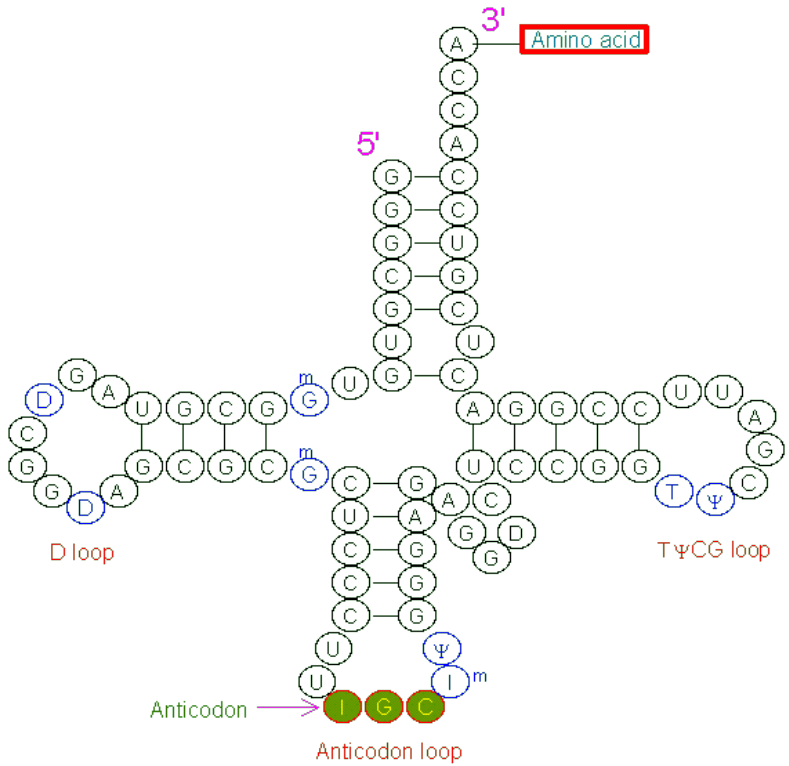


Structural conformations of biomolecules

- **Primary Structure**: sequence of monomeres (*ATCGAGATC...*)
- **Secondary Structure**: 2D-fold, defined by hydrogen bonds
- **Tertiary Structure**: 3D-fold
- **Quarternary Structure**: complex arrangement of multiple folded moleculesRNA tertiary structure

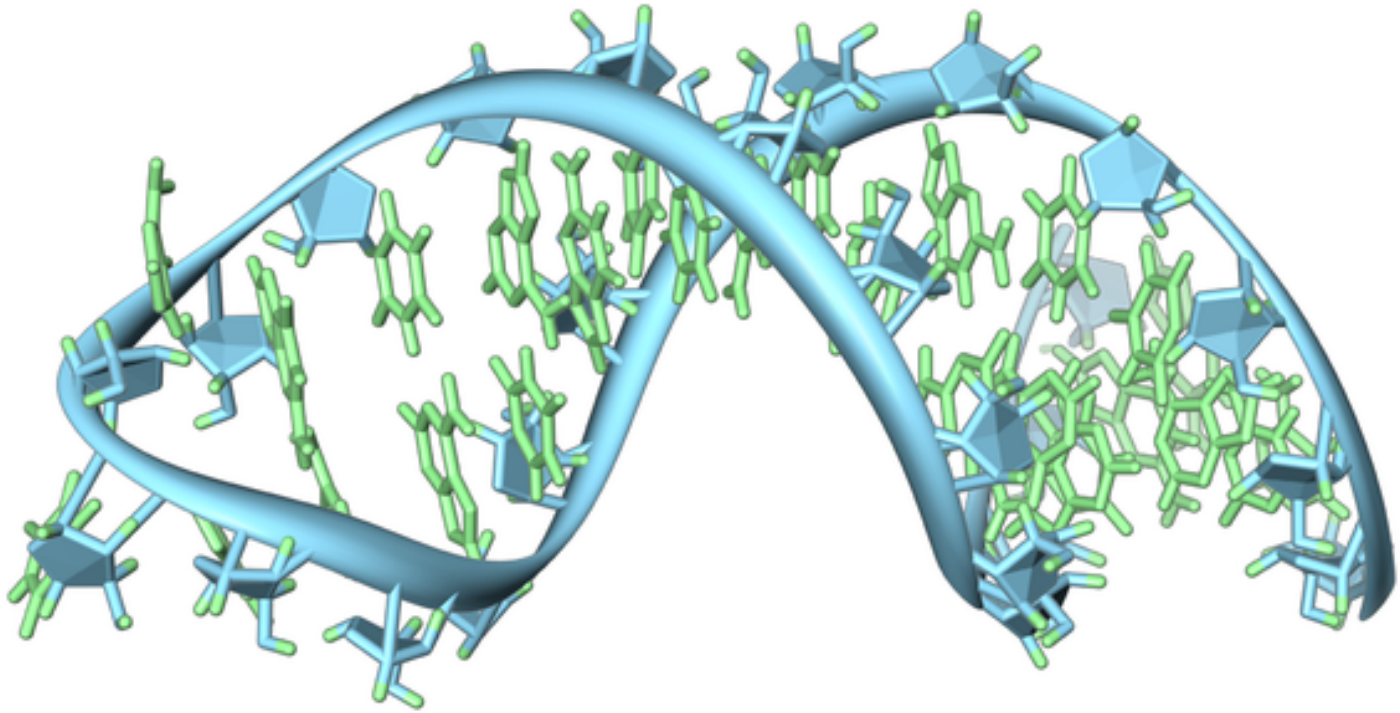


RNA seconday structure



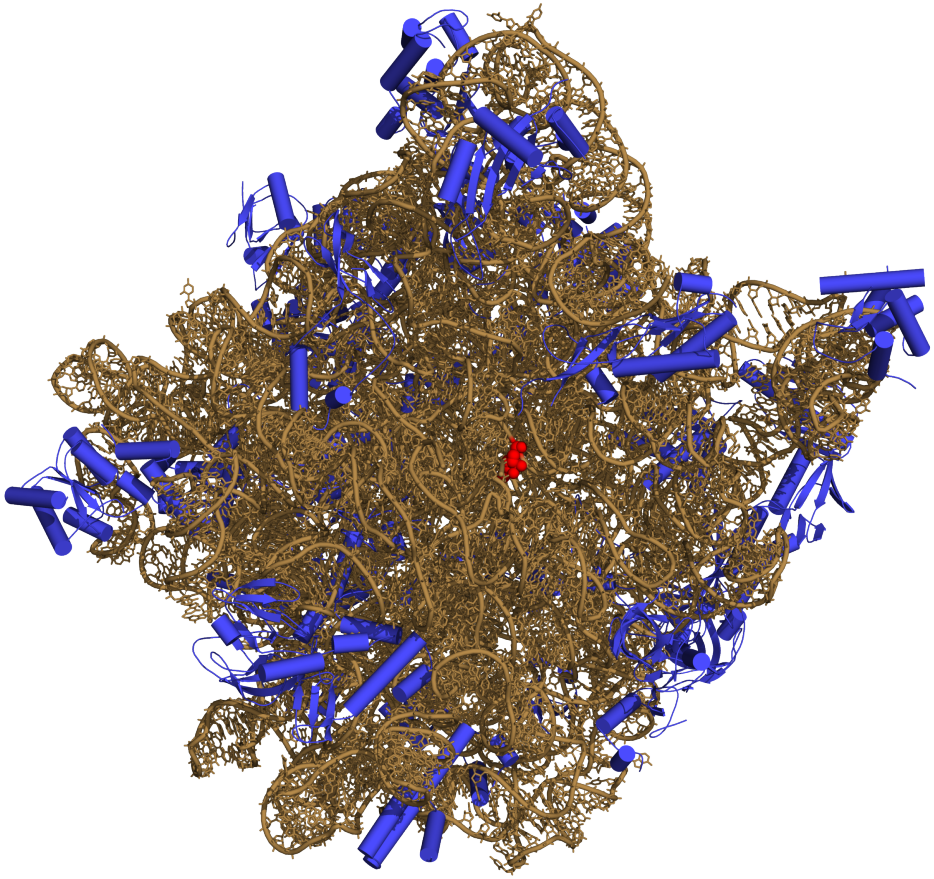
The major role of tRNA is to translate mRNA sequence into amino acid sequence. A tRNA molecule consists of $70 - 80$ nucleotides.

RNA tertiary structure



A hairpin loop from a pre-mRNA. Highlighted are the nucleobases (green) and the ribose-phosphate backbone (blue). Note that this is a single strand of RNA that folds back upon itself.

RNA tertiary structure



Three-dimensional representation of the 50S ribosomal subunit. RNA is in ochre, protein in blue. The active site is in the middle (red).

Prediction of secondary structure: FASTA format

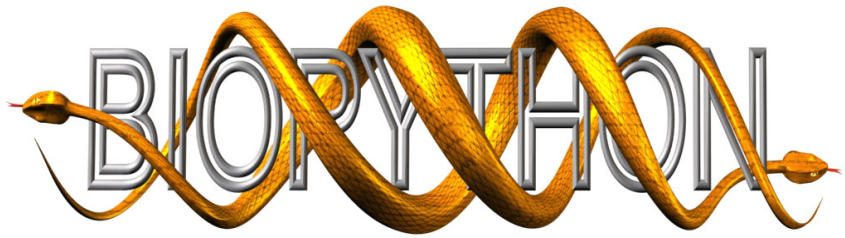
- **FASTA format** is a text-based format for representing either nucleotide sequences or peptide sequences, in which nucleotides or amino acids are represented using single-letter codes.
- The format also allows for sequence names and comments to precede the sequences.
- The format originates from the FASTA software package, but has now become a standard in the field of bioinformatics.
- The simplicity of FASTA format makes it easy to manipulate and parse sequences using text-processing tools and scripting languages like Python, Ruby, and Perl.

```
>MAMseq000312 Euarctos americanus mitochondrial transfer RNA-Pro and transfer RNA-Thr, 3´ ends.
aagactcaaggaagaagcaacagccccactattaacacccaaagctaatgttctatttaaactattccctg
>MAMseq000315 Nasua narica mitochondrial transfer RNA-Pro and transfer RNA-Thr, 3´ ends.
aagacttcaaggaagaagcaacagccacaccatcagcacccaaaactgatattctaactaaactattccttg
>MAMseq000316 Procyon lotor mitochondrial transfer RNA-Pro and transfer RNA-Thr, 3´ ends.
aagacttcaaggaagagacaacccatctcgccatcagcacccaaagctgatattctaactaaactactccttg
>MAMseq000318 Potos flavus mitochondrial transfer RNA-Pro and transfer RNA-Thr, 3´ ends.
aagacttcagggaagaagcaatagctccgccatcagtacccaaaactgacattcttactaaactatccccctg
```

Digression: BioXXX projects

- BioPython: `http://biopython.org/wiki/Main_Page`
- BioPerl: `http://www.bioperl.org/wiki/Main_Page`
- BioJava: `http://biojava.org/wiki/Main_Page`
- BioRuby: `http://bioruby.org`
- Bio (Haskell): `http://biohaskell.org/Libraries/Bio`
- BioCaml: `http://biocaml.org`

Digression: BioPython



- The Biopython Project is an international association of developers of non-commercial Python tools for computational molecular biology, as well as bioinformatics.
- BioPython is one of a number of Bio* projects designed to reduce code duplication.
- `http://biopython.org/wiki/Main_Page`

Digression: BioPython

The main function is `Bio.SeqIO.parse()` which takes a file handle and format name, and returns a `SeqRecord` iterator.

```
from Bio import SeqIO
handle = open("example.fasta", "rU")
for record in SeqIO.parse(handle, "fasta") :
    print record.id
handle.close()
```

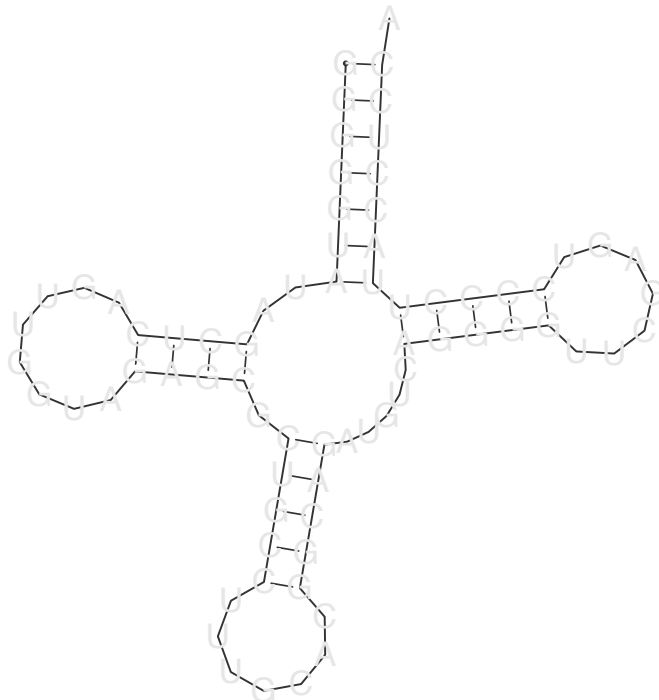
Digression: BioPython

```
from Bio import SeqIO
handle = open("example.fasta", "rU")
records = list(SeqIO.parse(handle, "fasta"))
handle.close()
print records[0].id  #first record
print records[-1].id #last record
```

```
from Bio import SeqIO
handle = open("example.fasta", "rU")
record_dict = SeqIO.to_dict(SeqIO.parse(handle, "fasta"))
handle.close()
print record_dict["gi:12345678"] #use any record ID
```

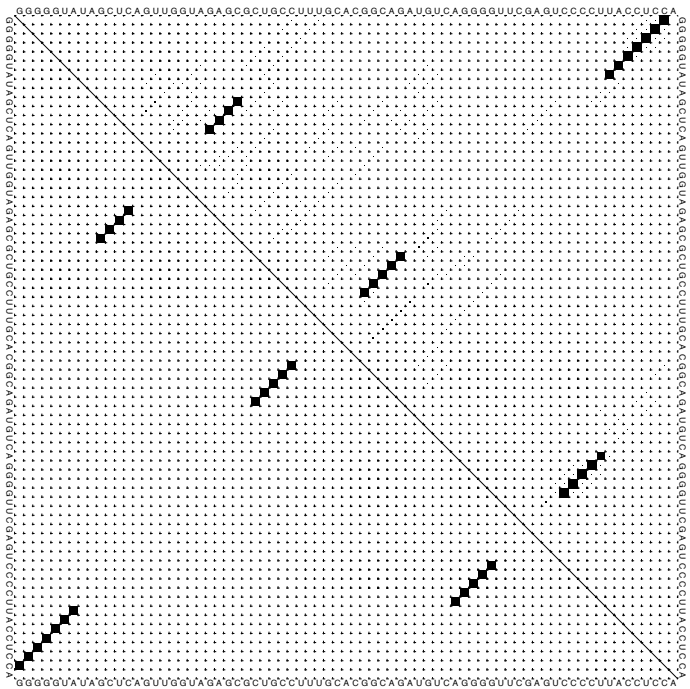
Prediction of secondary structure: RNAfold

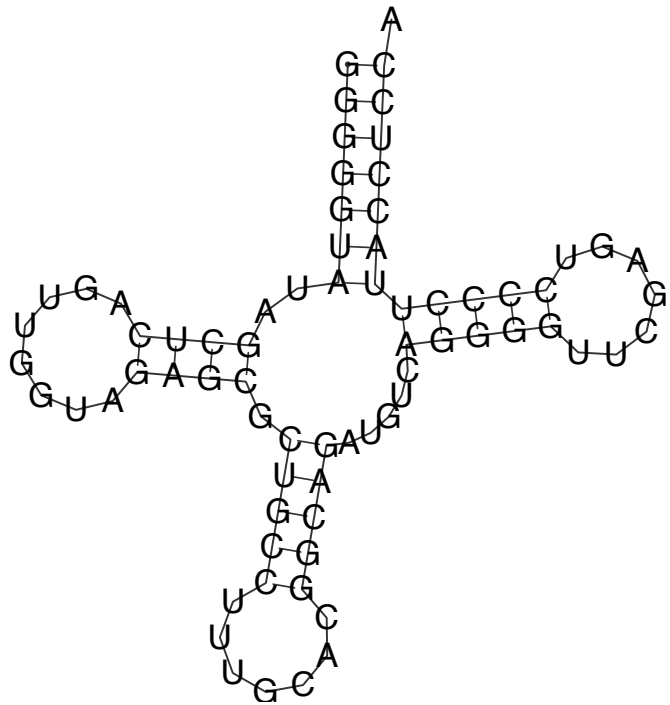
```
barbibul:rna-data$ RNAfold < trna.fa
>AF041468
GGGGGUAUAGCUCAGUUGGUAGAGCGCUGCCUUUGCACGGCAGAUGUCAGGGGUUCGAGUCCCCUUACCUCCA
(((((((..((((........)))).((((.......)))).....((((.......))))))))))))).
barbibul:rna-data$
```

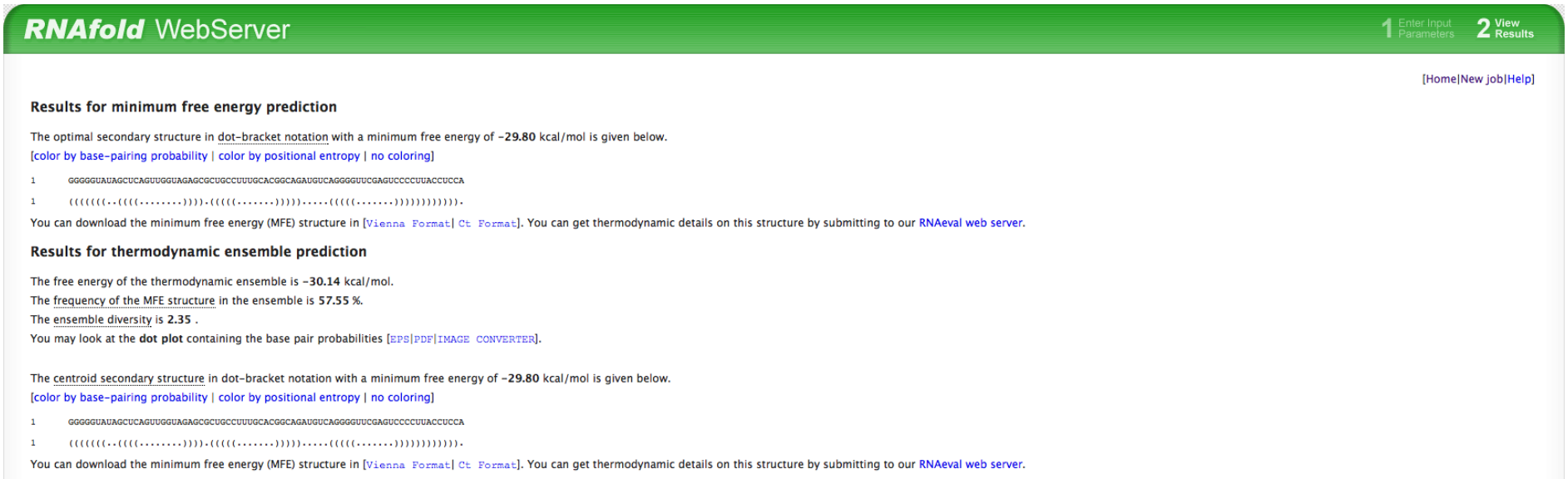
Prediction of secondary structure: RNAfold

```
barbibul:rna-data$ RNAfold < trna.fa
>AF041468
GGGGGUAUAGCUCAGUUGGUAGAGCGCUGCCUUUGCACGGCAGAUGUCAGGGGUUCGAGUCCCCUUACCUCCA
(((((((..((((........)))).(((((.......)))))).....((((.......))))))))))).
barbibul:rna-data$
```





Prediction of secondary structure: RNAfold



RNAfold WebServer

1 Enter Input Parameters 2 View Results

[Home|New job|Help]

Results for minimum free energy prediction

The optimal secondary structure in dot-bracket notation with a minimum free energy of **-29.80** kcal/mol is given below.

[color by base-pairing probability | color by positional entropy | no coloring]

```
1     GGGGGUAUAGCUCAGUUGGUAGAGCGCUGCCUUUGCACGGCAGAUGUCAGGGGUUCGAGUCCCCUUACCUCCA
1     (((((((..((((........)))).(((((.......)))))).....(((((.......)))))))))))).
```

You can download the minimum free energy (MFE) structure in [Vienna Format| Ct Format]. You can get thermodynamic details on this structure by submitting to our RNAeval web server.

Results for thermodynamic ensemble prediction

The free energy of the thermodynamic ensemble is **-30.14** kcal/mol.
The frequency of the MFE structure in the ensemble is **57.55** %.
The ensemble diversity is **2.35** .
You may look at the **dot plot** containing the base pair probabilities [EPS|PDF|IMAGE CONVERTER].

