Local RNA Motifs

Steffen Heyne

Institute of Computer Science
Bioinformatics Group
Albert-Ludwigs-University Freiburg

Bled, February 20, 2009

- miRNAs and siRNAs mediate the downregulation of gene expression
- snoRNAs modify ribosomal RNAs
- snRNAs are part of the splicing machinery
- Ribozymes, rRNAs, tRNAs,...

2008:

- Rfam 9.0 (July 2008) contains 603 RNA families
- Rfam 9.1 (Jan. 2009) contains 1372 RNA families
- massively new sequence data due to pyrosequencing

Basic analysis task: RNA comparison & homology search

Primary structure

GAAGCUGACCAGACAGUCGCCGCUUCGUCGUCGUC
CUCUUCGGGGGAGACGGGCGGAGGGAGGAAAGUC
CGGGCUCCAUAGGGCAG
GGGGGGAAACCCACGAC
ACCGCCGAUGGCCCGCG
GGUGAAAGGGUGCGGUA
CUUAUCGGUCAGUUUCA

Secondary structure

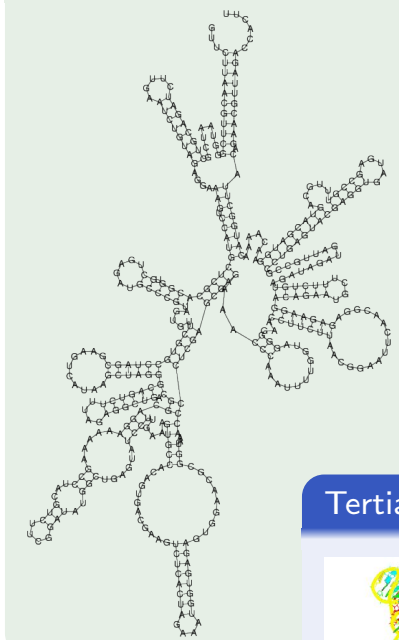
Tertiary structure

?

Primary structure

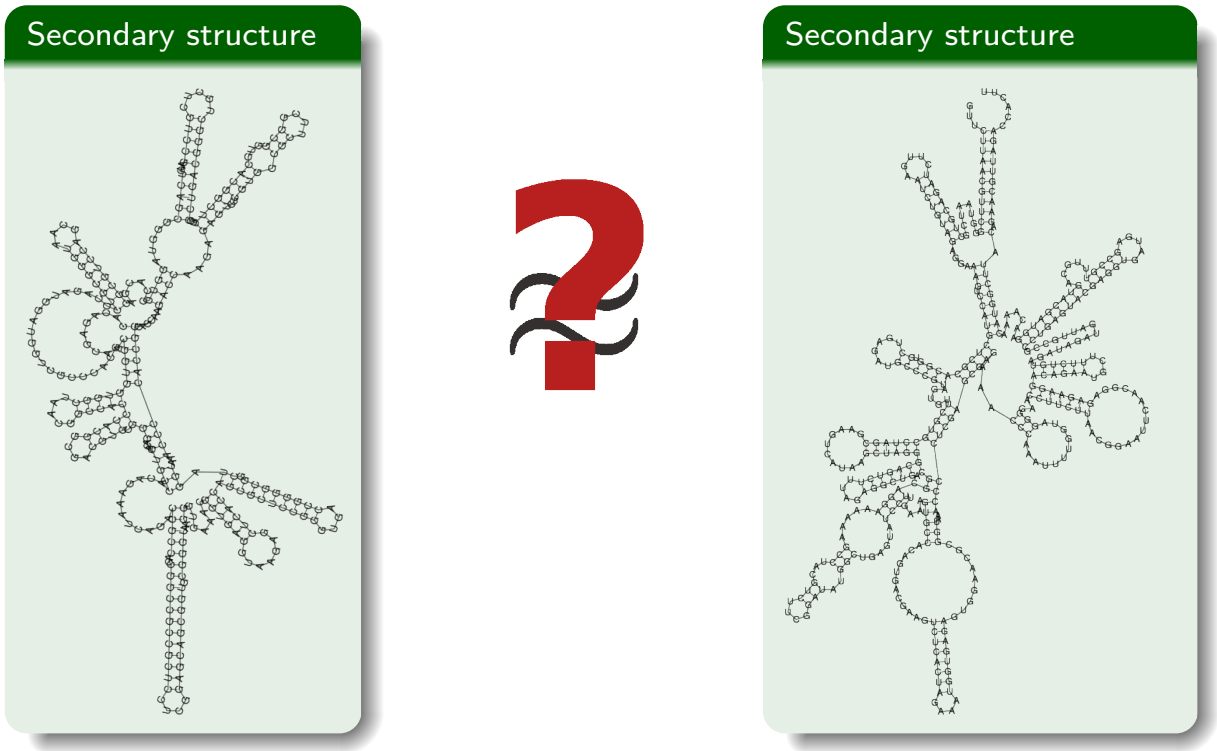
GUUCUUAACGUUCGGGUAAUCGCUGCAGAUCUUG
CCUAGCGAAGUC
GCUGACGGCAG
AUGGCUGAGUA
AAGUCUCACUA
UAAACC

Secondary structure



Tertiary structure

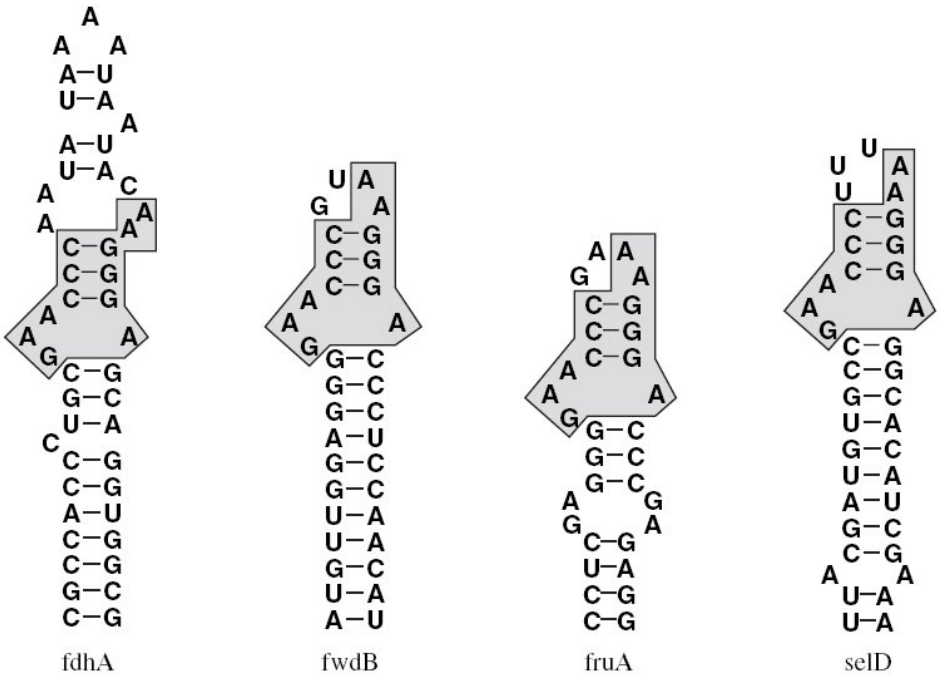
Basic analysis task: RNA comparison & homology search

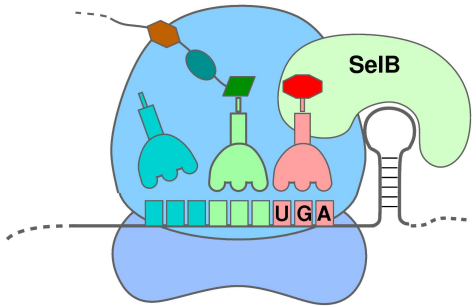


- Functional RNAs conserve structural elements
  $\Rightarrow$ Secondary structure is computationally tracktable