The centroid secondary structure in dot-bracket notation with a minimum free energy of **-29.80** kcal/mol is given below.

[color by base-pairing probability | color by positional entropy | no coloring]

```
1     GGGGGUAUAGCUCAGUUGGUAGAGCGCUGCCUUUGCACGGCAGAUGUCAGGGGUUCGAGUCCCCUUACCUCCA
1     (((((((..((((........)))).(((((.......)))))).....(((((.......)))))))))))).
```

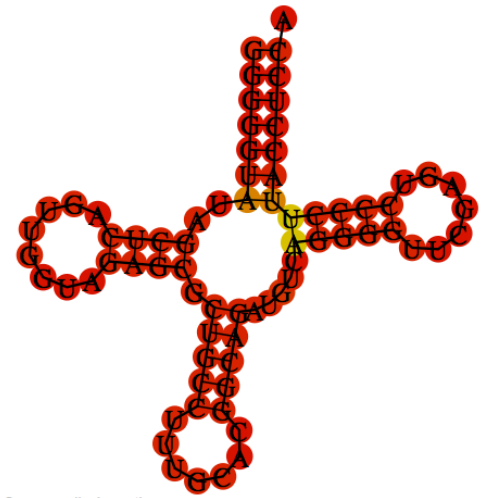
You can download the minimum free energy (MFE) structure in [Vienna Format| Ct Format]. You can get thermodynamic details on this structure by submitting to our RNAeval web server.

`http://rna.tbi.univie.ac.at/cgi-bin/RNAfold.cgi`

Prediction of secondary structure: RNAfold

Graphical output

You may look at the interactive drawing of the MFE structure below. If you do not see the interactive drawing and you are using Internet Explorer, please install the Adobe SVG plugin. **A note on base-pairing probabilities:** The structure below is colored by base-pairing probabilities. For unpaired regions the color denotes the probability of being unpaired.

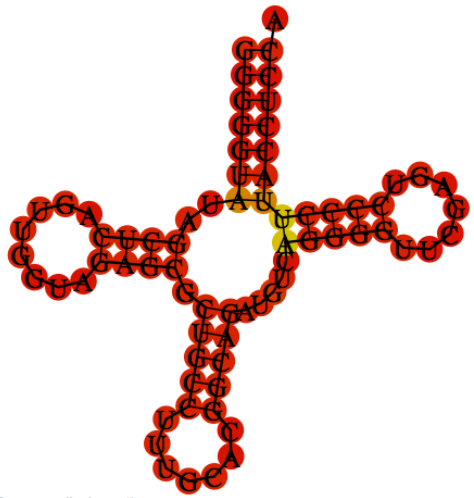


MFE secondary structure

Sequence display options
- (•) Plain Sequence
- ( ) No Sequence

Other display options
- (•) Base-pair probabilities
- ( ) Positional entropy
- ( ) None

0 — 1



Centroid secondary structure

Sequence display options
- (•) Plain Sequence
- ( ) No Sequence

Other display options
- (•) Base-pair probabilities
- ( ) Positional entropy
- ( ) None

0 — 1

Image description	Download options
MFE plain structure drawing	[EPS|PDF|IMAGE CONVERTER]
Centroid plain structure drawing	[EPS|PDF|IMAGE CONVERTER]
MFE structure drawing encoding base-pair probabilities	[EPS|PDF|IMAGE CONVERTER]
Centroid structure drawing encoding base-pair probabilities	[EPS|PDF|IMAGE CONVERTER]
MFE structure drawing encoding positional entropy	[EPS|PDF|IMAGE CONVERTER]
Centroid structure drawing encoding positional entropy	[EPS|PDF|IMAGE CONVERTER]

http://rna.tbi.univie.ac.at/cgi-bin/RNAfold.cgi

Plan

RNA secondary structure prediction

- Many plausible secondary structures can be drawn from a sequence.
- The number increases exponentially with sequence length.
- An RNA only 200 bases long has over 10^{50} possible base-paired structures.
- We must distinguish the biologically correct structure from all the incorrect.structures

Base pair maximisation: the Nussinov folding algorithm

- One (**naive**) approach is to find the structure with the most base pairs.
- Nussinov introduced an efficient dynamic programming algorithm for this problem.
- Although the criterion is too simplistic to give accurate structure predictions, the algorithm is instructive because the mechanics of the Nussinov folding algorithm are the same as those in the more sophisticated energy minimisation folding algorithms (and of probabilistic SCFG-based algorithms).

RNA secondary structure

Definition

Let $u \in \{A, C, G, U\}^*$ be a sequence. An **RNA-structure** over u is a set of pairs

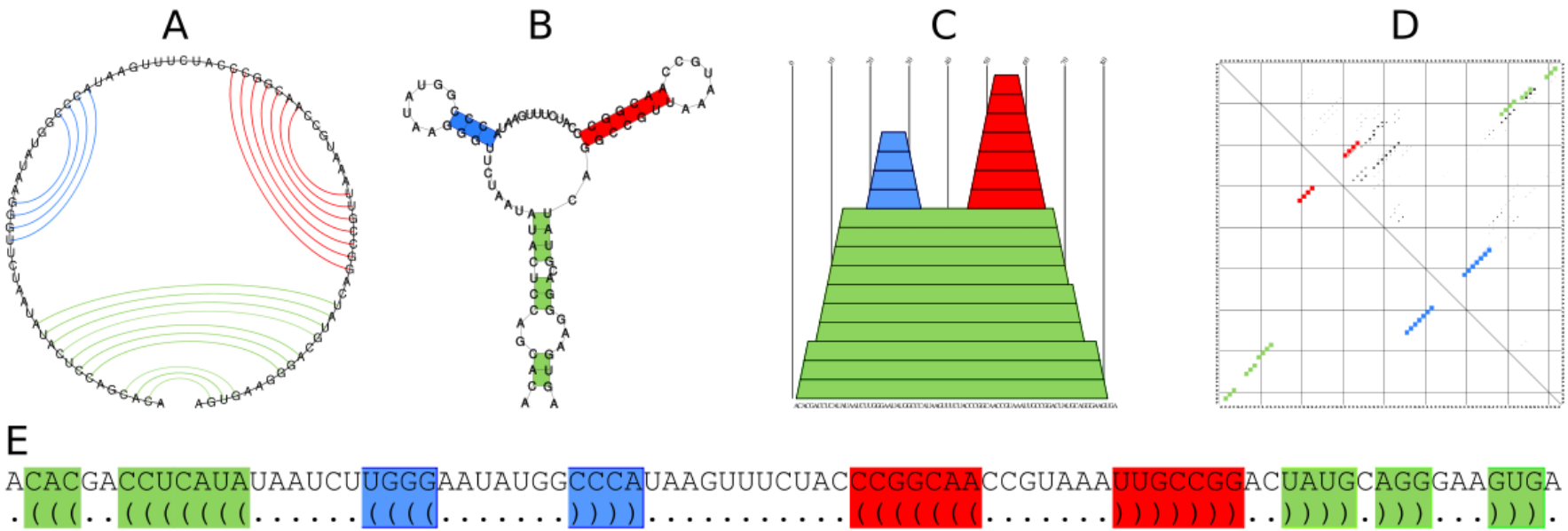
$$P = \{(i, j) : i < j, u[i] \text{ and } u[j] \text{ form a a WC or non-standard pair}\}$$

with the property that the associated graph has degree at most 1 (*i.e.*, every base can have at most one bond).

Remark

$$\begin{aligned} \forall (i,j), \quad & (i,j) \in P \Rightarrow \forall i', (i',j) \notin P \\ \forall (i,j), \quad & (i,j) \in P \Rightarrow \forall j', (i,j') \notin P \end{aligned}$$

RNA secondary structure



Purine riboswitch (Rfam RF00167)

The Nussinov folding algorithm

- **Idea (biological)**: Stacked base pairs of helical regions are considered to stabilize an RNA molecule.

  Therefore, the goal is to maximize the number of base pairs.

- **Idea (algorithmic)**: The optimal structure $S[i,j]$ on a subsequence $u[i,j]$ can only be formed by two distinct ways from a shorter subsequence $u[i+1,j]$:
  1. Base i is unpaired, followed by an arbitrary shorter structure.
  2. Base i is paired with some partner base k requiring the computation of two independent substructures: the structure enclosed by the bp and the remaining structure behind the pair.

The Nussinov folding algorithm

- Initialisation

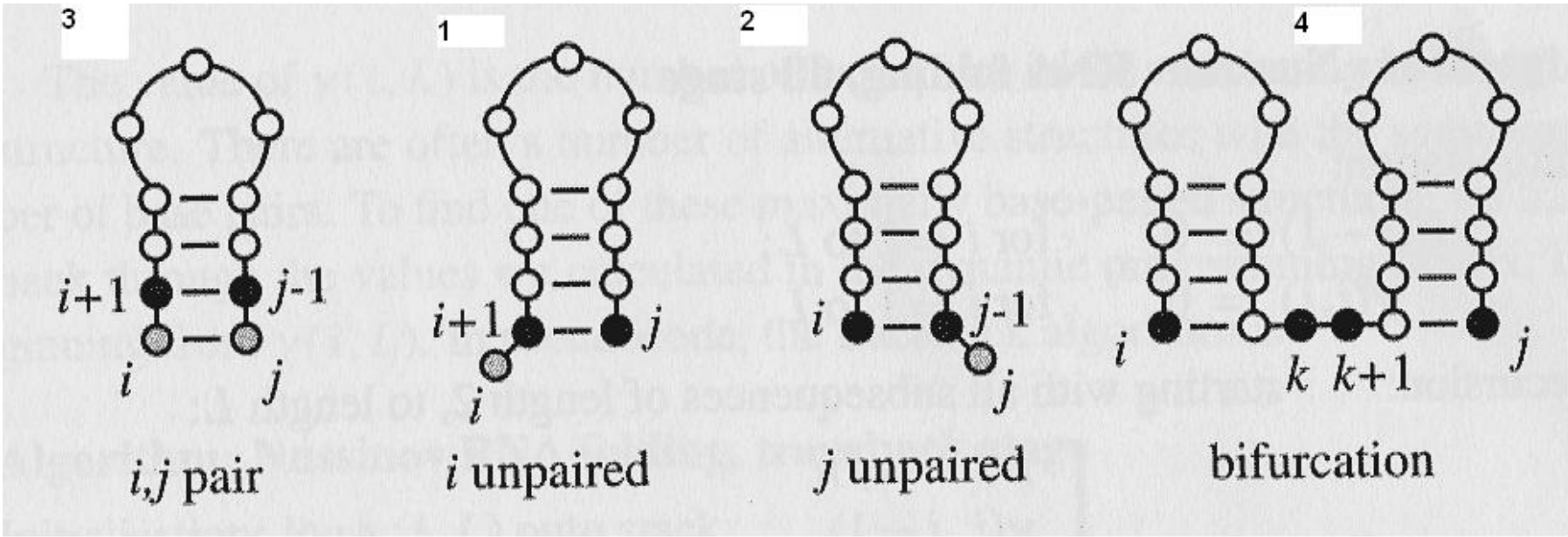
$$\gamma(i, i-1) = 0 \qquad 2 \leq i \leq n$$
$$\gamma(i, i) = 0 \qquad 1 \leq i \leq n$$

- Recursion

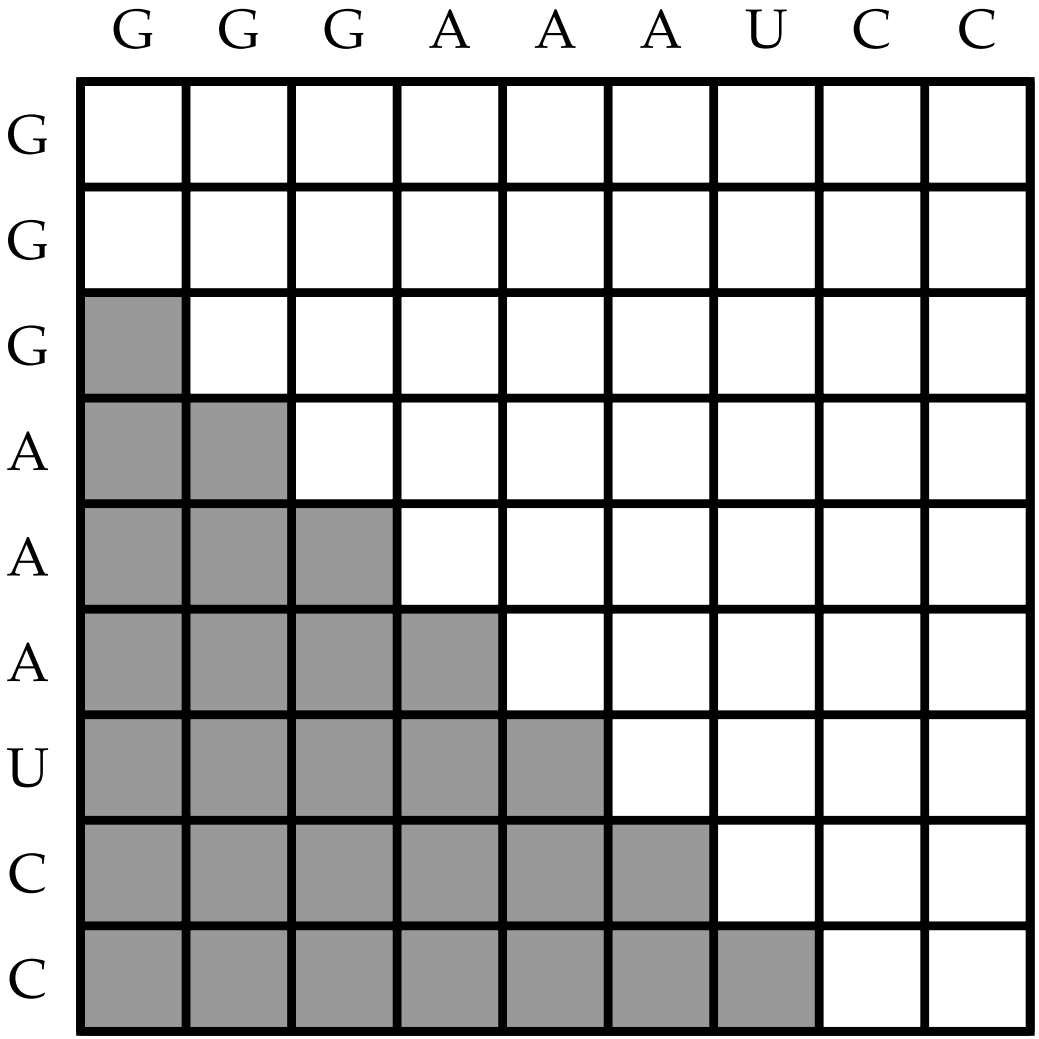
$$\gamma(i,j) = \max \begin{cases} \gamma(i+1,j) \\ \gamma(i,j-1) \\ \gamma(i+1,j-1) + \alpha(i,j) \\ \max_{i<k<j} \{\gamma(i,k) + \gamma(k+1,j)\} \end{cases}$$

- $O(n^3)$ **time** and $O(n^2)$ **space**.

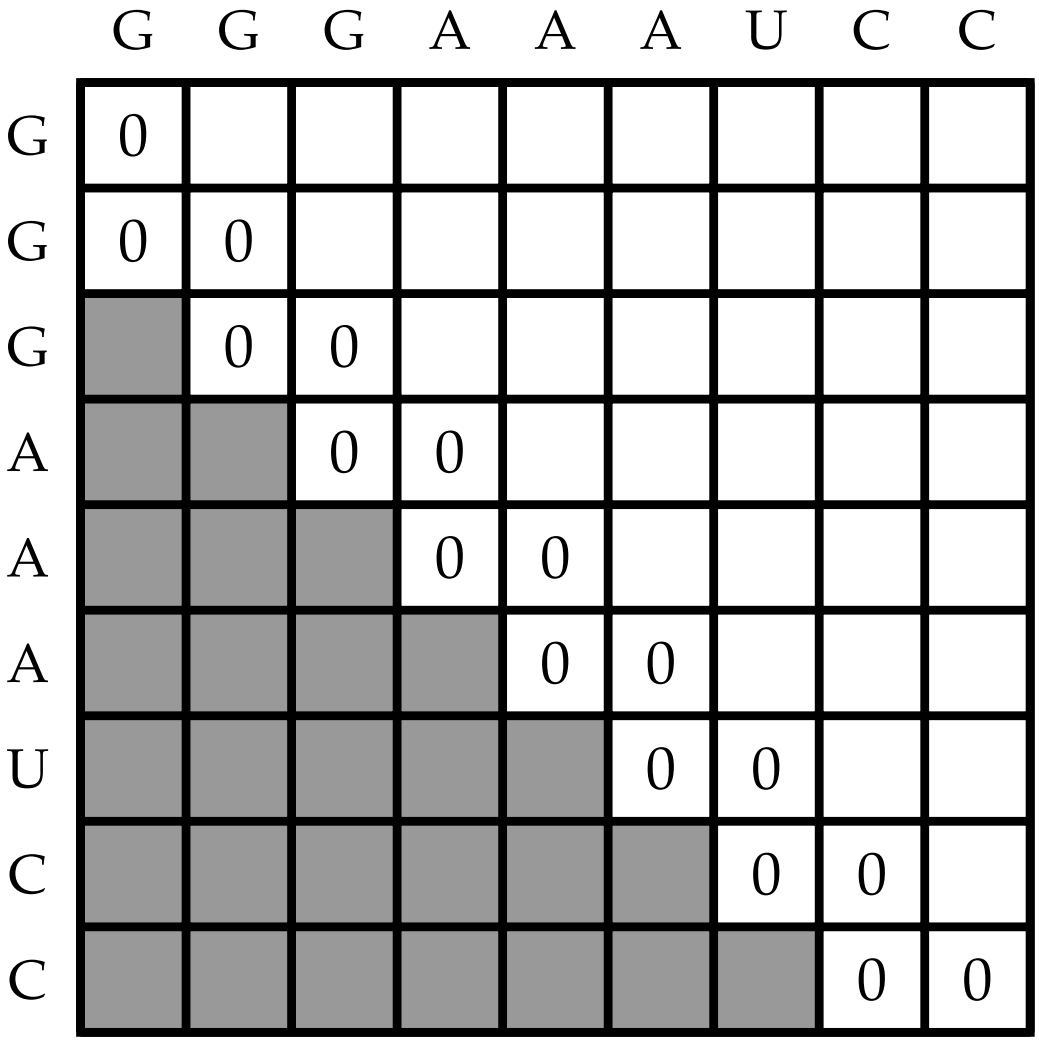
The Nussinov folding algorithm



The Nussinov folding algorithm: Example



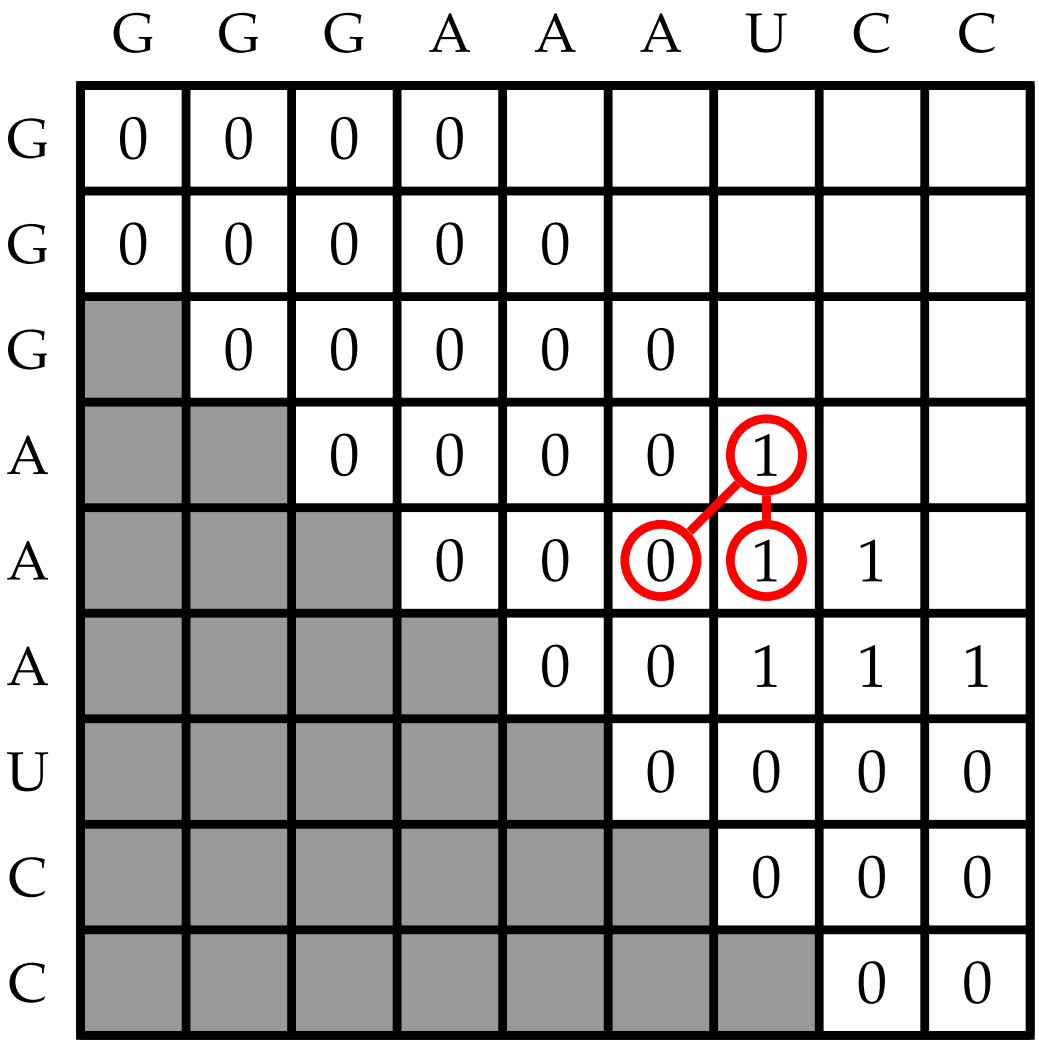
The Nussinov folding algorithm: Example



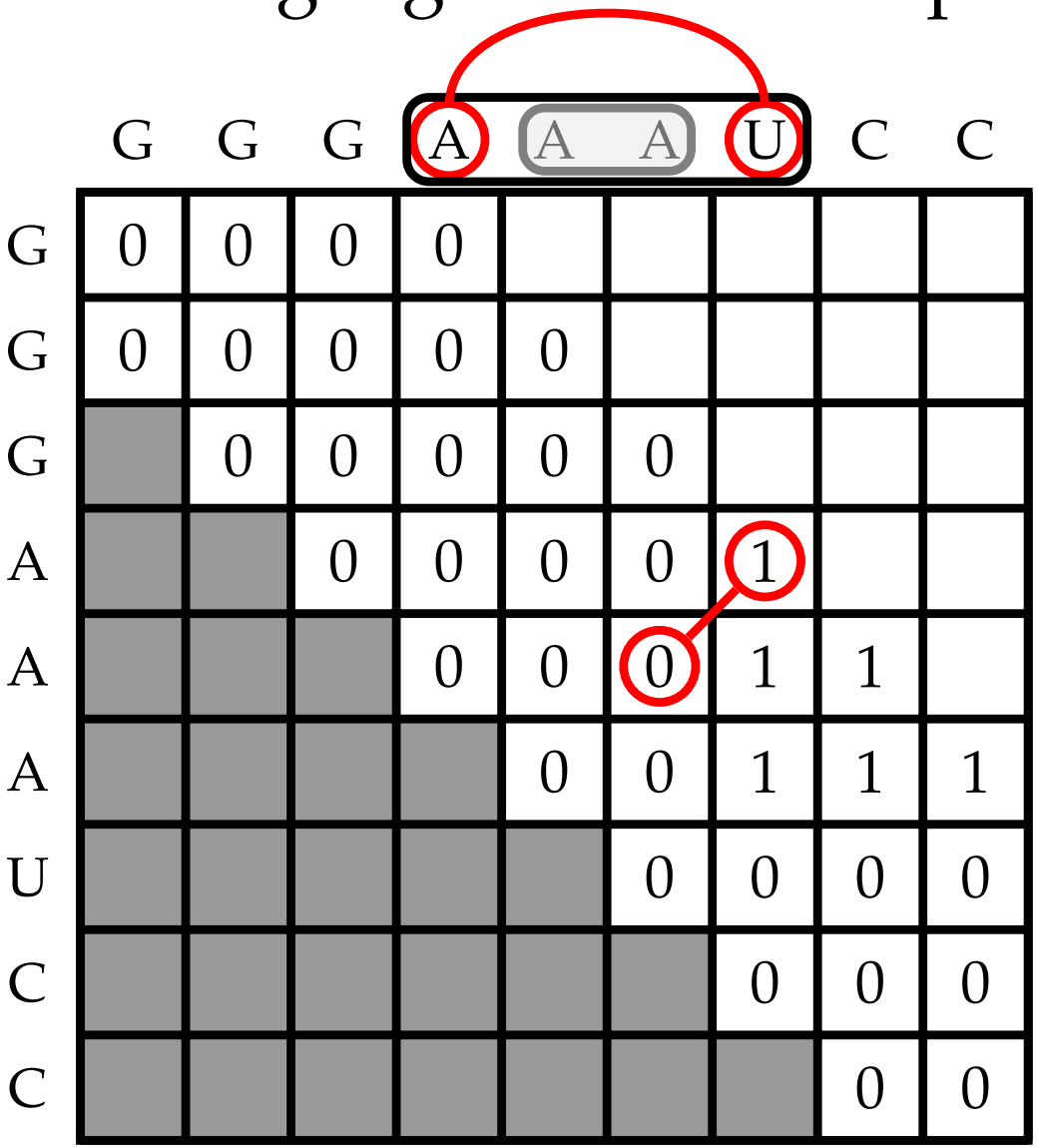
The Nussinov folding algorithm: Example

	G	G	G	A	A	A	U	C	C
G	0	0	0	0					
G	0	0	0	0	0				
G		0	0	0	0	0			
A			0	0	0	0	1		
A				0	0	0	1	1	
A					0	0	1	1	1
U						0	0	0	0
C							0	0	0
C								0	0

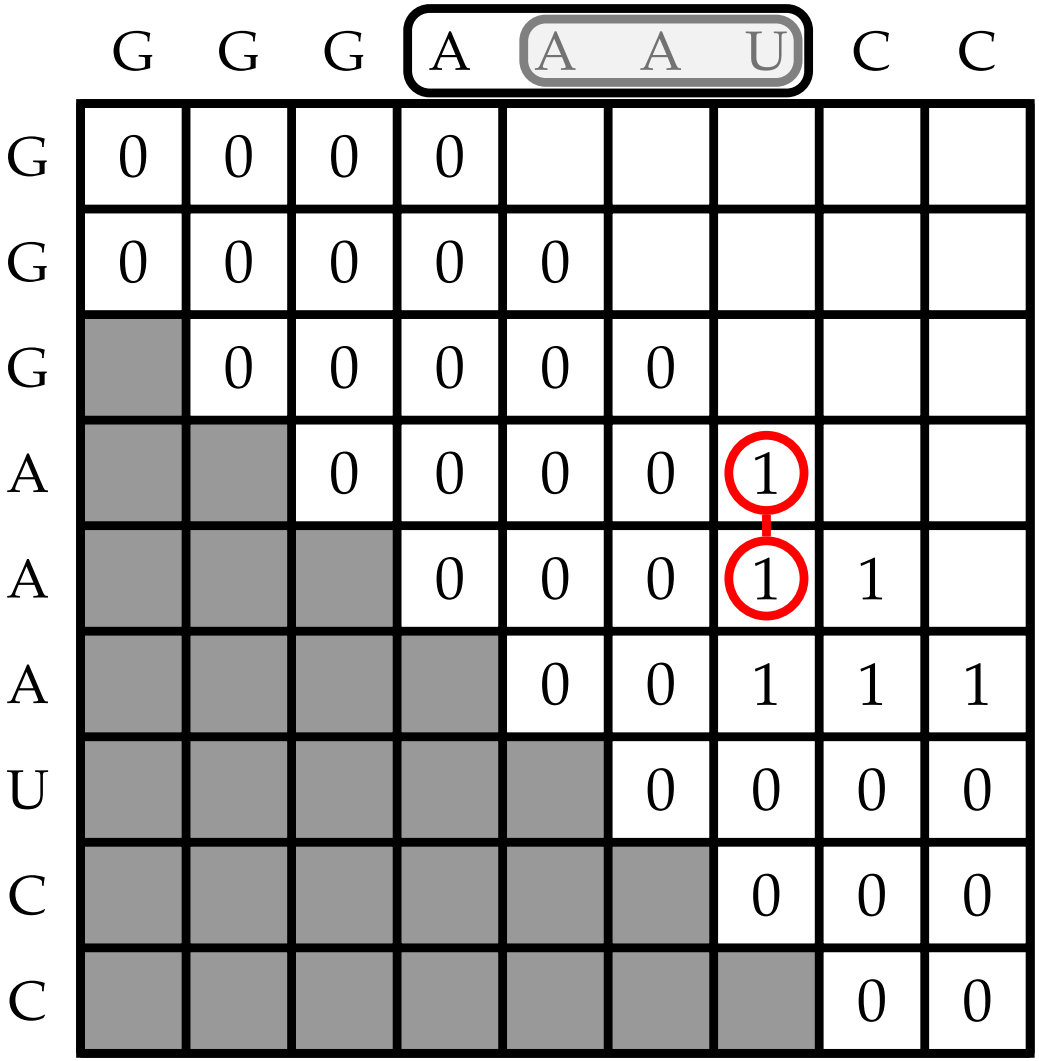
The Nussinov folding algorithm: Example



The Nussinov folding algorithm: Example



The Nussinov folding algorithm: Example



The Nussinov folding algorithm: Example

	G	G	G	A	A	A	U	C	C
G	0	0	0	0	0	0	1	2	3
G	0	0	0	0	0	0	1	2	3
G		0	0	0	0	0	1	2	2
A			0	0	0	0	1	1	1
A				0	0	0	1	1	1
A					0	0	1	1	1
U						0	0	0	0
C							0	0	0
C								0	0

The Nussinov folding algorithm

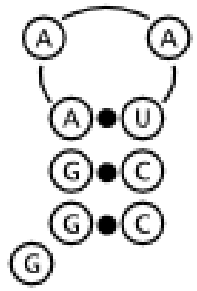
- The value $\gamma(1, n)$ is the number of base pairs in the maximally base-paired structure.
- There are often a number of alternatives structures with the same number of base-pairs.
- To find one of these maximally base-paired structures, we **trace back** through the values we calculated in the dynamic programming matrix, beginning from $\gamma(1, n)$.

The Nussinov folding algorithm: Traceback stage

```
1  S ← emptyStack
2  push(S, (1, n))
3  while S ≠ ∅ do
4      (i, j) ← pop(S)
5      if i < j then
6          if γ(i + 1, j) = γ(i, j) then
7              push(S, (i + 1, j))
8          else if γ(i, j − 1) = γ(i, j) then
9              push(S, (i, j − 1))
10         else if γ(i + 1, j − 1) + α(i, j) = γ(i, j) then
11             record pair (i, j)
12             push(S, (i + 1, j − 1))
13         else
14             for j + 1 ≤ k ≤ j − 1 do
15                 if γ(i, k) + γ(k + 1, j) = γ(i, j) then
16                     push(S, (k + 1, j))
17                     push(S, (i, k))
18                     break
```

The Nussinov folding algorithm: Traceback stage

	G	G	G	A	A	A	U	C	C
G	0	0	0	0	0	0	1	2	3
G	0	0	0	0	0	0	1	2	3
G		0	0	0	0	0	1	2	2
A			0	0	0	0	1	1	1
A				0	0	0	1	1	1
A					0	0	1	1	1
U						0	0	0	0
C							0	0	0
C								0	0



The Nussinov folding algorithm: Traceback stage

- Linear time and space.
- In the canonical implementation of the traceback step, whenever there are multiple structures that are equivalent in terms of number of base-pairs the first structure that works is chosen because the algorithm does not care about anything besides the number of base-pairs, so any structure with the same number of base pairs as the optimal one will do.
- However, this ignores important information that can lead it to choose an unstable structure over a more stable one.

The Nussinov folding algorithm: Traceback stage

Possible improvements:

- Incorporates the following assumptions:
  - Longer stems (consecutive base pairs) are more stable than shorter stems,
  - A single loop or bulge is more stable than one split in two by a base pair in the middle
- Report all optimal solutions.

The Nussinov folding algorithm: Drawbacks

The Nussinov folding algorithm does not determine biological relevant structures since:

- There are many (all!?) possibilities to form base pairs.
- Stackings of base pairs are not considered.
- The size of internals loops are not considered.

An SCFG version of te Nussinov folding algorithm

- A single non-terminal S;
- 14 production rules with associated probability parameters.

$$
\begin{array}{ll}
S \to aS \mid cS \mid gS \mid uS & (i \text{ unpaired}) \\
S \to Sa \mid Sc \mid Sg \mid Su & (j \text{ unpaired}) \\
S \to aSu \mid cSg \mid gSc \mid uSa & (i \text{ and } j \text{ paired}) \\
S \to SS & (\text{bifurcation}) \\
S \to \epsilon & (\text{termination})
\end{array}
$$

An SCFG version of te Nussinov folding algorithm

- Assume that the probabiliy parameters are known.
- The maximum probability parse of a sequence with this SCFG is an assignment of sequence positions to productions.
- Because the productions correspond to secondary structure elements (base pairs and single-stranded bases), the maximum probability parse is equivalent to the maximum probability secondary structure.
- If base pair productions have relatively high probability, the SCFG will favour parses which tend to maximise the number of base pairs in the structure.

RNA structure prediction: MFE-folding

- More realistic: thermodynamics and statistical mechanics.
- Stability of an RNA secondary structure coincides with thermodynamic stability.
- Quantified as the amount of free energy released/used by forming base pairs.

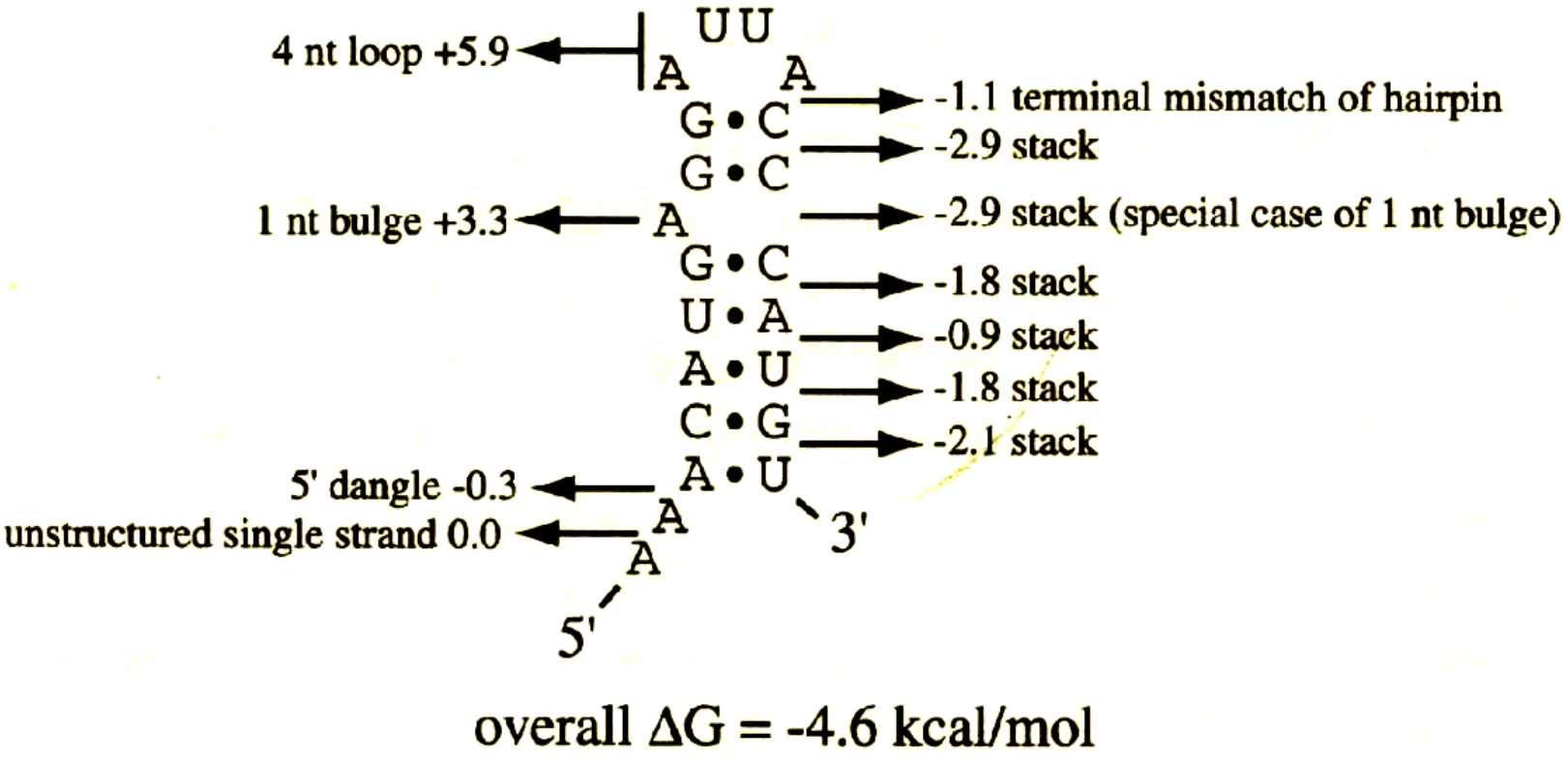
RNA structure prediction: MFE-folding

RNA molecules basically exist in a distribution of structures rather than a single ground-state conformation.

- "Most likely" conformation: structure exhibiting minimum of free energy (MFE).
- Energy contributions of different loop types have been measured.
- Since free energies are additive, a more sophisticated model, the standard energy model for RNA secondary structures, can be proposed.
- Based on loop decomposition, the total energy E of a structure S can be computed as the sum over the energy contributions of each constituent loop ℓ:

$$E(S) = \sum_{\ell \in S} E(\ell)$$

MFE folding: Example



Structural elements: Formal definition

Definition

Secondary structure elements Let u be a fixed sequence. Further, let S be an RNA secondary structure for u.

- A base pair $(i,j) \in S$ is a **hairpin loop** if

$$\forall i < i' \leq j' < i, \quad (i',j') \notin S$$

- A base pair $(i,j) \in S$ is called **stacking** if $(i+1, j-1) \in P$.
- Two base pairs $(i,j) \in S$ and $(i',j') \in S$ form an **internal loop** (i,j,i',j') if
  - $i < i' < j' < j$
  - $(i' - i) + (j - j') > 2$ (no stack)
  - there is no base pair (k,l) between (i,j) and (i',j')

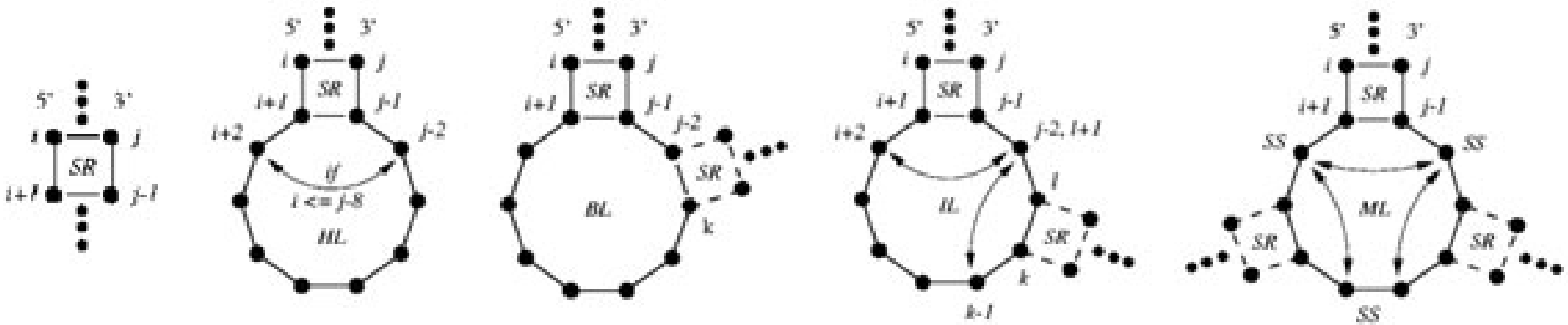
Structural elements: Formal definition

Definition

Secondary structure elements (ctd)

- An internal loop (i, j, i', j') is called **left** (resp. **right**) **bulge**, if $j = j' + 1$ (resp. $i' = i + 1$).
- A k-**multiloop** consists of k base pairs $(i_1, j_1), (i_2, j_2), \ldots, (i_k, j_k) \in S$ and a closing base pair $(i, j) \in S$ with the property that
  - $i < i_1 < j_1 < i_2 < j_2 < \ldots < i_k < j_k < j$
  - $i + 1, \ldots, i_1 - 1, j_1 + 1, \ldots, i_2 - 1, \ldots,$ $j_k - 1 + 1, \ldots, i_k - 1, j_k + 1, \ldots, j - 1$ are unpaired in S
  - $(i_1, j_1), (i_2, j_2), \ldots, (i_k, j_k)$ close the inner base pairs of the multiloop.