Examples for RNA motifs

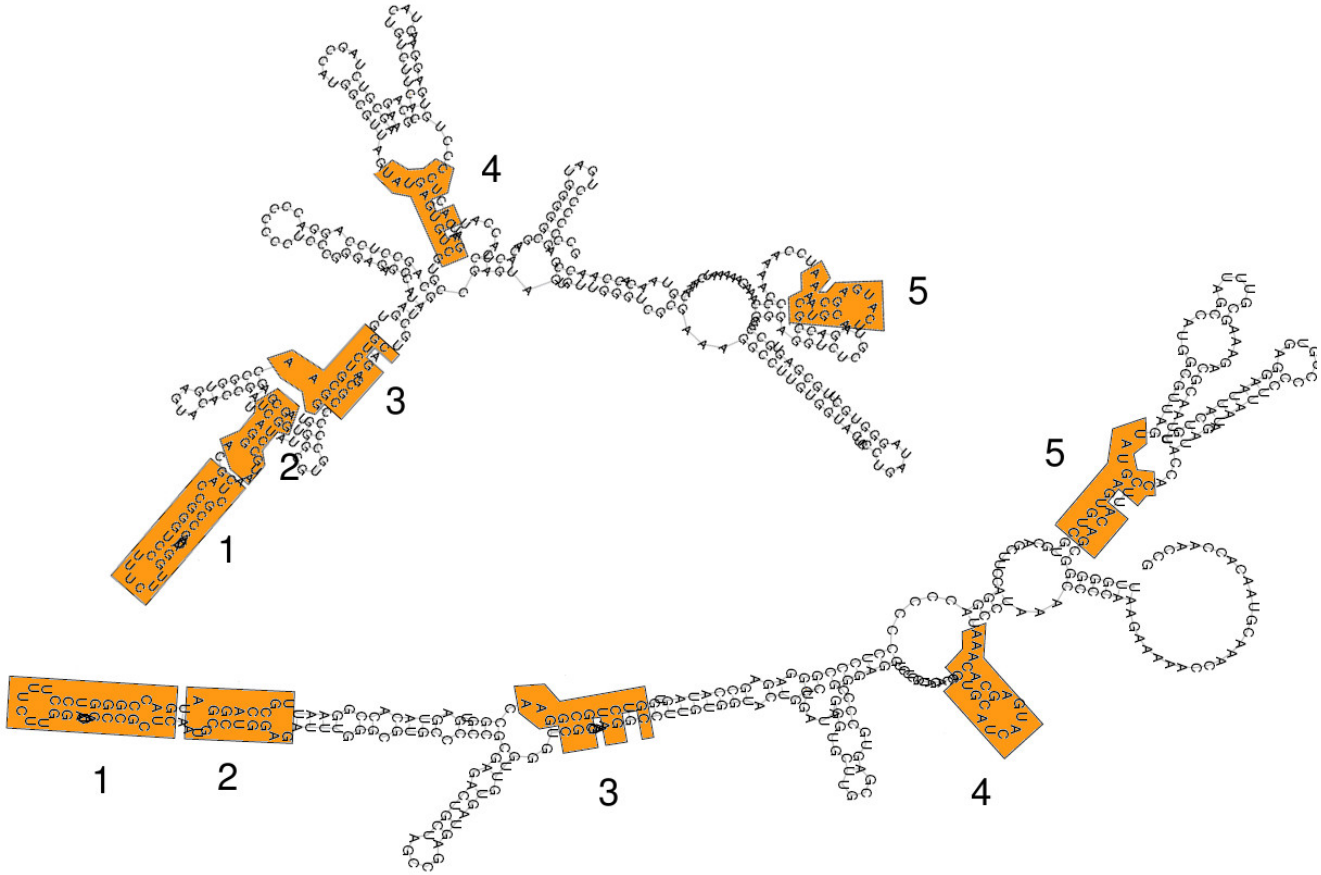
Examples for RNA motifs: SECIS elements





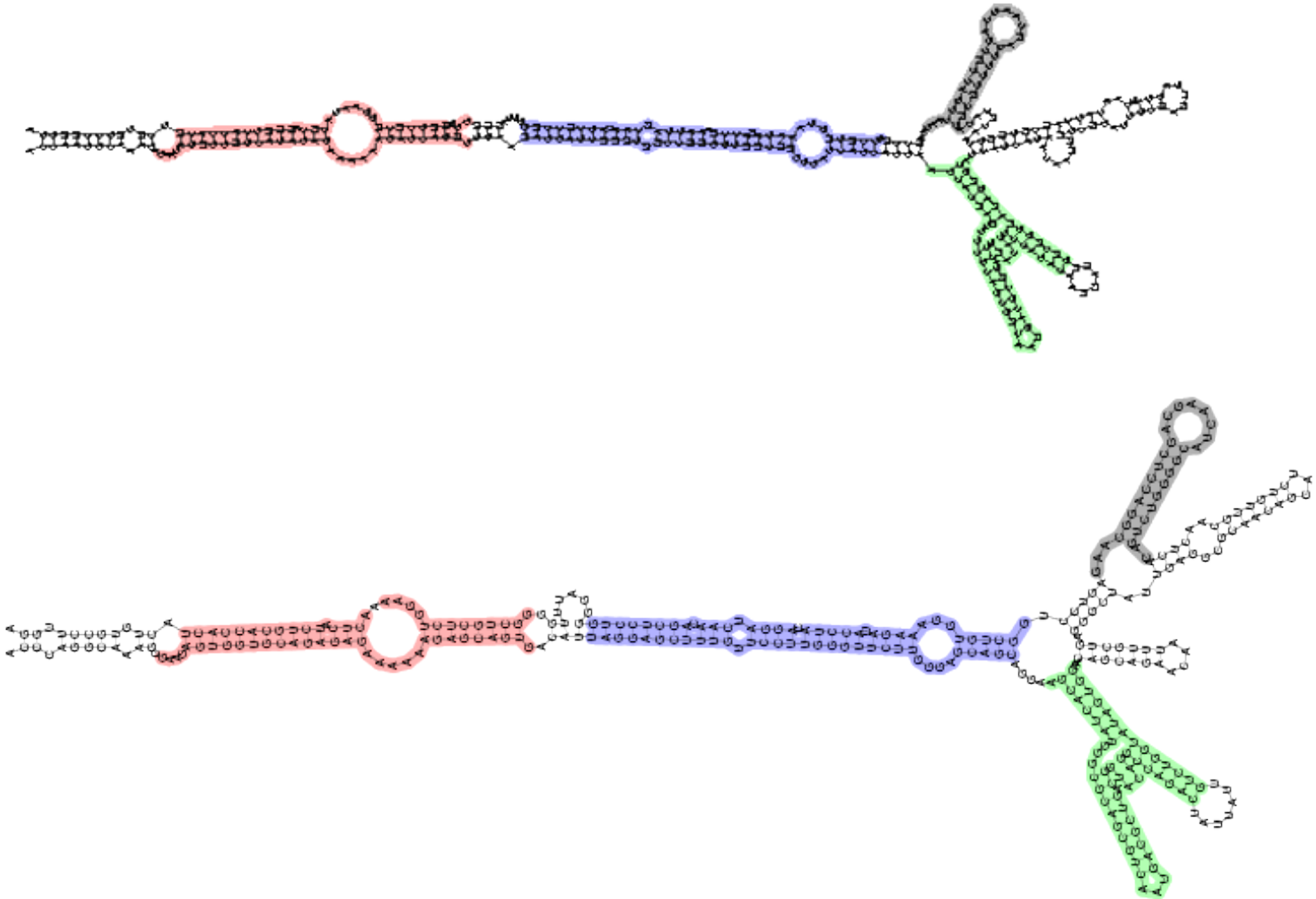
Putative SECIS elements in non-coding regions of *Methanococcus jannaschii*. (according to Wilting et al. 1997).