Structural elements: Formal definition

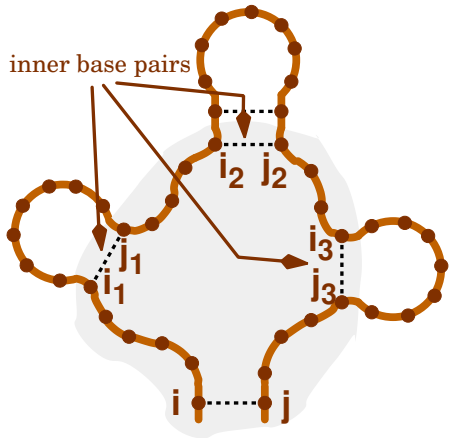


(a) Stack (b) Hairpin (c) Bulge (d) Internal (e) Multiple

Structural elements: k-multiloop

Remarks

- Usually hairpin loops have minimal loop size of $m = 3$ (*i.e.*, for all $(i,j) \in S, i < j - 3$).
- Each secondary structure element is defined uniquely by its closing basepair.
- For any basepair (i,j) we denote the corresponding secondary structure element with $sec(i,j)$.



Energy of secondary structural elements

Definition (Energy contribution of loops)

Energy contributions of the various structure elements:

hairpin loop (i,j) :	$\mathrm{eH}(i,j)$
stacking (i,j) :	$\mathrm{eS}(i,j)$
internal loop (i,j,i,j') :	$\mathrm{eL}(i,j,i',j')$
multiloop $(i,j,i_1,j_2,\ldots,i_k,j_k)$:	$\mathrm{eM}(i,j,i_1,j_2,\ldots,i_k,j_k)$

Remarks

- General multi loop contribution will be too expensive in prediction: exponential explosion!
- Use a simplified contribution scheme.

Energy of secondary structural elements

Definition (Simplified energy contribution of multiloops)

$$\mathrm{eM}(i,j,k,k') = a + bk + ck'$$

where

- a, b and c are weights (a is the energy contribution for closing of loop),
- k is the number of inner base pairs, and
- k' is the number of unpaired bases within loop.

MFE folding

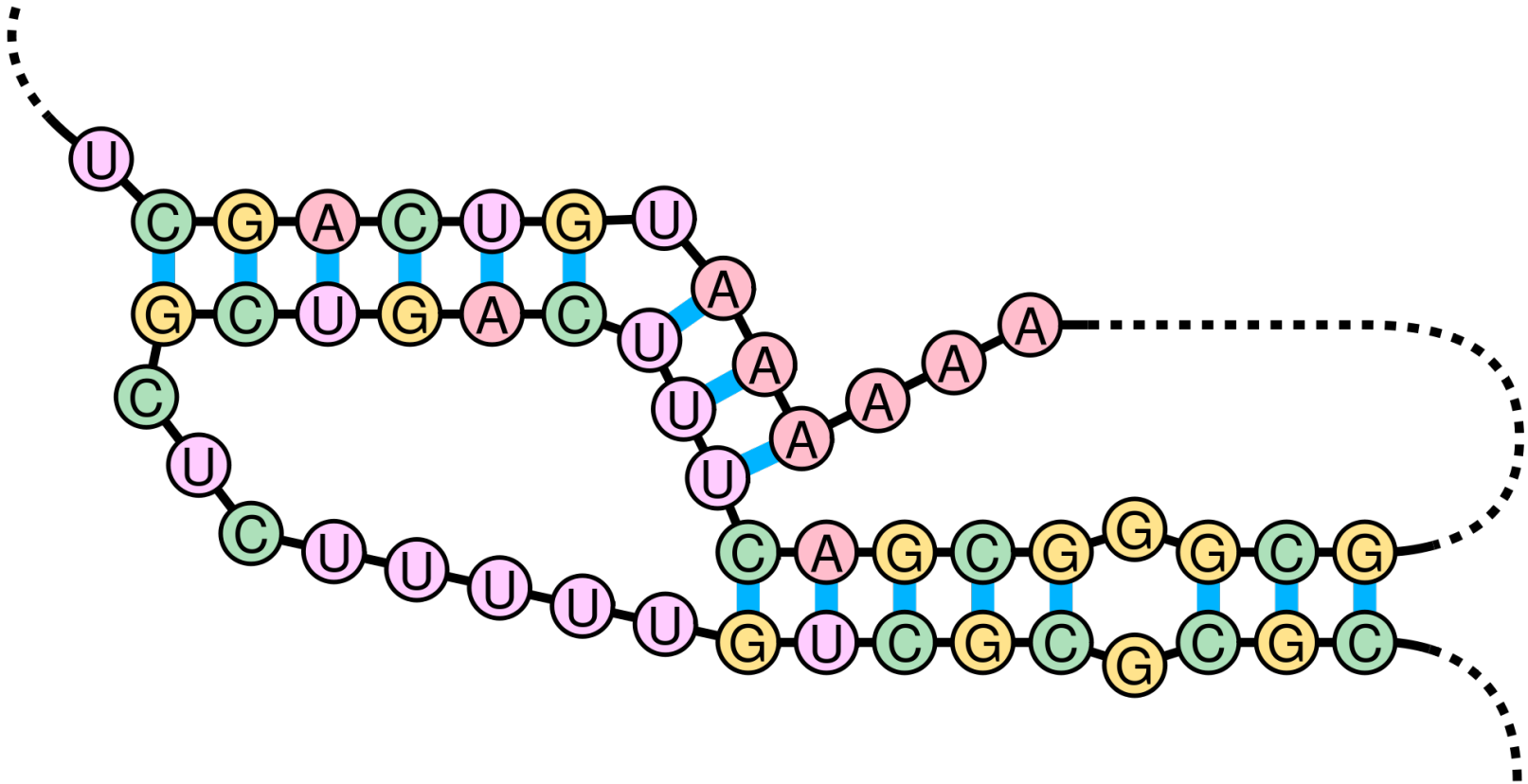
- The complexity of the dynamic programming algorithm is $O(n^4)$ time and $O(n^2)$ space.
- Using a trick, the time complexity can be reduced to $O(n^3)$.
- We assume traceback is done analogously to Nussinov-Traceback. Same reduced complexity. Only extension: trace through three matrices, (*i.e.*, keep track of matrix).

Plan

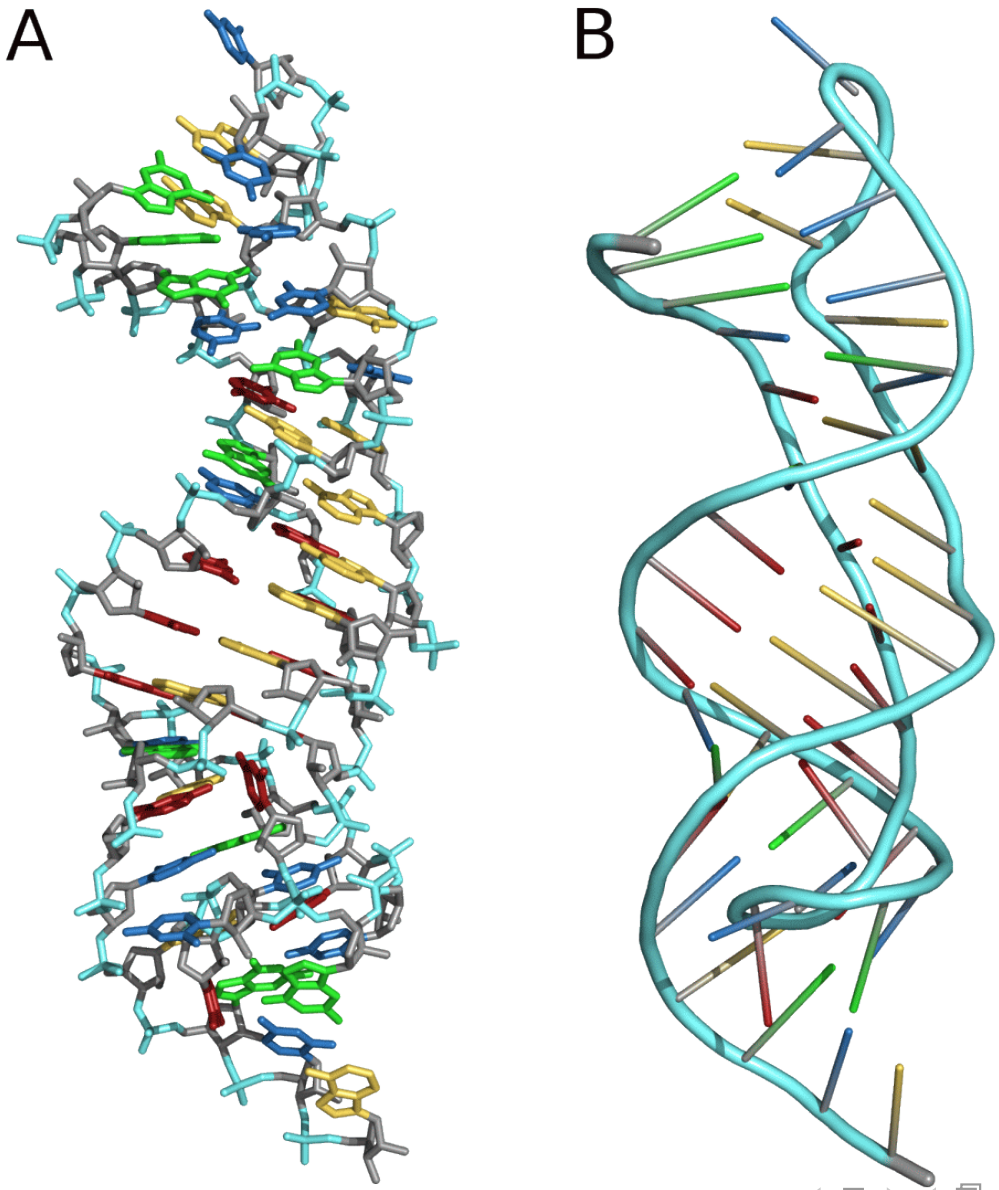
RNA pseudoknots

- A **pseudoknot** is a nucleic acid secondary structure containing at least two stem-loop structures in which half of one stem is intercalated between the two halves of another stem.
- The pseudoknot was first recognized in the turnip yellow mosaic virus in 1982.
- Pseudoknots fold into knot-shaped three-dimensional conformations but are not true topological knots.

RNA pseudoknots



RNA pseudoknots



RNA pseudoknots: Prediction and identification

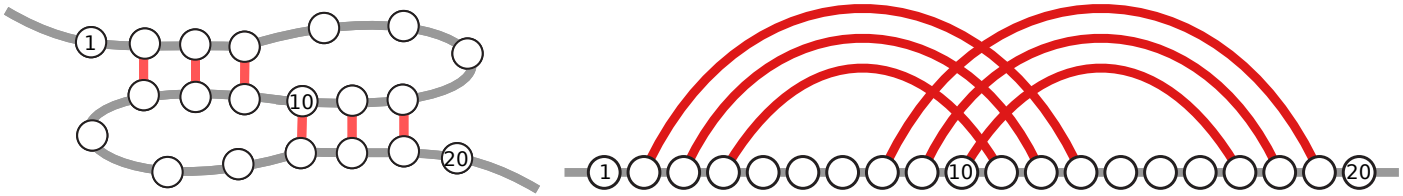
- The structural configuration of pseudoknots does not lend itself well to bio-computational detection due to its context-sensitivity or ”overlapping” nature.
- The presence of pseudoknots in RNA sequences is more difficult to predict by the standard method of dynamic programming, which use a recursive scoring system to identify paired stems and consequently, most cannot detect non-nested base pairs.
- Popular secondary structure prediction methods do not predict pseudoknot structures present in a query sequence.
- It is possible to identify a limited class of pseudoknots using dynamic programming, but these methods are not exhaustive and scale worse as a function of sequence length than non-pseudoknotted algorithms.
- The general problem of predicting lowest free energy structures with pseudoknots has been shown to be **NP**-complete.

RNA pseudoknots: Biological significance

- Several important biological processes rely on RNA molecules that form pseudoknots, which are often RNAs with extensive tertiary structure.
- For example, the pseudoknot region of RNase P is one of the most conserved elements in all of evolution.
- The telomerase RNA component contains a pseudoknot that is critical for activity.
- Several viruses use a pseudoknot structure to form a tRNA-like motif to infiltrate the host cell.

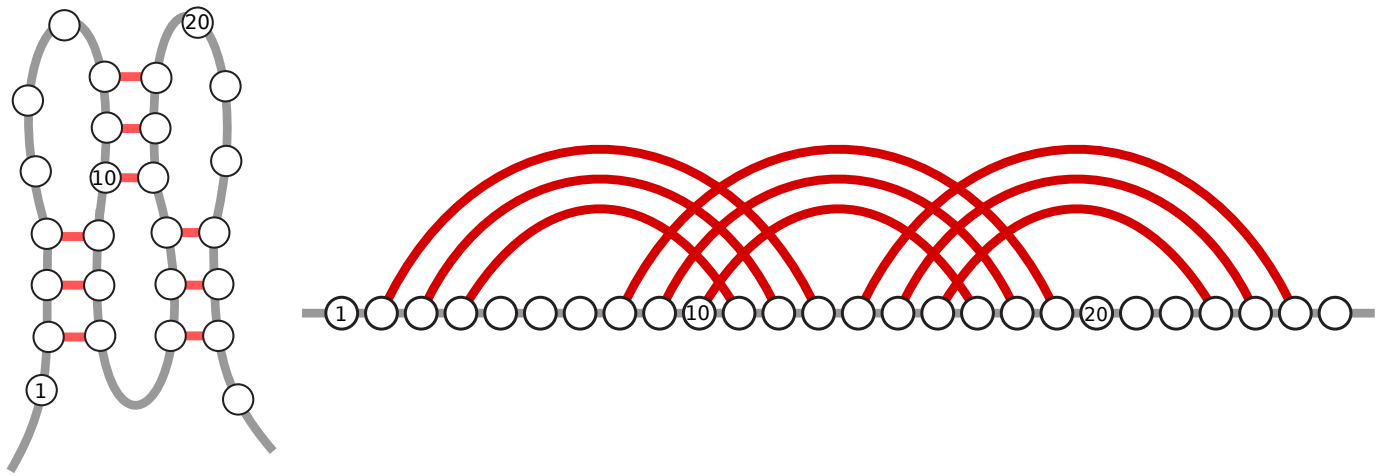
RNA pseudoknot type

- Simple, H-type



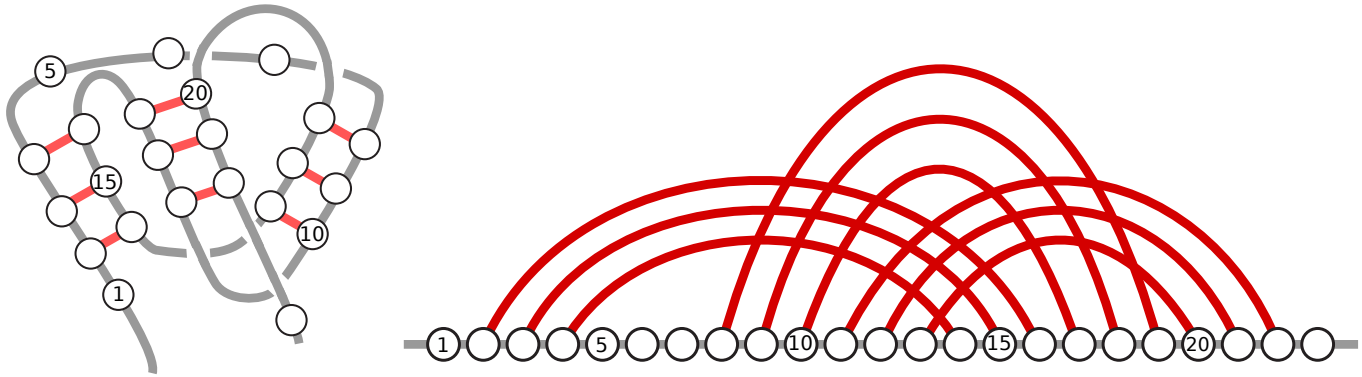
RNA pseudoknot type

- Kissing hairpin



RNA pseudoknot type

- Three-knot



RNA pseudoknot predition

	R&G	A/U	L&P	D&P	CCJ	R&E
Time	$O(n^4)$	$O(n^4)/O(n^5)$	$O(n^5)$	$O(n^5)$	$O(n^5)$	$O(n^6)$
Space	$O(n^2)$	$O(n^3)/O(n^3)$	$O(n^3)$	$O(n^4)$	$O(n^4)$	$O(n^4)$

Nearest Neighbor Energy Model

For a secondary structure S

- the number of **base pairs stackings** is

$$\mathrm{BPS}(S) = |\{(i,j) \in S : (i+1, j-1) \in S\}|$$

- the number of **stacking base pairs** is

$$\mathrm{SBP}(S) = |\{(i,j) \in S : (i+1, j-1) \in S \text{ or } (i-1, j+1) \in S\}|$$

Without pseudoknots

- Maximizing the number of base pairs is $O(n^3)$ **time** and $O(n^2)$ **space**:

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

- To maximize BPS or SBP, dynamic programming can be extended.

With arbitrary pseudoknots

- Maximizing the number of base pairs is $O(n^{2/5})$ **time**.

  The problem reduces to finding a maximum matching in a graph (solvable in $O(m\sqrt{n})$ time)

- To maximize BPS or SBP, matching becomes inadequate, and dynamic programming cannot be extended.

Complexity BPS(S)

Theorem

Given a sequence u and a positive integer K, it is **NP***-complete to decide whether there exists a structure S that is legal under the canonical RNA folding model and with* $\mathrm{BPS}(S) \geq k$.

BIN PACKING

Definition (The BIN PACKING problem)

- **Input**: k items of sizes $a_1, a_2, \ldots, a_k$ and B bins each with capacity $\mathcal{C}$.
- **Question**: Decide whether the items fit into the bins.

Remarks

- Strongly **NP**-complete.
- A straightforward greedy algorithm achieves an approximation factor of 2.
- Does not have a polynomial-time approximation scheme (**PTAS**) unless $\mathbf{P} = \mathbf{NP}$.
- for any $0 < \epsilon \leq 1$, it is possible to find a solution using at most $(1+\epsilon)\mathbf{opt} + 1$ bins in polynomial time (asymptotic PTAS).

Strongly **NP**-complete

Definition

A problem is said to be **strongly NP-complete** (or **NP-complete in the strong sense**), if it remains so even when all of its numerical parameters are bounded by a polynomial in the length of the input.

- Normally numerical parameters to a problem are given in binary notation, so a problem of input size n might contain parameters whose size is exponential in n.
- If we redefine the problem to have the parameters given in unary notation, then the parameters must be bounded by the input size.
- Thus strong **NP**-completeness or **NP**-hardness may also be defined as the **NP**-completeness or **NP**-hardness of this unary version of the problem.

Strongly **NP**-complete

Definition

A problem is said to be **strongly NP-complete** (or **NP-complete in the strong sense**), if it remains so even when all of its numerical parameters are bounded by a polynomial in the length of the input.

- BIN PACKING is strongly **NP**-complete while the $0-1$ KNAPSACK problem is only weakly **NP**-complete.

  Thus the version of BIN PACKING *where the object and bin sizes are integers bounded by a polynomial remains* **NP***-complete, while the corresponding version of the* KNAPSACK *problem can be solved in polynomial time by dynamic programming.*

- Any strongly **NP**-hard optimization problem with a polynomially bounded objective function cannot have an **FPTAS** unless $\mathbf{P} = \mathbf{NP}$.

Complexity BPS(S)

Theorem

Given a sequence u and a positive integer K, it is **NP***-complete to decide whether there exists a structure S that is legal under the canonical RNA folding model and with* $\mathrm{BPS}(S) \geq k$.

Proof.

Construction:

$$u = C^{a_1} A C^{a_2} A \ldots A C^{a_k} A A (A G^{C})^{B}$$

$$K = \sum_{1 \leq i \leq k} a_i - k$$

As A's can only form base pairs with U's in the canonical folding model, all base pairs in a legal structure for u wil be $G \cdot C$.

□

Complexity BPS(S)

Theorem

Given a sequence u and a positive integer K, it is **NP***-complete to decide whether there exists a structure S that is legal under the canonical RNA folding model and with* $\mathrm{BPS}(S) \geq k$.

Proof.

- $|u| = \sum_{1 \leq i \leq k} a_i + B\mathcal{C} + B + k + 1$.
- Since the BIN PACKING problem is **strongly NP**-complete we can assume that the $B, \mathcal{C}, a_1, a_2, \ldots, a_k$ are all polynomially bounded by the size of the originally BIN PACKING instance.