Examples for RNA motifs: IRES motifs in viruses



Two mfe structures of Hepatitis C Virus Internal Ribosomal Entry Sites (IRES).
The five largest exact common substrutures are indicated.

Examples for RNA motifs: Rev response element (HIV)



The four largest pattern of two HIV RREs.

RNA decay motifs in yeast:

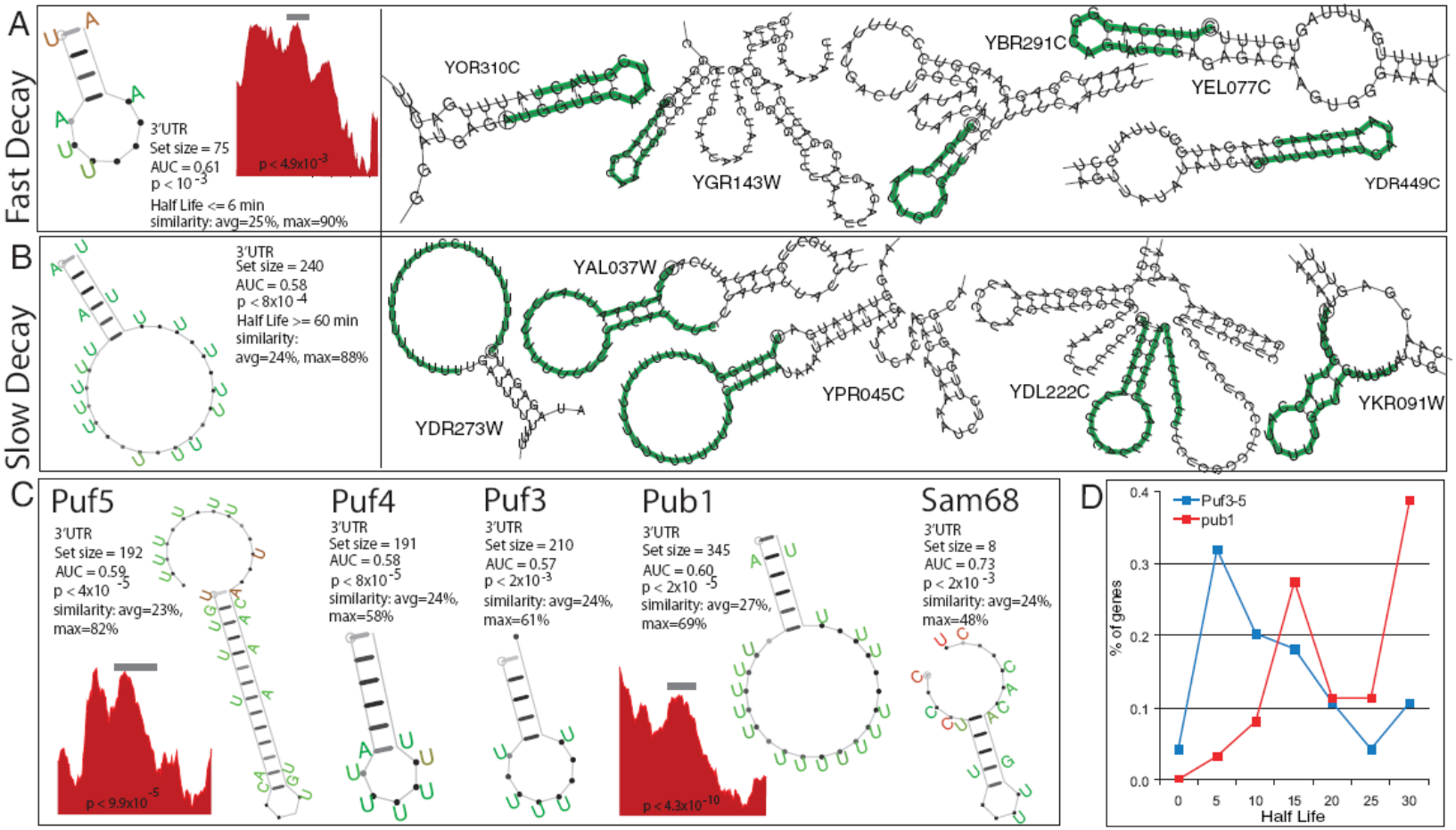


Fig. 2. mRNA decay motifs. (*A* and *B*) Fast- and slow-decaying mRNAs. Shown is the predicted motif (*Left*) and the sequence conservation profile, along with top-scoring motif examples (*Right*). The motif position is annotated in green, and the 5′ end of the motif is circled. (*C*) Predicted motifs for RBPs mRNA targets. Shown are the motif consensus and sequence conservation profile. (*D*) Half-life distribution for the top 20% of targets of the Puf proteins (blue line) and Pub1 (red line). Puf targets have faster decay rates than Pub1 targets.