□

Complexity BPS(S)

Theorem

Given a sequence $u \in \{0,1\}^*$ and a positive integer K, it is **NP***-complete to decide whether there exists a structure S that is legal under the general RNA folding model with $\mathbb{B} = \{(0,1),(1,0)\}$ with* $\mathrm{BPS}(S) \geq k$.

Proof.

- Reduction from BIN PACKING:
  - $3 \leq a_i \leq C$ for $1 \leq i \leq k$, and
  - $2 \leq B \leq k$.
- $u = 0^{a_1}110^{a_2}11\ldots110^{a_k}(01^C)^B$.
- $K = \sum_{1\leq i\leq k} a_i - k + B$.

□

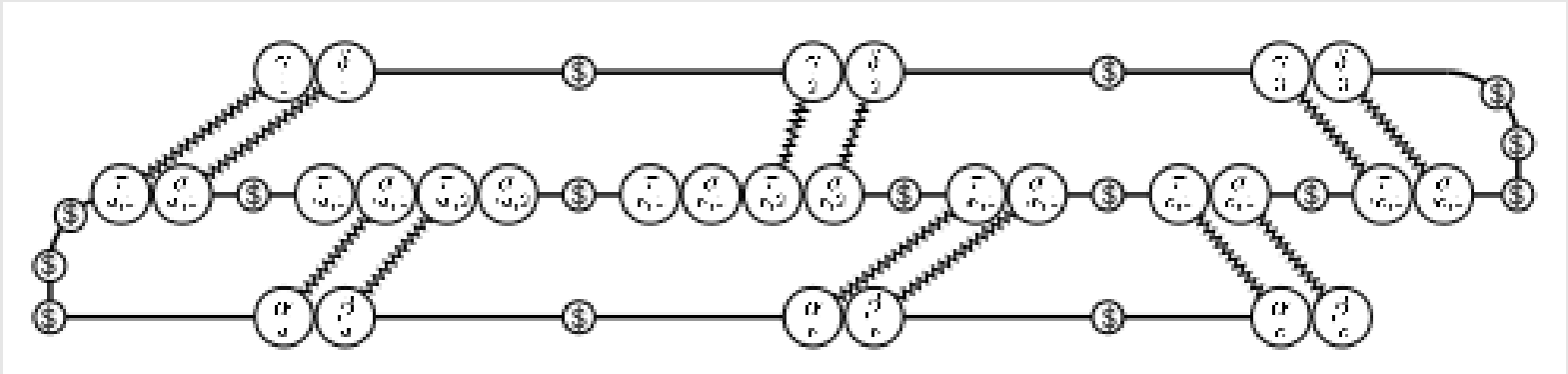
Complexity SBP(S)

Theorem

Given an alphabet Σ, a set of legal base pairs $\mathbb{B} \subseteq \Sigma \times \Sigma$, a sequence $u \in \Sigma^*$ and a positive integer K, it is **NP***-complete to decide whether there exists a structure S that is legal under the general RNA folding model with* $\mathrm{SBP}(S) \geq k$.

Proof.

- Reduction from RESTRICTED SATISFIABILITY:



□

2-intervals

Definition (2-intervals)

- A 2-**interval** $D = (I, J)$ consists of two disjoint (closed) intervals I and J such that $I < J$ (*i.e.*, I is completely on the left of J).
- Two 2-intervals $D_1 = (I_1, J_1)$ and $D_2 = (I_2, J_2)$ are **disjoint** if the four intervals I_1, J_1, I_2 and J_2 are pairwise disjoint.

Definition (Relations between disjoint 2-intervals)

Let $D_1 = (I_1, J_1)$ and $D_2 = (I_2, J_2)$ be two disjoint 2-intervals. We have the following relations:

- $D_1 < D_2$ if $I_1 < J_1 < I_2 < J_2$.
- $D_1 \sqsubset D_2$ if $I_2 < I_1 < J_1 < J_2$.
- $D_1 \between D_2$ if $I_1 < I_2 < J_1 < J_2$.

2-intervals: Models

Definition (Models)

- A **model** is a non-empty subset of $\{<, \sqsubset, \between\}$.
- A set of 2-intervals $\mathcal{D}$ is $\mathcal{R}$-structured if any two distinct 2-intervals in $\mathcal{D}$ is R-comparable for some $R \in \mathcal{R}$.

Key idea: Model $\mathcal{R} = \{<, \sqsubset\}$ denotes pseudoknot-free structures.

2-intervals: Structured subsets

Definition (The 2-INTERVAL PATTERN problem)

- **Input**: A set of 2-intervals $\mathcal{D}$, a model $\mathcal{R}$ and a positive integer K.
- **Question**: Decide whether there exists a $\mathcal{R}$-structured subset $\mathcal{D}' \subseteq \mathcal{D}$ of size K.

If each 2-interval $D \in \mathcal{D}$ is associated with a non-negative weight $w(D)$, we are left with the WEIGHTED 2-INTERVAL PATTERN problem (*i.e.*, decide whether there exist a $\mathcal{R}$-structured subset $\mathcal{D}' \subseteq \mathcal{D}$ of total weight at least K).

2-intervals: Support and restriction

Definition (Support)

The **support** of a set of 2-intervals $\mathcal{D}$ is the set of intervals $\{I, J : (I, J) \in \mathcal{D}\}$.

Definition (Restriction)

Support restrictions:

- **Unlimited**: no restriction.
- **Balanced**: every 2-intervals consists of two intervals of equal length.
- **Unit**: every 2-intervals consists of two intervals of unit length.
- **Point**: the intervals in the support are pairwise disjoint.

2-intervals: State-of-the-art

Model	Unlimited	Balanced	Unit	Point
$\{<, \sqsubset, \not\between\}$		**APX**-hard		$O(n\sqrt{n})$
$\{\sqsubset, \not\between\}$		**APX**-hard		$O(n\log(n)+\mathcal{L})$
$\{<, \not\between\}$		**NP**-complete		
$\{<, \sqsubset\}$		$O(n\log(n)+dn)$		
$\{\not\between\}$		$O(n\log(n)+\mathcal{L})$		
$\{<\}$		$O(n\log(n))$		
$\{\sqsubset\}$		$O(n\log(n))$		

2-intervals: Approximation ratios

Model	Unlimited	Balanced	Unit	Point
$\{<, \sqsubset, \between\}$	4	4	$2+\epsilon$	N/A
$\{\sqsubset, \between\}$	4	4	$2+\epsilon$	N/A
$\{<, \between\}$		PTAS		

2-intervals: Model $\mathcal{R} = \{<, \sqsubset, \between\}$

Theorem

The 2-INTERVAL PATTERN *problem for unlimited (resp. unit) 2-intervals and model* $\mathcal{R} = \{<, \sqsubset, \between\}$ *is approximable within ratio* 4 *(resp.* 3*.).*

Remarks

- The approximation algorithm for unit 2-intervals is $O(n \log(n))$ time, where n is the number of input 2-intervals.
- The approximation algorithm for unlimited 2-interval uses linear programming techniques, which in practice are very often too time costly.

2-intervals: Model $\mathcal{R} = \{<, \sqsubset, \between\}$

Theorem

The 2-INTERVAL PATTERN *problem for balanced 2-intervals and model* $\mathcal{R} = \{<, \sqsubset, \between\}$ *is approximable within ratio* 4 ($O(n^2)$ *time algorithm).*

Proof.

Data: A set of balanced 2-intervals $\mathcal{D}$
Result: A $\{<, \sqsubset, \between\}$-structured subset of $\mathcal{D}$

```
1 D_sol ← ∅
2 while D ≠ ∅ do
3 |   Let D_min be the smallest 2-interval left in D
4 |   D_sol ← D_sol ∪ {D_min}
5 |   D ← D \ {D ∈ D : D ∩ D_min ≠ ∅}
6 return D_sol
```

□

2-intervals: Covering intervals

Definition

Let $D = (I, J)$ be a 2-interval. The **covering interval** of D, denoted $c(D)$, is the smallest interval that covers D (*i.e.*, $c(D) = [l(I), r(J)]$, where $l(i)$ (resp. $r(J)$) is the left (resp. right) endpoint of I (resp. J).

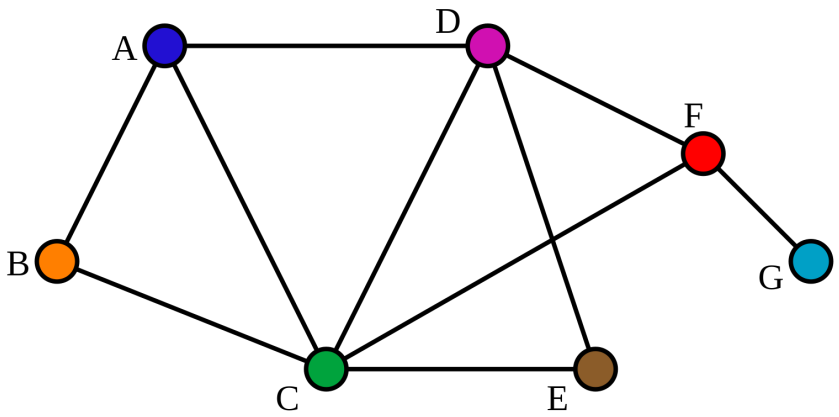
Observation

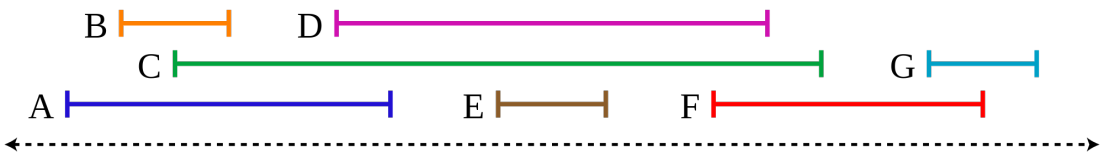
Let $\mathcal{D}$ be a set of 2-intervals. For any $\{\sqsubset, \between\}$-structured subset $\mathcal{D}' \subseteq \mathcal{D}$, the associated covering intervals $c(\mathcal{D}')$ are pairwise intersecting.

Interval graphs

Definition (Interval graph)

An **interval graph** is the intersection graph of a multiset of intervals on the real line. It has one vertex for each interval in the set, and an edge between every pair of vertices corresponding to intervals that intersect.





Interval graphs

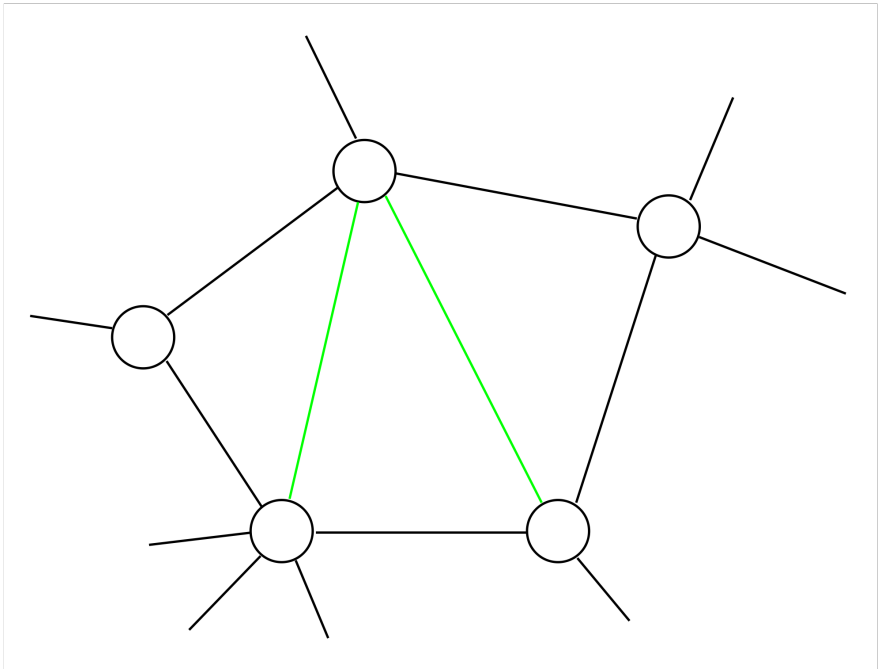
Remarks

- Determining whether a given graph $G = (V, E)$ is an interval graph can be done in $O(|V| + |E|)$ time by seeking an ordering of the maximal cliques of G that is consecutive with respect to vertex inclusion.
- A graph is an interval graph if and only if it is chordal and its complement is a comparability graph.
- The number of maximal cliques in a chordal graph is linear in the size of the graph.

Chordal graphs

Definition (Chordal graph)

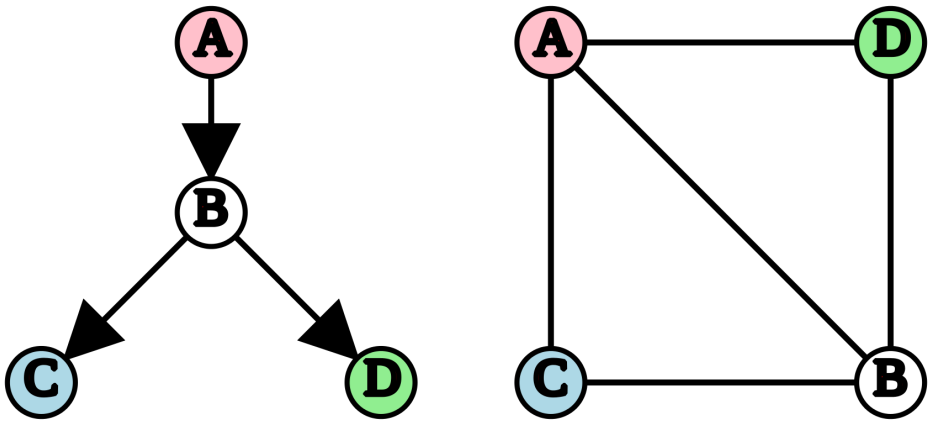
A graph is **chordal** if each of its cycles of four or more vertices has a chord, which is an edge that is not part of the cycle but connects two vertices of the cycle.



Comarability graphs

Definition (Comparability graph)

A **comparability graph** is a graph that has a transitive orientation, an assignment of directions to the edges of the graph (*i.e.*, an orientation of the graph) such that the adjacency relation of the resulting directed graph is transitive: whenever there exist directed edges (x, y) and (y, z), there must exist an edge (x, z).



2-intervals: $\mathcal{R} = \{\sqsubset, \between\}$

Theorem

The 2-INTERVAL PATTERN *problem for unlimited (resp. unit) 2-intervals and model* $\mathcal{R} = \{\sqsubset, \between\}$ *is approximable within ratio* 4 *(resp.* 3*). (The algorithm is* $O(n^2 \log(n))$ *time for unit 2-intervals.)*

Proof.

Data: A set of 2-intervals $\mathcal{D}$
Result: A $\{\sqsubset, \between\}$-structured subset of $\mathcal{D}$

```
c(D) ← {c(D) : D ∈ D}
K ← all maximal cliques of Ω(c(D))
foreach maximal clique K ∈ K do
    D_K ← {D ∈ D : c(D) ∈ K}
    D'_K ← (Approximate) pairwise disjoint subset of D_K
return the largest D'_K found
```

□

Trapezoid graphs

Definition (Trapezoid)

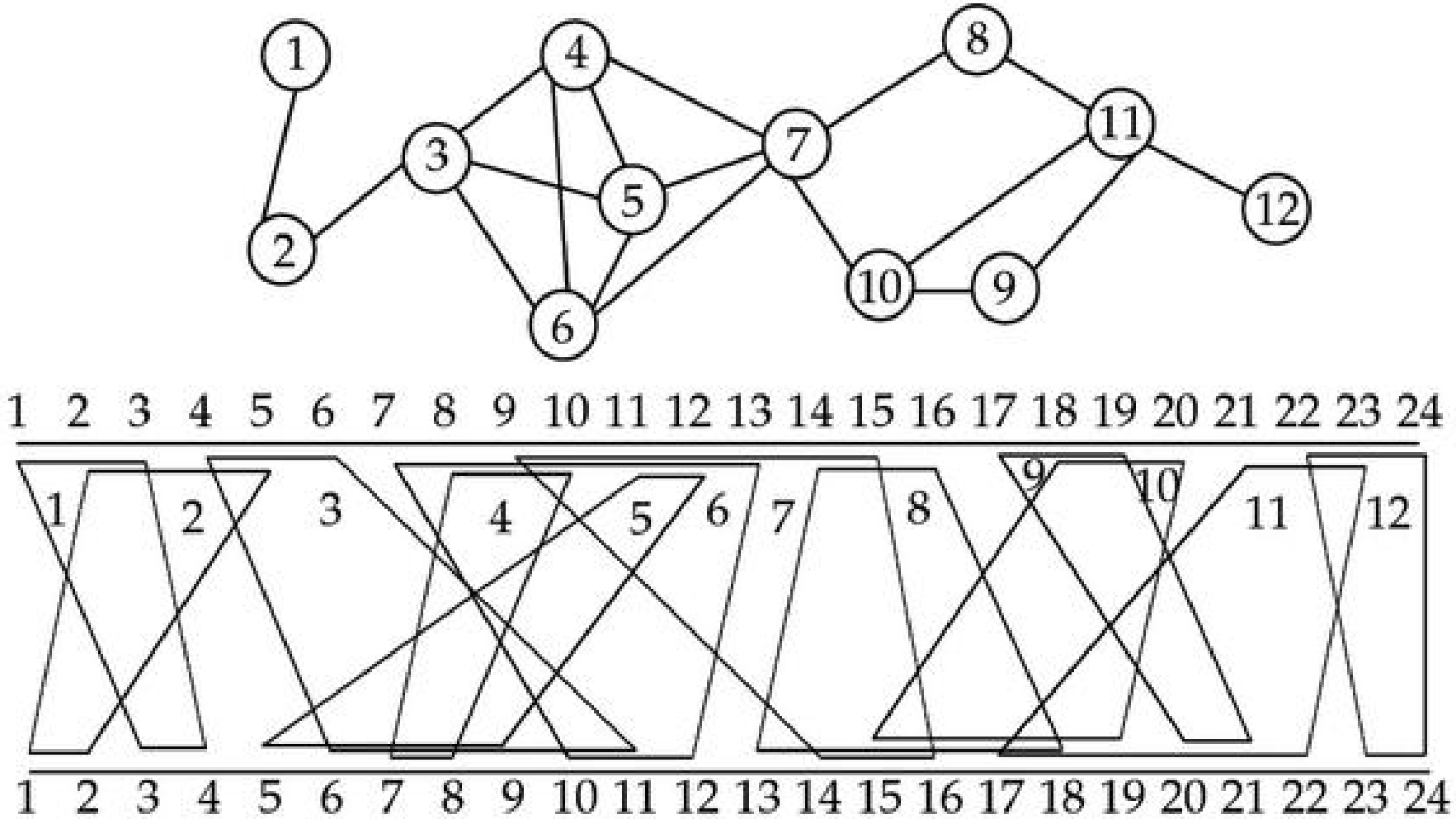
Consider two intervals I and J defined over two distinct horizontal lines. The **trapezoid** $T = (I, J)$ is the convex set of points bounded by I and J, and the two arcs connecting the right and left endpoints of I and J. The interval I and J are the **top interval** and the **bottom interval** of T.

A **family of trapezoids** is a finite set of trapezoids which are all defined over the same two horizontal lines.

Definition (Trapezoid graph)

A **trapezoid graph** is the intersection graph of a family of trapezoids.

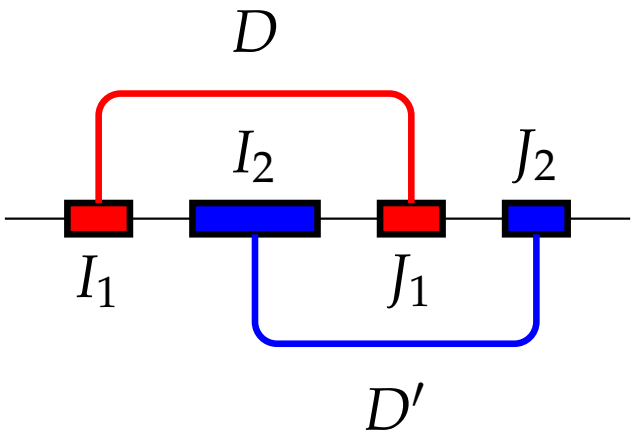
Trapezoid graphs

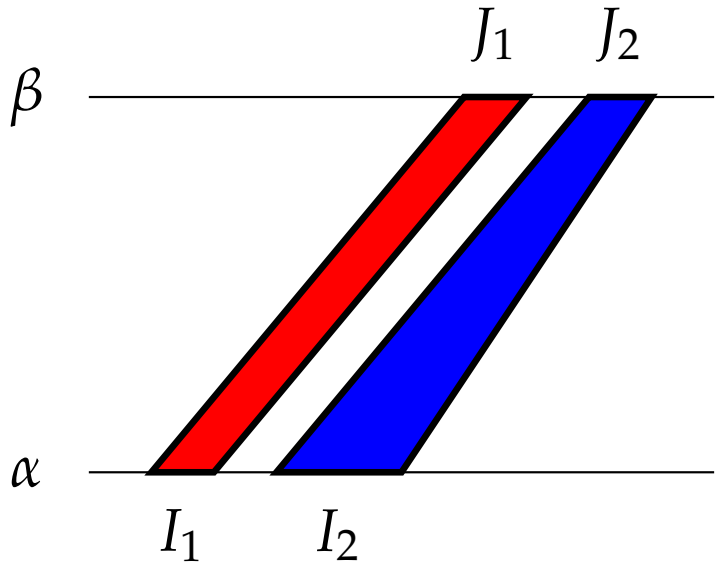


Turning 2-intervals to trapezoids

Definition (Corresponding trapezoid family)

Let $\mathcal{D}$ be a set of 2-intervals and let α and β be two distinct horizontal lines such that α is below β. The **corresponding trapezoid family** of $\mathcal{D}$, denoted $\mathcal{T}(\mathcal{D})$, is defined as the family containing a single trapezoid $T = (I', J')$ for each 2-interval $D = (I, J) \in \mathcal{D}$, where I' is defined over α, J' is defined over β, and $I' = I$ and $J' = J$.