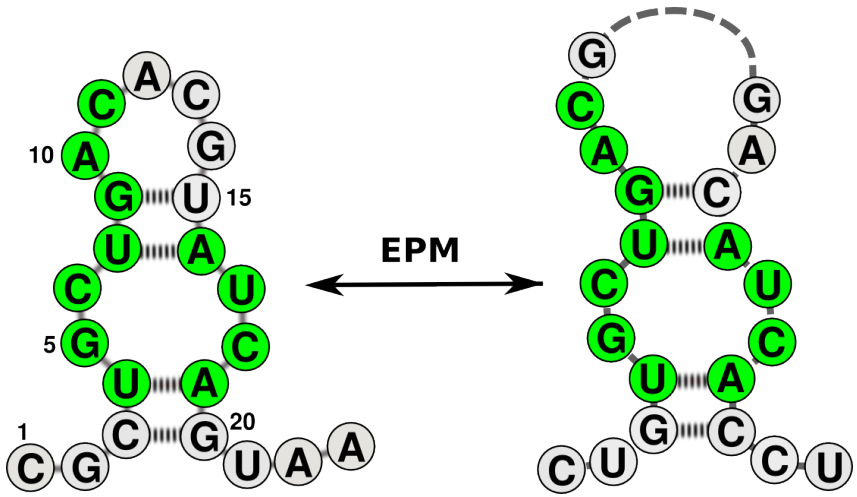
Rabani et al., 2008

Pairwise Comparison of RNA Secondary Structures with Exact Pattern Matchings

Common Substructures of two RNAs

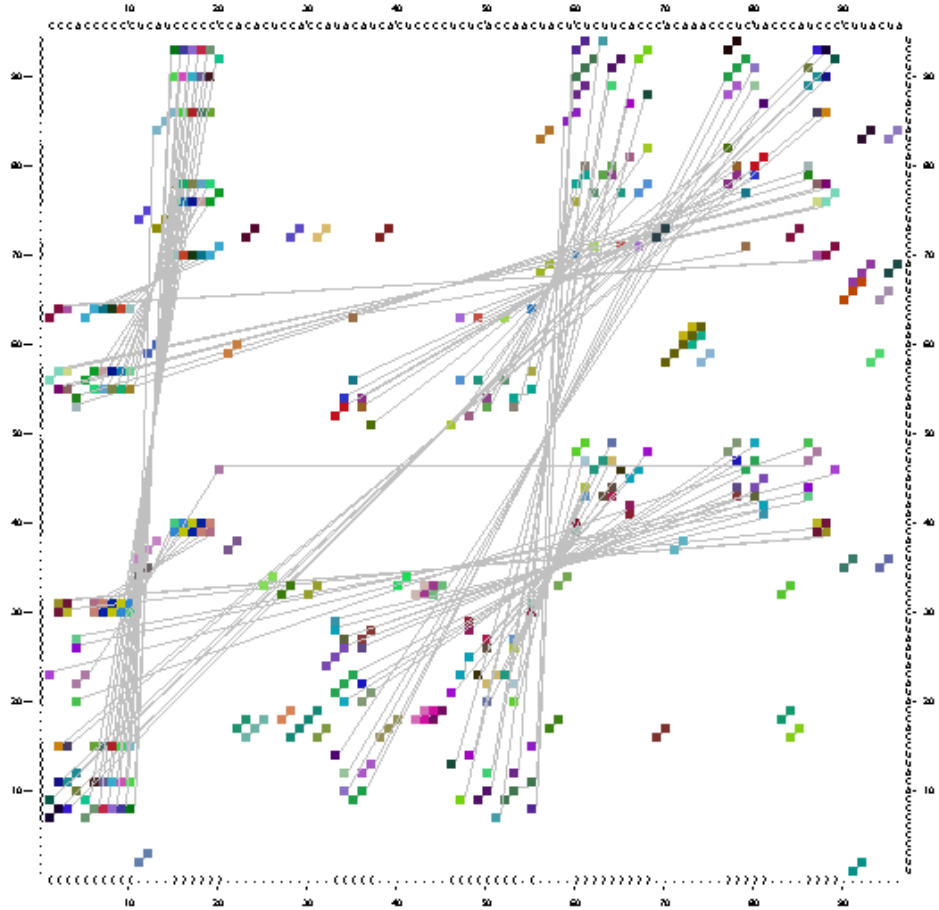
Exact Pattern Matching (EPM):

- matching between sets of positions of two RNAs, EPM $\subseteq V_1 \times V_2$
- ordered matching, i.e. for all $(p, q), (p', q') \in$ EPM it holds that $p < p'$ iff $q < q'$ and $p = p'$ iff $q = q'$.
- all pairs $(p, q) \in$ EPM consist of similar nucleotides with equal structure type
- all base pairse are preserved
- each EPM is maximal
- the matching graph is connected



First step: finding EPMs

- Backofen/Siebert algorithm[1]:
  ⇒**input:** two RNAs with primary and secondary structure
  ⇒**output:** set of all overlapping and crossing EPMs ($\mathcal{E}$), "library" denoted as $\mathbf{E}_{\gamma}^{1,2}$, fast: $O(nm)$!
- maximal $O(nm)$ different EPMs
- each pair $(p, q) \in \mathcal{E}$ is unique in $\mathbf{E}_{\gamma}^{1,2}$
- γ denotes the minimal EPM size



$\mathbf{E}_{\gamma}^{1,2}$ as dot-plot

[1] Rolf Backofen and Sven Siebert. Fast Detection of Common Sequence Structure Patterns in RNAs. Journal of Discrete Algorithms, 5(2):212-228, 2007.

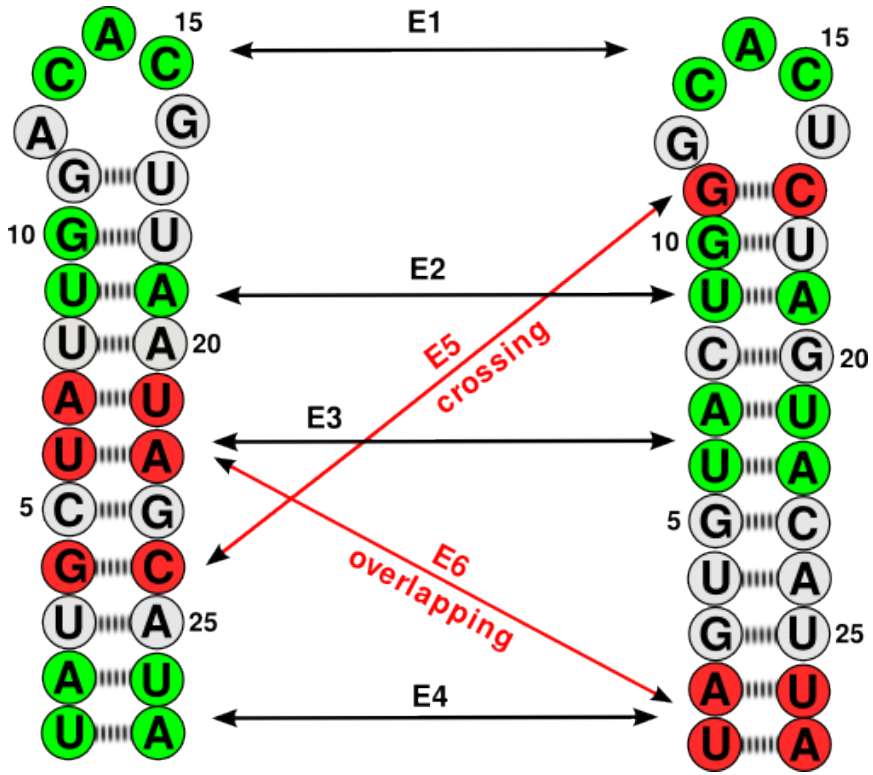
Second step:

- we want to find a "meaningful" subset of EPMs from a library $\mathbf{E}_{\gamma}^{1,2}$

  $\rightarrow$ **exclude overlapping and crossing EPMs**

  $\rightarrow$ **selected EPMs have to be nested**

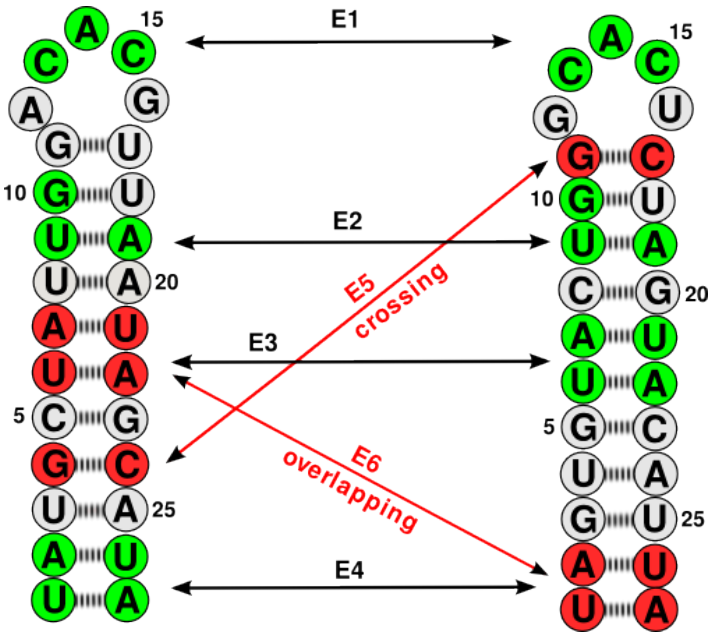
  $\rightarrow$ **our algorithm computes the maximal possible set of EPMs (LCS-EPM)**



The Longest Common Subsequence of Exact Pattern Matchings

... is the problem to find a longest common subsequence which preserves the EPMs in a library $\mathbf{E}_{\gamma}^{1,2}$ over two RNAs, i.e. finding a mapping $\mathcal{M}_{\text{EPM}} \subseteq V_1 \times V_2$ of maximal length such that:

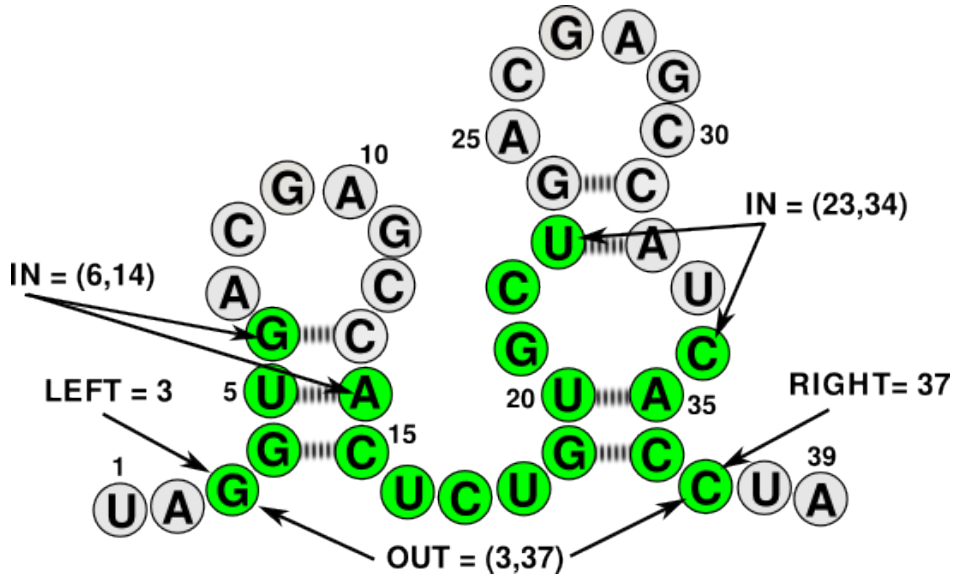
1. for each pair $(p, q) \in \mathcal{M}_{\text{EPM}}$ there exists one EPM in $\mathbf{E}_{\gamma}^{1,2}$:
   $\forall (p, q) \in \mathcal{M}_{\text{EPM}} : \exists \mathcal{E} \in \mathbf{E}_{\gamma}^{1,2}$ with $(p, q) \in \mathcal{E}$ and $\mathcal{E} \subseteq \mathcal{M}_{\text{EPM}}$
2. $\mathcal{M}_{\text{EPM}}$ is a bijective mapping and preserves the order of the nucleotides:
   $\forall (p, q), (p', q') \in \mathcal{M}_{\text{EPM}} : p = p' \iff q = q'$, $p < p' \iff q < q'$



Note: $\mathcal{M}_{\text{EPM}}$ is an arc-preserving subsequence!

Bounds and holes:

given EPM $\mathcal{E} \in \mathbf{E}_{\gamma}^{1,2}$
of size $k = 18$:

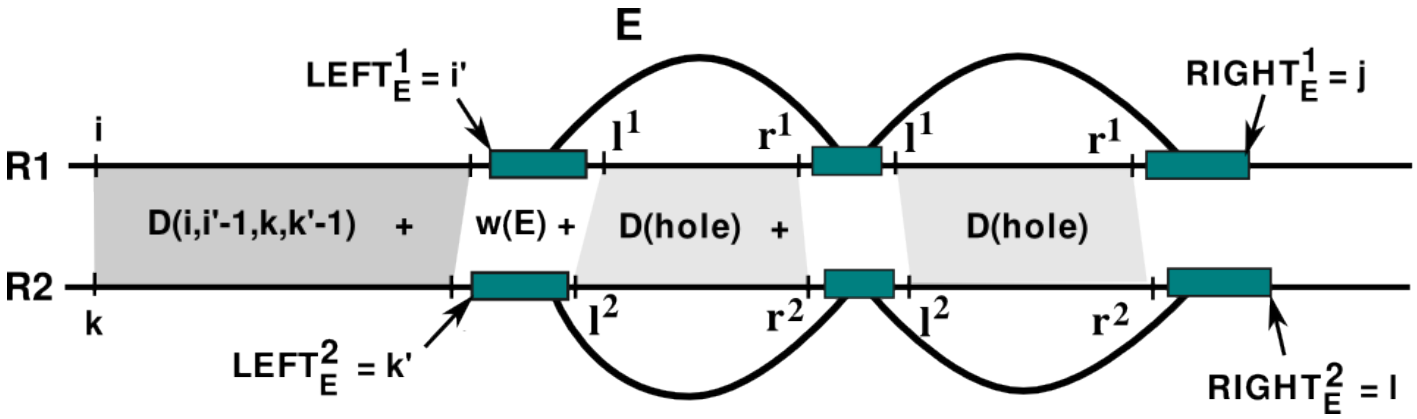


pattern $\mathcal{P}_1 = \langle \mathbf{3}, 4, 5, \mathbf{6}, \mathbf{14}, 15, 16, 17, 18, 19, 20, 21, 22, \mathbf{23}, \mathbf{34}, 35, 36, \mathbf{37} \rangle$ in $\mathcal{R}_1$
pattern $\mathcal{P}_2 = \langle q_1, q_2, ..., q_k \rangle$ in $\mathcal{R}_2$

left-outside-bounds: $\text{LEFT}_{\mathcal{E}} = (3, q_1)$, $\text{LEFT}_{\mathcal{E}}^1 = 3$
right-outside-bounds: $\text{RIGHT}_{\mathcal{E}} = (37, q_{18})$, $\text{RIGHT}_{\mathcal{E}}^1 = 37$
inside-bounds: $\text{IN}_{\mathcal{E}} = \{\langle (6,14), (q_4, q_5) \rangle, \langle (23,34), (q_{14}, q_{15}) \rangle\}$
holes: $\text{HOLES}_{\mathcal{E}}^1 = \{\langle 7, 13 \rangle, \langle 24, 33 \rangle\}$