2-intervals: Model $\mathcal{R} = \{<, \between\}$

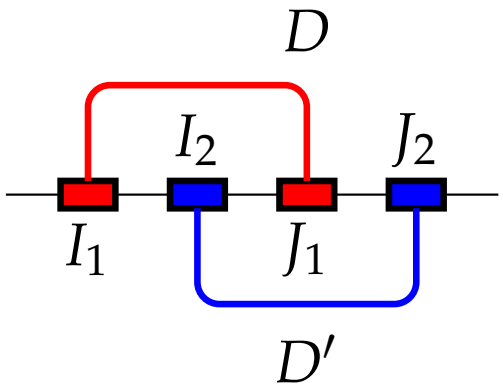
Observation

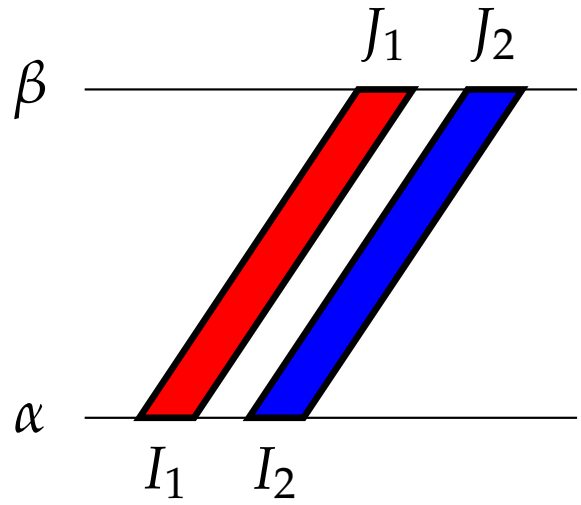
Any two **disjoint** 2-intervals in $\mathcal{D}$ are $\{<, \between\}$-comparable if and only their corresponding trapezoids in $\mathcal{T}(\mathcal{D})$ are disjoint.

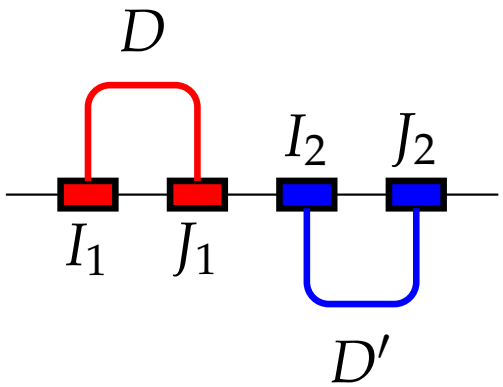
Remarks

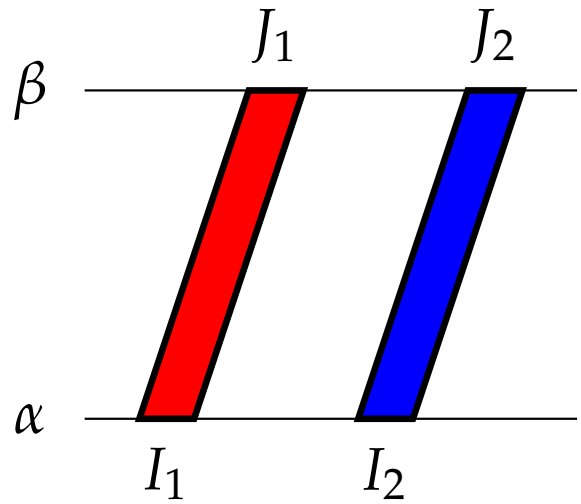
- Felsner et al. gave a $O(n \log(n))$ time algorithm for finding a maximum disjoint subset in a family of trapezoids.
- But there may be disjoint trapezoids in $\mathcal{T}(\mathcal{D})$ that correspond to non-disjoint 2-intervals in $\mathcal{D}$.
- $\{\sqsubset\}$-comparable 2-intervals in $\mathcal{D}$ correspond to intersecting trapezoids in $\mathcal{T}(\mathcal{D})$.

Turning 2-intervals to trapezoids

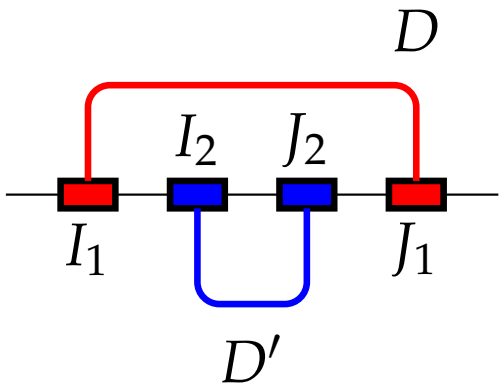


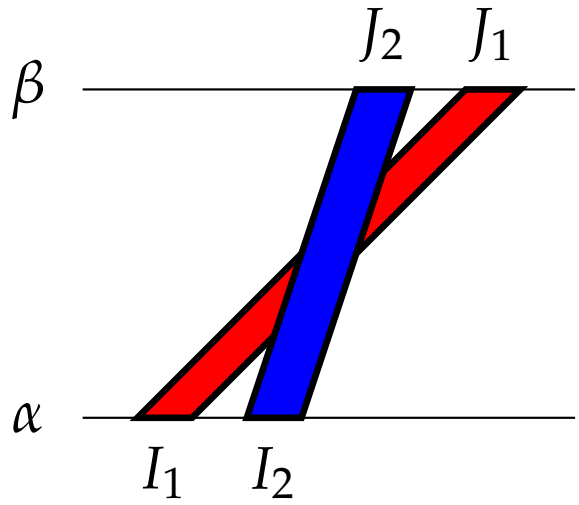


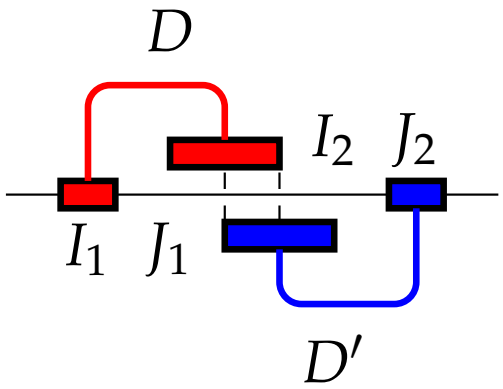


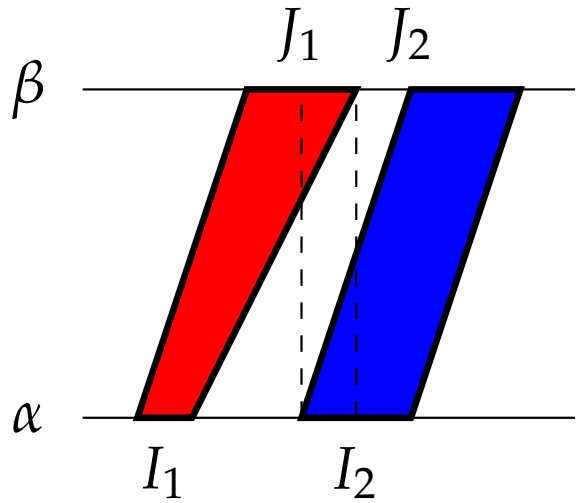


Turning 2-intervals to trapezoids







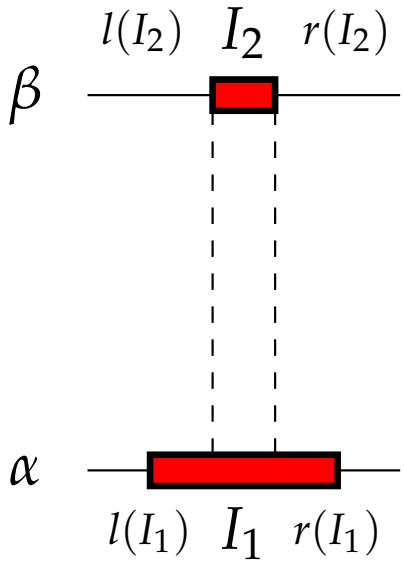


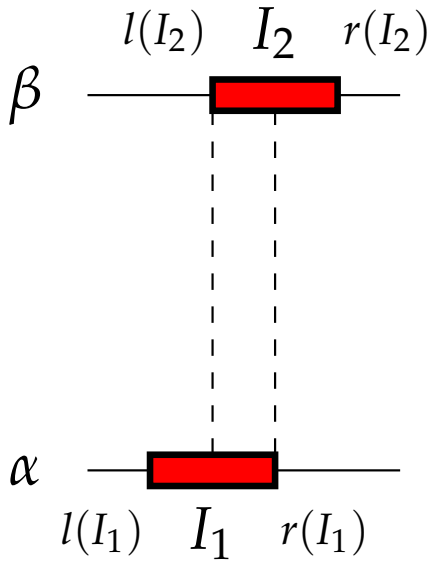
Turning 2-intervals to trapezoids

Definition (Clashing intervals)

Let $I_1 = [l(I_1), r(I_1)]$ and $I_2 = [l(I_2), r(I_2)]$ be two distinct intervals defined over two distinct horizontal lines such that $l(I_1) < l(I_2)$. The two intervals I_1 and I_2 **clash** if either

- $l(I_1) \leq l(I_2) \leq r(I_2) \leq r(I_1)$, or
- $l(I_1) \leq l(I_2) \leq r(I_1) \leq r(I_2)$.



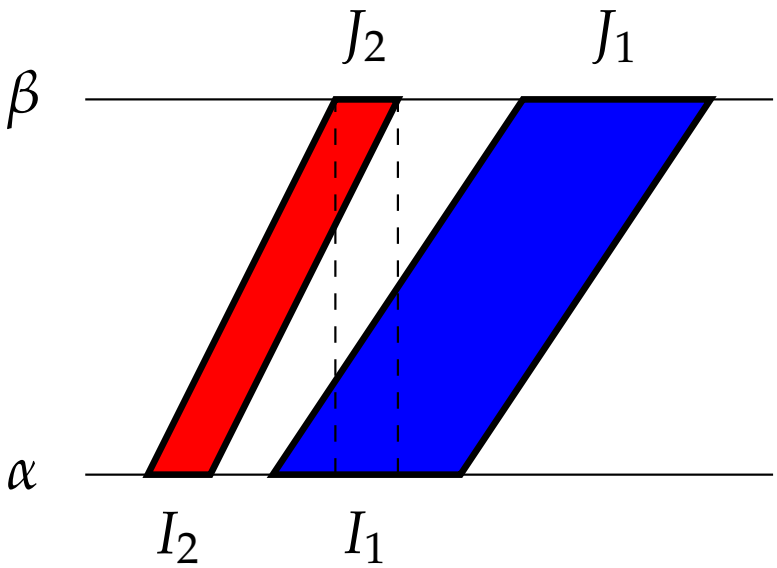


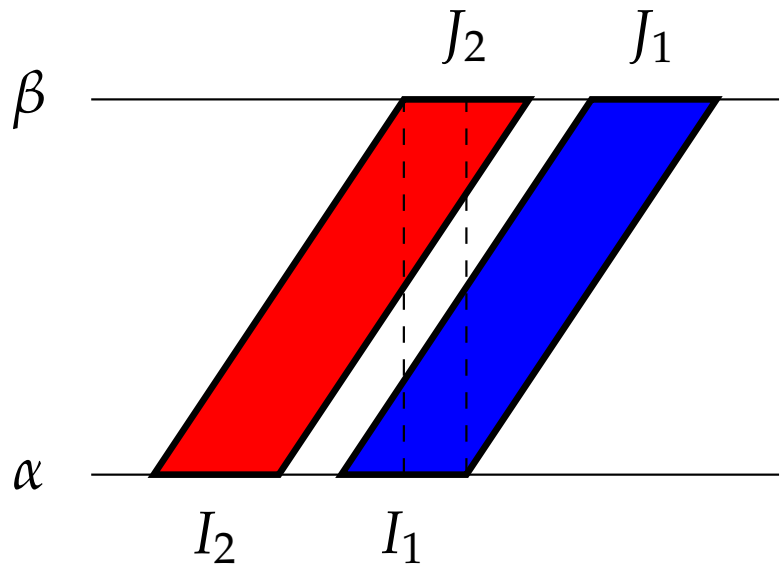
Turning 2-intervals to trapezoids

Definition (Clashing trapezoids)

Let $T_1 = (I_1, J_1)$ and $T_2 = (I_2, J_2)$ be two distinct trapezoids in a family of trapezoids. The two trapezoids T_1 and T_2 **clash** if either

- I_1 and J_2 clash, or
- I_2 and J_1 clash.





Turning 2-intervals to trapezoids

Observation

Any two distinct 2-intervals in $\mathcal{D}$ are $\{<, \between\}$-comparable if and only their corresponding trapezoids in $\mathcal{T}(\mathcal{D})$ are disjoint and do not clash.

2-intervals: Model $\mathcal{R} = \{<, \between\}$

Theorem

The 2-INTERVAL PATTERN *problem for unit (resp. point) 2-intervals and model $\mathcal{R} = \{<, \between\}$ is approximable within ratio 3 (resp. 2). (The algorithm is $O(n^2)$ time for unit 2-intervals.)*

Proof.

Data: A set of 2-intervals $\mathcal{D}$
Result: A $\{<, \between\}$-structured subset of $\mathcal{D}$
1 Construct the corresponding trapezoid family $\mathcal{T}(\mathcal{D})$
2 Compute $\mathcal{T}' \subseteq \mathcal{T}(\mathcal{D})$: the maximum pairwise disjoint subset of $\mathcal{T}(\mathcal{D})$
3 $\mathcal{T}_{\text{sol}} \leftarrow \emptyset$
4 **while** $\mathcal{T}' \neq \emptyset$ **do**
5 Let T_0 be the leftmost trapezoid in $\mathcal{T}'$
6 $\mathcal{T}_{\text{sol}} \leftarrow \mathcal{T}_{\text{sol}} \cup \{T_0\}$
7 Omit T_0 and all trapezoids clashing with T_0 in $\mathcal{T}'$
8 **return** *the 2-intervals corresponding to trapezoids in* $\mathcal{T}_{sol}$

2-intervals: Model $\mathcal{R} = \{<, \between\}$

Theorem

The 2-INTERVAL PATTERN *problem for unit (resp. point) 2-intervals and model* $\mathcal{R} = \{<, \between\}$ *is approximable within ratio* 3 *(resp.* 2*). (The algorithm is* $O(n^2)$ *time for unit 2-intervals.)*

Proof.

Data: A set of 2-intervals $\mathcal{D}$
Result: A $\{<, \between\}$-structured subset of $\mathcal{D}$

Construct the corresponding trapezoid family $\mathcal{T}(\mathcal{D})$
Compute $\mathcal{T}' \subseteq \mathcal{T}(\mathcal{D})$: the maximum pairwise disjoint subset of $\mathcal{T}(\mathcal{D})$
$\mathcal{T}_{\text{sol}} \leftarrow \emptyset$
while $\mathcal{T}' \neq \emptyset$ **do**
    Let T_0 be the leftmost trapezoid in $\mathcal{T}'$
    $\mathcal{T}_{\text{sol}} \leftarrow \mathcal{T}_{\text{sol}} \cup \{T_0\}$
    Omit T_0 and all trapezoids clashing with T_0 in $\mathcal{T}'$
return *the* 2*-intervals corresponding to trapezoids in* $\mathcal{T}_{sol}$

□

2-intervals: Model $\mathcal{R} = \{<, \wr\}$

Theorem

The 2-INTERVAL PATTERN *problem for unit (resp. point) 2-intervals and model* $\mathcal{R} = \{<, \wr\}$ *is approximable within ratio* 3 *(resp.* 2*). (The algorithm is* $O(n^2)$ *time for unit 2-intervals.)*

Proof.

- Let T_0 be the leftmost trapezoid $\mathcal{T}'$ and let D_0 be its corresponding 2-interval in $\mathcal{D}$.
- By our definition of a 2-interval and of $\mathcal{T}(\mathcal{D})$, any trapezoid in $\mathcal{T}(\mathcal{D})$ has a bottom interval which is completely to the left of its top interval.
- Thus, T_0 can only clash with trapezoids on its right in $\mathcal{T}'$.

□

2-intervals: Model $\mathcal{R} = \{<, \wr\}$

Theorem

The 2-INTERVAL PATTERN *problem for unit (resp. point)* 2*-intervals and model* $\mathcal{R} = \{<, \wr\}$ *is approximable within ratio* 3 *(resp.* 2*). (The algorithm is* $O(n^2)$ *time for unit* 2*-intervals.)*

Proof.

- if $\mathcal{D}$ is a point 2-interval set, then all 2-intervals with left intervals intersecting the right interval of D_0 have the same left interval, and as $\mathcal{T}'$ is pairwise disjoint, at most one of these has a corresponding trapezoid in $\mathcal{T}'$.

□

2-intervals: Model $\mathcal{R} = \{<, \between\}$

Theorem

The 2-INTERVAL PATTERN *problem for unit (resp. point) 2-intervals and model* $\mathcal{R} = \{<, \between\}$ *is approximable within ratio* 3 *(resp.* 2*). (The algorithm is* $O(n^2)$ *time for unit 2-intervals.)*

Proof.

- if $\mathcal{D}$ is a unit 2-interval set, intersecting intervals involved in $\mathcal{D}$ must overlap.
- Thus, any trapezoid in $\mathcal{T}'$ clashing with T_0 corresponds to a 2-interval with a left interval which contains either endpoints, but not both, of the right interval of D_0.
- Since $\mathcal{T}'$ is pairwise disjoint, there can be at most two such trapezoids in $\mathcal{T}'$.

□

2-intervals: Model $\mathcal{R} = \{<, \between\}$

Theorem

The 2-INTERVAL PATTERN *problem for balanced* 2*-intervals and model* $\mathcal{R} = \{<, \between\}$ *is approximable within ratio* 5*. (The algorithm is* $O(n^2)$ *time.)*

Proof.

Data: A set of balanced 2-intervals $\mathcal{D}$
Result: A $\{<, \between\}$-structured subset of $\mathcal{D}$

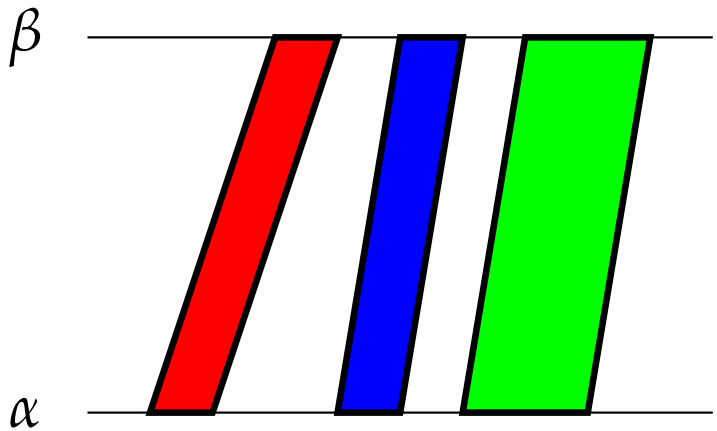
Construct the corresponding trapezoid family $\mathcal{T}(\mathcal{D})$
Compute $\mathcal{T}' \subseteq \mathcal{T}(\mathcal{D})$: the maximum pairwise disjoint subset of $\mathcal{T}(\mathcal{D})$
$\mathcal{T}_{\text{sol}} \leftarrow \emptyset$
while $\mathcal{T}' \neq \emptyset$ **do**
    Let T_0 be the smallest trapezoid in $\mathcal{T}'$
    $\mathcal{T}_{\text{sol}} \leftarrow \mathcal{T}_{\text{sol}} \cup \{T_0\}$
    Omit T_0 and all trapezoids clashing with T_0 in $\mathcal{T}'$
return *the* 2*-intervals corresponding to trapezoids in* $\mathcal{T}_{sol}$

□

2-intervals: Model $\mathcal{R} = \{<, \rotatebox{90}{)}\}$

Definition (Proper trapezoid family)

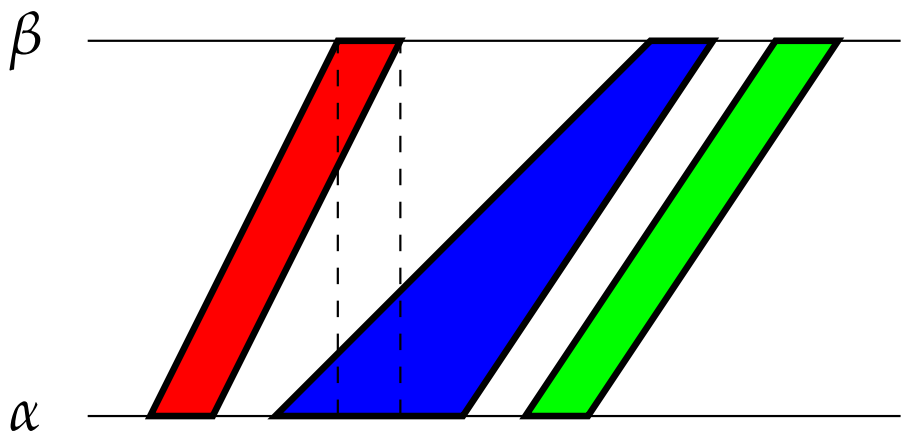
A family of trapezoids $\mathcal{T}$ is **proper** if for any two distinct trapezoids $T_1 = (I_1, J_1)$ and $T_2 = (I_2, J_2)$ in $\mathcal{T}$, $I_1 \cap I_2 = \emptyset$ and $J_1 \cap J_2 = \emptyset$.



2-intervals: Model $\mathcal{R} = \{<, \between\}$

Definition (Strongly proper trapezoid family)

A proper family of trapezoids $\mathcal{T}$ is **strongly proper** if for any two distinct trapezoids $T_1 = (I_1, J_1)$ and $T_2 = (I_2, J_2)$ in $\mathcal{T}$, if J_1 and I_2 clash then $l(I_2) \leq l(J_1) < r(J_1) \leq r(I_2)$, where $l(J_1), r(J_1)$ and $l(I_2), r(I_2)$ are the left and right endpoints of J_1 and I_2, respectively.



2-intervals: Model $\mathcal{R} = \{<, \wr\}$

Remarks

- Any pairwise disjoint family of trapezoids is proper (but the converse is not true).
- Thus Step 2 of the preceding algorithm yields a proper family of trapezoids $\mathcal{T}'$.

2-intervals: Model $\mathcal{R} = \{<, \between\}$

Remarks

- Computing a strongly proper subset $\mathcal{T}' \subseteq \mathcal{T}''$ can be done easily by adjusting the loop step: Instead of omitting all trapezoids clashing with the leftmost trapezoid in this iteration, we need only to omit a small subset of these trapezoids.

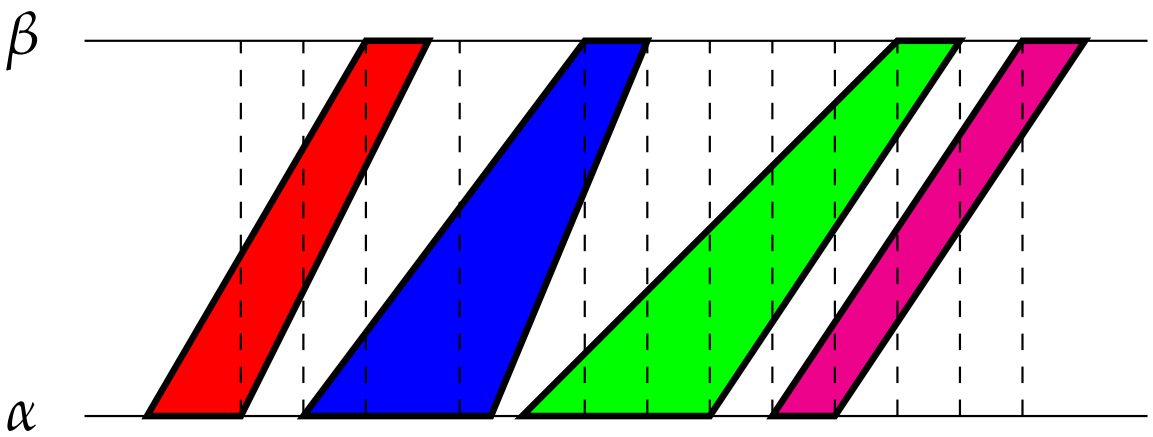
  More specifically, let $T_0 = (I_0, J_0)$ be the leftmost trapezoid in $\mathcal{T}'$. We only omit trapezoids $T = (I, J)$ with
  - $l(I) \leq l(I_0) \leq r(I)$, or
  - $l(I) \leq r(I_0) \leq r(I)$.

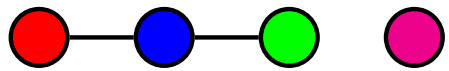
- We obtain a strongly proper trapezoid family $\mathcal{T}'' \subseteq \mathcal{T}'$ if we proceed in this fashion such that $3\,|\mathcal{T}''| \geq |\mathcal{T}'|$.

2-intervals: Model $\mathcal{R} = \{<, \between\}$

Definition (Clashing trapezoid graph)

Let $\mathcal{T}$ be a family of trapezoids. The **clashing trapezoid graph** of $\mathcal{T}$, denoted $G_{\mathcal{T}}$, is the graph such that each vertex in $G_{\mathcal{T}}$ correspond to a distinct trapezoid in $\mathcal{T}$, and two vertices are connected by an edge if and only if their corresponding trapezoid clash.





2-intervals: Model $\mathcal{R} = \{<, \rotatebox{90}{)}\!\!\rotatebox{90}{(}\}$

Theorem

Let $\mathcal{T}$ be a family of trapezoids. If $\mathcal{T}$ is strongly proper then $G_{\mathcal{T}}$ is a forest.

Proof.

2-intervals: Model $\mathcal{R} = \{<, \between\}$

Theorem

Let $\mathcal{T}$ be a family of trapezoids. If $\mathcal{T}$ is strongly proper then $G_{\mathcal{T}}$ is a forest.

Proof.

- Define $\vec{G}_{\mathcal{T}}$ as the directed graph obtained by orienting the edges of $G_{\mathcal{T}}$ according to the precedence relation in $\mathcal{T}$:

$$V(\vec{G}_{\mathcal{T}}) = V(G_{\mathcal{T}})$$
$$E(\vec{G}_{\mathcal{T}}) = \{(T_1, T_2) : \{T_1, T_2\} \in E(G_{\mathcal{T}}) \text{ and } T_1 < T_2\}$$

- Since $\mathcal{T}$ is strongly proper, every trapezoid in $\mathcal{T}$ clashes with at most one trapezoid on its left, and hence the in-degree of every vertex $T \in V(\vec{G}_{\mathcal{T}})$ is at most 1.

□

2-intervals: Model $\mathcal{R} = \{<, \overset{\frown}{\underset{\smile}{}}\}$

Theorem

Let $\mathcal{T}$ be a family of trapezoids. If $\mathcal{T}$ is strongly proper then $G_{\mathcal{T}}$ is a forest.

Proof.

- Hence any cycle $(T_0, \ldots, T_k, T_0)$ in $G_{\mathcal{T}}$ is a directed cycle in $\vec{G}_{\mathcal{T}}$.
- Then we must have $T_0 < T_k < T_0$ by definition of $\vec{G}_{\mathcal{T}}$. This is a contradiction.
- Then it follows that $G_{\mathcal{T}}$ is acyclic.

□

2-intervals: Model $\mathcal{R} = \{<, \between\}$

Remarks

- A maximum independent set in any forest of size n is of size at least $\frac{n}{2}$. (This set can be found in linear time with respect to n.)
- if $\mathcal{T}$ is a pairwise disjoint family of trapezoids, then any independent set of $G_{\mathcal{T}}$ corresponds to a pairwise disjoint non-clashing set of trapezoids, and hence corresponds to a $\{<, \between\}$-comparable set of 2-intervals.

2-intervals: Model $\mathcal{R} = \{<, \not\between\}$

Theorem

The 2-INTERVAL PATTERN *problem for model* $\mathcal{R} = \{<, \not\between\}$ *is approximable within ratio* 6. *(The algorithm is* $O(n^2)$ *time.)*

Proof.

Data: A set of 2-intervals $\mathcal{D}$
Result: A $\{<, \not\between\}$-structured subset of $\mathcal{D}$
1 Construct $\mathcal{T}(\mathcal{D})$, the corresponding trapezoid set of $\mathcal{D}$
2 Compute $\mathcal{T}'$, the maximum pairwise disjoint subset of $\mathcal{T}(\mathcal{D})$
3 Compute $\mathcal{T}''$, a strongly proper subset of $\mathcal{T}'$, such that $3|\mathcal{T}''| \geq |\mathcal{T}'|$
4 Compute $G_{\mathcal{T}''}$ and the maximum independent set of $G_{\mathcal{T}''}$
5 **return** *the* 2*-intervals corresponding to the maximum independent set of* $G_{\mathcal{T}''}$

□

2-intervals: Model $\mathcal{R} = \{<, \wr\}$

Theorem

The 2-INTERVAL PATTERN *problem for model* $\mathcal{R} = \{<, \wr\}$ *is approximable within ratio* 2. *(The algorithm is* $O(n^3 \log(n)))$ *time.)*

2-intervals: Model $\mathcal{R} = \{<, \not)\}$

Definition (Precedence or crossing)

For two 2-intervals $D_1 = (I_1, J_1)$ and $D_2 = (I_2, J_2)$, the **precedence or crossing** relation $\not)$ is defined by:

$$D_1 \not) D_2 \qquad \text{if and only if} \qquad D_1 < D_2 \text{ or } D_1 \not) D_2$$

Remarks

- If $D_1 \not) D_2$ then $I_1 < I_2$ and $J_1 < J_2$.
- The $\not)$ relation specifies a total order for any $\{<, \not)\}$-structured 2-interval set.

2-intervals: Model $\mathcal{R} = \{<, \wr\}$

Let $\mathcal{D}$ be a $\{<, \wr\}$-structured 2-interval set and let $\mathcal{D}_{\not\wr}$ denotes $\mathcal{D}$ ordered by the $\not\wr$ relation (viewed as an ordered sequence).

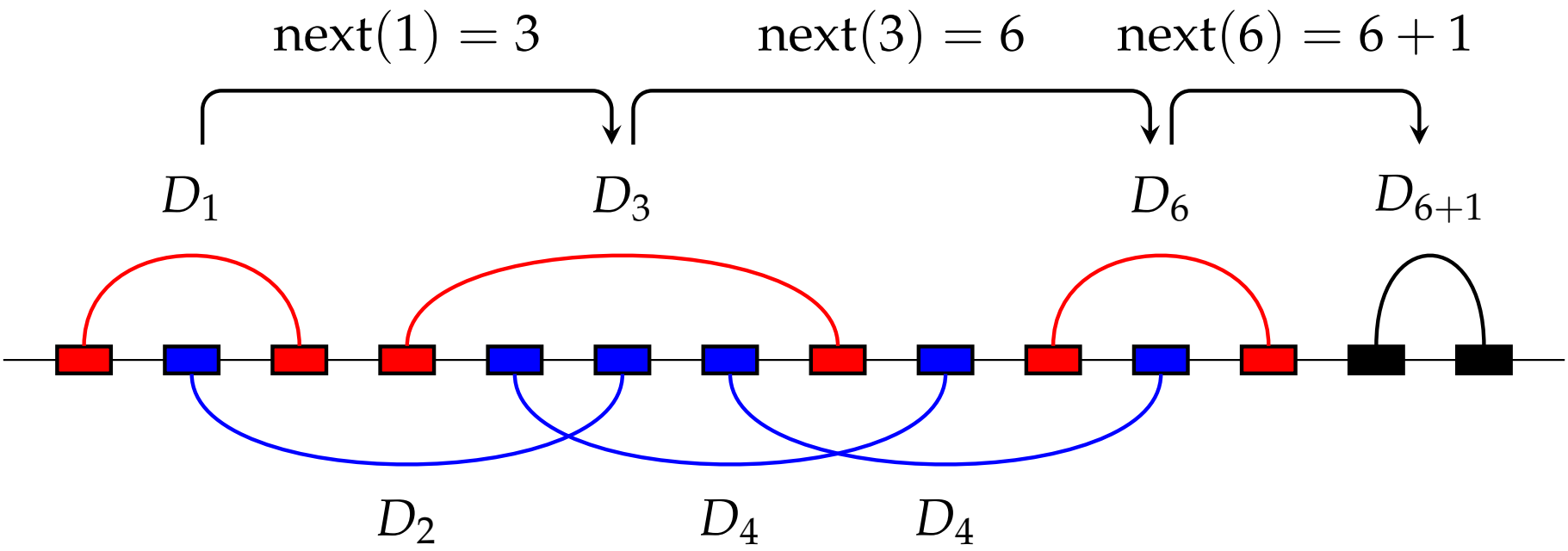
- $\mathcal{D}_{\not\wr}[i]$ denotes the 2-interval with rank i in $\mathcal{D}$.
- $\mathcal{D}_{\not\wr}[i,j]$ denotes the subsequence $\mathcal{D}_{\not\wr}[i], \mathcal{D}_{\not\wr}[i+1], \ldots, \mathcal{D}_{\not\wr}[j]$.

2-intervals: Model $\mathcal{R} = \{<, \between\}$

- For each index $1 \leq i \leq |\mathcal{D}_{\between}|$, define $\text{next}(i)$ as the smallest index j, $1 \leq j \leq |\mathcal{D}_{\between}|$ such that $\mathcal{D}_{\between}[i] < \mathcal{D}_{\between}[j]$. If such an index j does not exist define $\text{next}(i) = |\mathcal{D}_{\between}| + 1$.
- Define the **backbone indices** of $\mathcal{D}_{\between}$ as the sequence of indices $i_1, i_2, \ldots, i_k$ such that $i_1 = 1$, $i_j = \text{next}(i_{j-1})$ and $\text{next}(i_k) = |\mathcal{D}_{\between}| + 1$.

  (For convenience, we define $\text{next}(i_k) = |\mathcal{D}_{\between}| + 1$ and imagine a 2-interval $\mathcal{D}_{\between}[i_{k+1}]$ such that $\mathcal{D}_{\between}[i] < \mathcal{D}_{\between}[i_{k+1}]$ for all $1 \leq i \leq |\mathcal{D}_{\between}|$.)

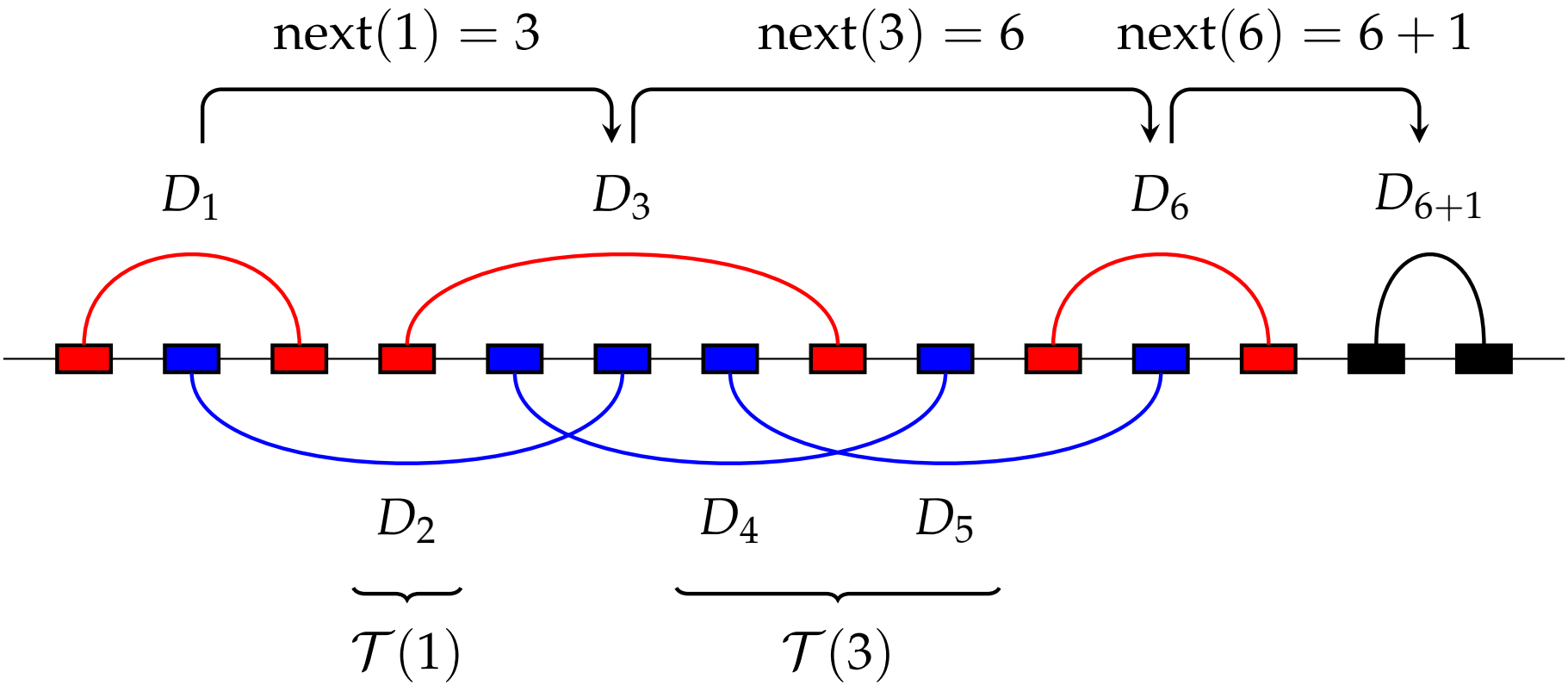
2-intervals: Model $\mathcal{R} = \{<, \wr\}$



2-intervals: Model $\mathcal{R} = \{<, \text{)(}\}$

- For each backbone index $1 \leq i_s \leq k$, define a **stripe** $\mathcal{T}(i_s) = \mathcal{D}_{\not x}[i_s + 1, i_{s+1} - 1]$.
- The stripe is **odd** if s is odd; it is **even** if s is even.
- For each 2-interval $D \in \mathcal{T}(i_s)$, we observe that $\mathcal{D}_{\not x}[i_s] \text{)(} D \not x \mathcal{D}_{\not x}[i_s + 1]$.
- Every stripe of $\mathcal{D}_{\not x}$ is $\{\text{)(}\}$-structured.

2-intervals: Model $\mathcal{R} = \{<, \wr\}$



2-intervals: Model $\mathcal{R} = \{<, \not\!\!\!\bigcirc\}$

- A $\{<, \not\!\!\!\bigcirc\}$-structured sequence $\mathcal{D}_{\not\!\!\!\bigcirc}$ is **striped** if either
  - its odd stripes are all empty, or
  - its even stripes are all empty.

- Although $\mathcal{D}_{\not\!\!\!\bigcirc}$ is not always striped, it contains two striped subsequences:

$$\mathcal{D}_{\not\!\!\!\bigcirc}[i_1]\ \mathcal{T}(i_1)\ \mathcal{D}_{\not\!\!\!\bigcirc}[i_2]\ \mathcal{D}_{\not\!\!\!\bigcirc}[i_3]\ \mathcal{T}(i_3)\ \mathcal{D}_{\not\!\!\!\bigcirc}[i_4]\ \ldots$$
$$\mathcal{D}_{\not\!\!\!\bigcirc}[i_1]\ \mathcal{D}_{\not\!\!\!\bigcirc}[i_2]\ \mathcal{T}(i_2)\ \mathcal{D}_{\not\!\!\!\bigcirc}[i_3]\ \mathcal{D}_{\not\!\!\!\bigcirc}[i_4]\ \mathcal{T}(i_4)\ \ldots$$

These two subsequences together cover the sequence $\mathcal{D}_{\not\!\!\!\bigcirc}$: the 2-intervals at the backbone indices are covered twice, each remaining 2-interval is covered once.

One of the two subsequences has a length of at least $|\mathcal{D}_{\not\!\!\!\bigcirc}|/2$.

2-intervals: Model $\mathcal{R} = \{<, \text{⟨}\text{⟩}\}$

- A $\{<, \text{⟨}\text{⟩}\}$-structured sequence $\mathcal{D}_{\text{⟨}\text{⟩}}$ is **striped** if either
  - its odd stripes are all empty, or
  - its even stripes are all empty.

- Although $\mathcal{D}_{\text{⟨}\text{⟩}}$ is not always striped, it contains two striped subsequences:

$$\mathcal{D}_{\text{⟨}\text{⟩}}[i_1]\ \mathcal{T}(i_1)\ \mathcal{D}_{\text{⟨}\text{⟩}}[i_2]\ \mathcal{D}_{\text{⟨}\text{⟩}}[i_3]\ \mathcal{T}(i_3)\ \mathcal{D}_{\text{⟨}\text{⟩}}[i_4]\ \ldots$$
$$\mathcal{D}_{\text{⟨}\text{⟩}}[i_1]\ \mathcal{D}_{\text{⟨}\text{⟩}}[i_2]\ \mathcal{T}(i_2)\ \mathcal{D}_{\text{⟨}\text{⟩}}[i_3]\ \mathcal{D}_{\text{⟨}\text{⟩}}[i_4]\ \mathcal{T}(i_4)\ \ldots$$

  These two subsequences together cover the sequence $\mathcal{D}_{\text{⟨}\text{⟩}}$: the 2-intervals at the backbone indices are covered twice, each remaining 2-interval is covered once.

  One of the two subsequences has a length of at least $|\mathcal{D}_{\text{⟨}\text{⟩}}|/2$.

2-intervals: Model $\mathcal{R} = \{<, \wr\}$

Theorem

The 2-INTERVAL PATTERN *problem for model* $\mathcal{R} = \{<, \wr\}$ *is approximable within ratio* 2. *(The algorithm is* $O(n^3 \log(n)))$ *time.)*

Proof.

2-intervals: Model $\mathcal{R} = \{<, \between\}$

Theorem

The 2-INTERVAL PATTERN *problem for model* $\mathcal{R} = \{<, \between\}$ *is approximable within ratio* 2. *(The algorithm is* $O(n^3 \log(n)))$ *time.)*

Proof.

- **Step 1.** Make a dummy 2-interval D_ω such that $D_\gamma < D_\omega$ for all $D_\gamma \in \mathcal{D}$.

  Set $\mathcal{D}^+ = \mathcal{D} \cup \{D_\omega\}$.

□

2-intervals: Model $\mathcal{R} = \{<, \not)($\}$

Theorem

The 2-INTERVAL PATTERN *problem for model* $\mathcal{R} = \{<, \not)($\}$ *is approximable within ratio* 2. *(The algorithm is* $O(n^3 \log(n)))$ *time.)*

Proof.

- **Step 2.** For each pair of 2-intervals D_α and D_β in $\mathcal{D}^+$, $D_\alpha < D_\beta$, find the subset of 2-intervals

$$\mathcal{D}^+_{\alpha,\beta} = \{D_\gamma : D_\gamma \in \mathcal{D}^+ \text{ and } D_\alpha \not)(D_\gamma \not)(D_\beta\}$$

Then compute $\mathcal{C}_{\alpha,\beta}$, a maximum size $\{\not)(\}$-structured subset of $\mathcal{D}^+_{\alpha,\beta}$.

□

2-intervals: Model $\mathcal{R} = \{<, \between\}$

Theorem

The 2-INTERVAL PATTERN *problem for model* $\mathcal{R} = \{<, \between\}$ *is approximable within ratio* 2. *(The algorithm is* $O(n^3 \log(n)))$ *time.)*

Proof.

- **Step 3.1.** Process the 2-intervals in $\mathcal{D}^+_{\alpha,\beta}$ in an arbitrary order that conforms to the partial order specified by the $<$ relation. For each 2-interval D_β in $\mathcal{D}^+$, find the subset of 2-intervals

$$\mathcal{D}^+_\beta = \{D_\alpha : D_\alpha \in \mathcal{D}^+ \text{ and } D_\alpha < D_\beta\}$$

□

2-intervals: Model $\mathcal{R} = \{<, \between\}$

Theorem

The 2-INTERVAL PATTERN *problem for model* $\mathcal{R} = \{<, \between\}$ *is approximable within ratio* 2. *(The algorithm is* $O(n^3 \log(n)))$ *time.)*

Proof.

- **Step 3.2.** If $\mathcal{D}_\beta^+ = \emptyset$, $\mathcal{A}_\beta \leftarrow \{\beta\}$ and $\mathcal{B}_\beta \leftarrow \{\beta\}$. Otherwise
  - Find $D_\alpha \in \mathcal{D}_\beta^+$ such that $|\mathcal{B}_\alpha|$ is maximum and set $\mathcal{A}_\beta \leftarrow \mathcal{B}_\beta \cup \{D_\beta\}$,
  - Find $D_\alpha \in \mathcal{D}_\beta^+$ such that $|\mathcal{A}_\alpha| + |\mathcal{C}_{\alpha,\beta}|$ is maximum and set $\mathcal{B}_\beta \leftarrow \mathcal{A}_\alpha \cup \mathcal{C}_{\alpha,\beta} \cup \{D_\beta\}$.

□

2-intervals: Model $\mathcal{R} = \{<, \between\}$

Theorem

The 2-INTERVAL PATTERN *problem for model* $\mathcal{R} = \{<, \between\}$ *is approximable within ratio* 2. *(The algorithm is* $O(n^3 \log(n)))$ *time.)*

Proof.

- **Step 4.** Let $\mathcal{D}_{\text{sol}}$ be either $\mathcal{A}_\omega$ or $\mathcal{B}_\omega$ such that $|\mathcal{D}_{\text{sol}}|$ is maximum.

  Return $\mathcal{D}_{\text{sol}} \setminus \{\omega\}$.

□

2-intervals: Model $\mathcal{R} = \{<, \between\}$

Theorem

The 2-INTERVAL PATTERN *problem for model* $\mathcal{R} = \{<, \between\}$ *is approximable within ratio* 2. *(The algorithm is* $O(n^3 \log(n)))$ *time.)*

Proof.

Notes

- In the algorithm we use $\mathcal{A}_\beta$ and $\mathcal{B}_\beta$ to represent the two different alternating patterns, with β as both the last element backbone element.
- The 2-interval D_α in steps 3.1 and 3.2 represents the second-to-last backbone element in $\mathcal{A}_\beta$ and $\mathcal{B}_\beta$.
- The subset $\mathcal{C}_{\alpha,\beta}$ represents the maximum size stripe between the two backbone elements D_α and D_β.

□

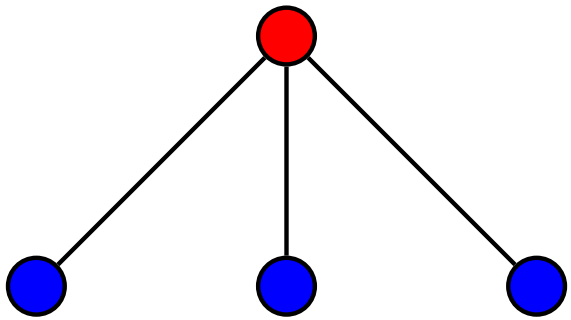
d-claw free graphs

Definition (d-claw, d-claw-free)

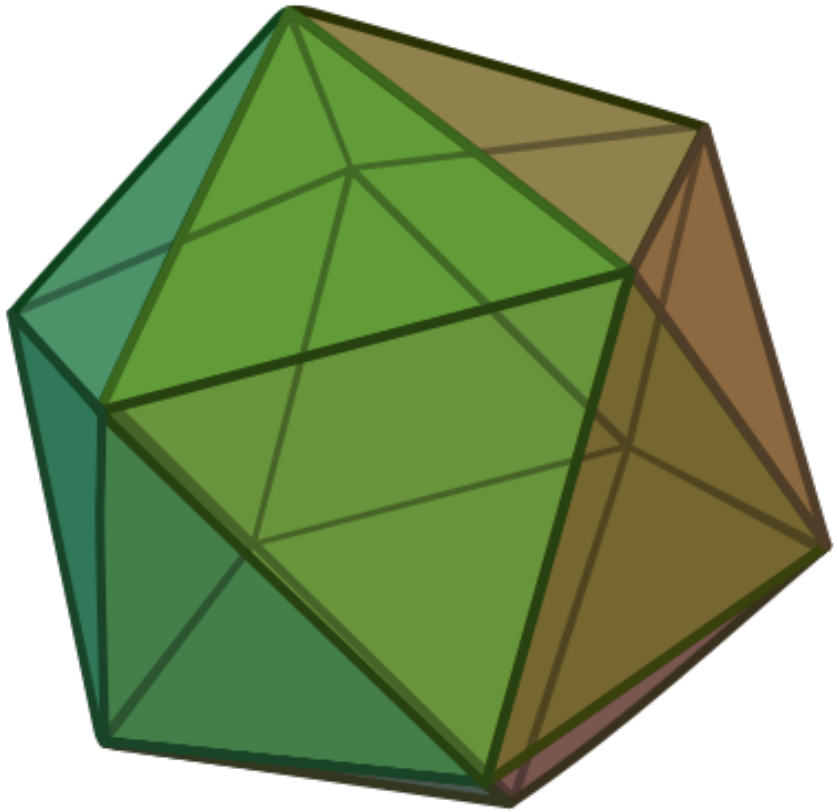
For an undirected graph G, a **d-claw** C is an induced subgraph $K_{1,d}$ that consists of an independent set T_c of d vertices (called **talons**) and a **center** vertex z_C that is connected to all the talons.

A graph is **d-claw-free** if it has no d-claws.

(A 3-claw is commonly called a claw so that a graph is claw-free if and only if it does not contain the complete bipartite graph $K_{1,3}$ (known as the "**claw graph**") as an induced subgraph.)



d-claw free graphs



The regular icosahedron, a polyhedron whose vertices and edges form a claw-free graph.

d-claw free graphs

Definition (The MAXIMUM WEIGHT INDEPENDENT SET problem)

- **Input**: A graph $G = (V, E)$ and a weight function $w : V \to \mathbb{N}$.
- **Solution**: A set of independent vertices $V' \subseteq V$ that maximises $\sum_{v \in V'} w(v)$.

Remarks

In d-claw-free graphs

- Arbitrary weight: $(d/2 + \epsilon)$-approximation
- Small weight: $(d/2)$-approximation
- Unit weight: $((d-1)/2 + \epsilon)$-approximation

2-intervals and d-claw free graphs

Theorem

For a set of 2-intervals $\mathcal{D}$ with interval length ℓ, $a \leq \ell \leq b$, the 2-interval graph $G(\mathcal{D})$ is d-claw-free for

$$d = 5 + \frac{2(b-2)}{a}.$$

Proof.

- Let I be an interval and let $\mathcal{I}$ be a set of disjoint intervals that intersect I.
- All intervals in $\mathcal{I}$ are completely contained in I except possibly the leftmost one and the rightmost one.

2-intervals and d-claw free graphs

Theorem

For a set of 2-intervals $\mathcal{D}$ with interval length ℓ, $a \leq \ell \leq b$, the 2-interval graph $G(\mathcal{D})$ is d-claw-free for

$$d = 5 + \frac{2(b-2)}{a}.$$

Proof.

- Let I be an interval and let $\mathcal{I}$ be a set of disjoint intervals that intersect I.
- All intervals in $\mathcal{I}$ are completely contained in I except possibly the leftmost one and the rightmost one.

□

2-intervals and d-claw free graphs

Corollary

Let $\mathcal{D}$ be a set of 2-intervals.

- *if all intervals have the same length (unit support), the associated 2-interval graph $G(\mathcal{D})$ is 5-claw-free;*
- *if all intervals have length 2 or 3, the associated 2-interval graph $G(\mathcal{D})$ is 5-claw-free*

Corollary

The (WEIGHTED) 2-INTERVAL PATTERN *problem is approximable within ratio $2.5 + \epsilon$ for arbitrary weights and $2 + \epsilon$ for unit weights.*

Nearest Neighbor Energy Model

For a secondary structure S

- the number of **base pairs stackings** is

$$\mathrm{BPS}(S) = |\{(i,j) \in S : (i+1, j-1) \in S\}|$$

- the number of **stacking base pairs** is

$$\mathrm{SBP}(S) = |\{(i,j) \in S : (i+1, j-1) \in S \text{ or } (i-1, j+1) \in S\}|$$

Nearest Neighbor Energy Model

Definition (The MAXIMUM BASE PAIRS STACKING (BPS) problem)

- **Input**: A sequence u.
- **Solution**: A secondary structure S for u that maximises $\mathrm{BPS}(S)$.

Definition (The MAXIMUM STACKINGBASE PAIRS (SBP) problem)

- **Input**: A sequence u.
- **Solution**: A secondary structure S for u that maximises $\mathrm{SBP}(S)$.

Approximating the MAXIMUM BASE PAIRS STACKING problem

Theorem

The MAXIMUM BASE PAIRS STACKING *problem is approximable within ratio* $8/3$.

Proof.

Approximating the MAXIMUM BASE PAIRS STACKING problem

Theorem

The MAXIMUM BASE PAIRS STACKING *problem is approximable within ratio* $8/3$.

Proof.

- **Step 1**. Repeatedly find the leftmost 5 consecutive stacking loops (*i.e.*, find the 2-interval $([x, x+5], [y-5, y])$ where x is as small as possible). Add these stacking loops to S.
- **Step 2**. Repeatedly find any 4 consecutive stacking loops. Add these stacking loops to S.
- **Step 3**. Repeatedly find any 3 consecutive stacking loops. Add these stacking loops to S.

□

Approximating the MAXIMUM BASE PAIRS STACKING problem

Theorem

The MAXIMUM BASE PAIRS STACKING *problem is approximable within ratio* $8/3$.

Proof.

- **Step 4.1**. Construct a 2-interval set $\mathcal{D}$ by associating a 2-interval to each 2 consecutive stacking loop.
- **Step 4.2**. Construct the 2-interval graph $G(\mathcal{D})$ and assign each vertex a weight: 1 for a single stacking loop and 2 for two consecutive stacking loops.

□

Approximating the MAXIMUM BASE PAIRS STACKING problem

Theorem

The MAXIMUM BASE PAIRS STACKING *problem is approximable within ratio* $8/3$.

Proof.

- **Step 4.3**. Find a maximum weight independent set $\mathcal{D}'$ in $G(\mathcal{D})$ (5/2-approximation algorithm for 5-claw-free graphs).
- **Step 4.4**. For each 2-interval in $\mathcal{D}'$, add the corresponding stacking loop in S.

□

Approximating the MAXIMUM BASE PAIRS STACKING problem

Theorem

The MAXIMUM BASE PAIRS STACKING *problem is approximable within ratio* $8/3$.

Proof.

- Let s_1, s_2, s_3 and s_4, respectively, be the number of stacking loops found by the first, second, third and fourth of our algorihm.

□

Approximating the MAXIMUM BASE PAIRS STACKING problem

Theorem

The MAXIMUM BASE PAIRS STACKING *problem is approximable within ratio* $8/3$.

Proof.

- Let S^* be the set of stacking loops in an optimal secondary structure.
- Let s_1^*, s_2^* and s_3^*, respectively, be the number of stacking loops in S^* that intersect the stacking loops found by the first, second and third step of our algorithm.
- Let s_4^* be the number of remaining stacking loops in S^* which are represented by 2-intervals in $\mathcal{D}$.

□

Approximating the MAXIMUM BASE PAIRS STACKING problem

Theorem

The MAXIMUM BASE PAIRS STACKING *problem is approximable within ratio* $8/3$.

Proof.

$$|S| = s_1 + s_2 + s_3 + s_4$$
$$|S^*| = s_1^* + s_2^* + s_3^* + s_4^*$$

□

Approximating the MAXIMUM BASE PAIRS STACKING problem

Theorem

The MAXIMUM BASE PAIRS STACKING *problem is approximable within ratio* $8/3$.

Proof.

For each k consecutive stacking loops D found by the first three steps of our algorithm, the number of stacking loops in S^* that intersect them is at most $2(k+2)$ (*i.e.*, $k+2$ for each interval of the 2-interval D).

□

Approximating the MAXIMUM BASE PAIRS STACKING problem

Theorem

The MAXIMUM BASE PAIRS STACKING *problem is approximable within ratio* $8/3$.

Proof.

Step 1.

- By always choosing the leftmost 5-consecutive stacking loop D_5, we can guarantee that the left interval of the 2-interval D_5 intersects at most $5+1$ stacking loops in S^*.

□

Approximating the MAXIMUM BASE PAIRS STACKING problem

Theorem

The MAXIMUM BASE PAIRS STACKING *problem is approximable within ratio* $8/3$.

Proof.

Step 1.

- Suppose the contrary that the left interval of D_5 intersects 7 stacking loops in S^*.
- Then these 7 stacking loops must be consecutive, and the leftmost 5 of these stacking loops should have been choosen instead of D_5.

□

Approximating the MAXIMUM BASE PAIRS STACKING problem

Theorem

The MAXIMUM BASE PAIRS STACKING *problem is approximable within ratio* $8/3$.

Proof.

Step 1.

$$\frac{s_1^*}{s_1} \leq \frac{5+1+5+2}{5} = \frac{13}{5} = 2.6$$

□

Approximating the MAXIMUM BASE PAIRS STACKING problem

Theorem

The MAXIMUM BASE PAIRS STACKING *problem is approximable within ratio* $8/3$.

Proof.

Step 2.

- With all 5 consecutive stacking loops found by the first step, we can guarantee that each interval of a 2-interval D_4 (consisting of 4 consecutive stacking loops) found by the second step of our algorithm intersects at most $4 + 1$ stacking loops in S^*.

□

Approximating the MAXIMUM BASE PAIRS STACKING problem

Theorem

The MAXIMUM BASE PAIRS STACKING *problem is approximable within ratio* $8/3$.

Proof.

Step 2.

- Suppose the contrary that an interval of D_5 intersects 6 stacking loops in S^*.
- Then these 6 stacking loops must be consecutive, and hence must contain 5 consecutive stacking loops.
- This is a contradiction.

□

Approximating the MAXIMUM BASE PAIRS STACKING problem

Theorem

The MAXIMUM BASE PAIRS STACKING *problem is approximable within ratio* $8/3$.

Proof.

Step 2.

$$\frac{s_2^*}{s_2} \leq \frac{4+1+4+1}{4} = \frac{10}{4} = 2.5$$

□

Approximating the MAXIMUM BASE PAIRS STACKING problem

Theorem

The MAXIMUM BASE PAIRS STACKING *problem is approximable within ratio* $8/3$.

Proof.

Step 3.

$$\frac{s_3^*}{s_3} \leq \frac{3+1+3+1}{4} = \frac{8}{3} \simeq 2.67$$

□

Approximating the MAXIMUM BASE PAIRS STACKING problem

Theorem

The MAXIMUM BASE PAIRS STACKING *problem is approximable within ratio* $8/3$.

Proof.

Step 4.

- Each 2-interval in $\mathcal{D}$ is balanced and corresponds to either a single stacking loop (with interval length 2) ot two consecutive stacking loops (with interval length 3).
- Therefore the 2-interval graph $G(\mathcal{D})$ is 5-claw-free.

□

Approximating the MAXIMUM BASE PAIRS STACKING problem

Theorem

The MAXIMUM BASE PAIRS STACKING *problem is approximable within ratio* $8/3$.

Proof.

Step 4.

- The MAXIMUM WEIGHT INDEPENDENT SET problem in 5-claw-free graphs is approximable within ratio $5/2$.

$$\frac{s_4^*}{s_4} \leq \frac{5}{2} = 2.5$$

□

Approximating the MAXIMUM BASE PAIRS STACKING problem

Theorem

The MAXIMUM BASE PAIRS STACKING *problem is approximable within ratio* $8/3$.

Proof.

$$\frac{|S^*|}{|S|} = \frac{\sum_{i=1}^{4} s_i^*}{\sum_{i=1}^{4} s_i}$$
$$= \sum_{i=1}^{4} \frac{s_i^*}{\sum_{j=1}^{4} s_j}$$

□

Approximating the MAXIMUM BASE PAIRS STACKING problem

Theorem

The MAXIMUM BASE PAIRS STACKING *problem is approximable within ratio* $8/3$.

Proof.

$$\begin{aligned}
\frac{|S^*|}{|S|} &\leq \frac{13}{5}\left(\frac{s_1}{\sum_{j=1}^{4} s_j}\right) + \frac{10}{4}\left(\frac{s_2}{\sum_{j=1}^{4} s_j}\right) + \frac{8}{3}\left(\frac{s_3}{\sum_{j=1}^{4} s_j}\right) + \frac{5}{2}\left(\frac{s_4}{\sum_{j=1}^{4} s_j}\right) \\
&\leq \frac{8}{3}\left(\frac{s_1}{\sum_{j=1}^{4} s_j}\right) + \frac{8}{3}\left(\frac{s_2}{\sum_{j=1}^{4} s_j}\right) + \frac{8}{3}\left(\frac{s_3}{\sum_{j=1}^{4} s_j}\right) + \frac{8}{3}\left(\frac{s_4}{\sum_{j=1}^{4} s_j}\right) \\
&= \frac{8}{3}
\end{aligned}$$

□