Dynamic programming algorithm to solve LCS-EPM:

$D(i,j,k,l) =$

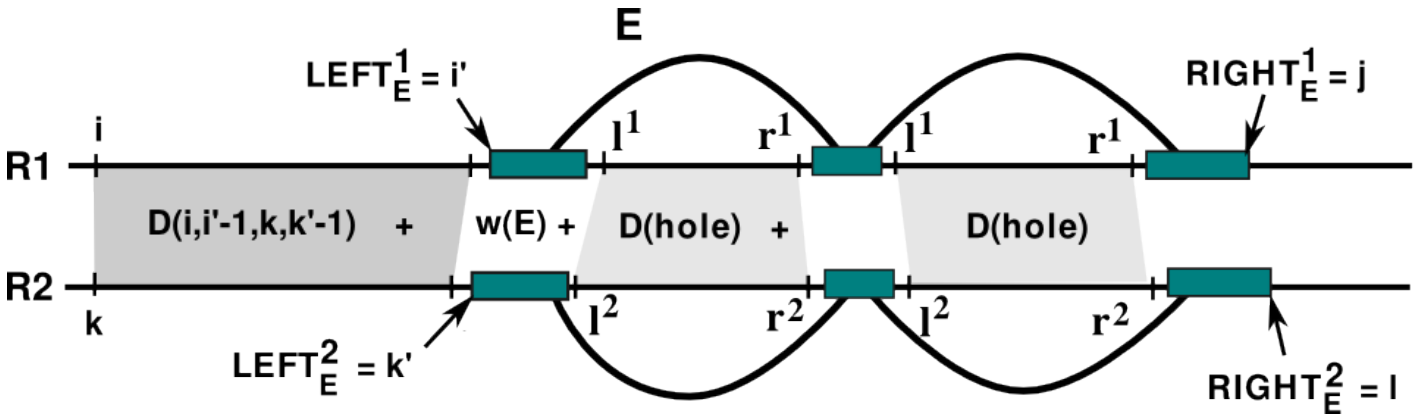


1. treat each EPM only at its right-outside-bounds RIGHT$_\varepsilon$
2. compute score of each "hole" recusively and combine it with score before left-outside-bounds LEFT$_\varepsilon$

$\longrightarrow$ would lead to a 4-dimensional matrix

Dynamic programming algorithm to solve LCS-EPM:

$D(i,j,k,l) =$



1. treat each EPM only at its right-outside-bounds RIGHT$_\mathcal{E}$
2. compute score of each "hole" recusively and combine it with score before left-outside-bounds LEFT$_\mathcal{E}$

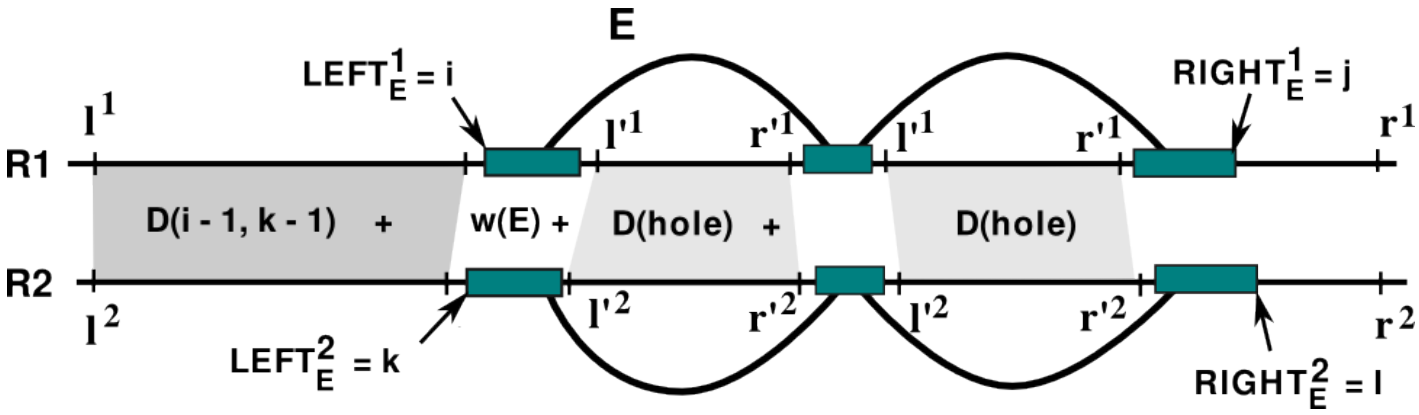
$\longrightarrow$ would lead to a 4-dimensional matrix

Better idea:

1. sort holes according to their size: $h_i \preceq_{\text{HOLES}^1} h_j \iff (r_i^1 - l_i^1) \leq (r_j^1 - l_j^1)$

Dynamic programming algorithm to solve LCS-EPM:

$D(j, l) \quad =$



1. treat each EPM only at its right-outside-bounds RIGHT$_\varepsilon$
2. compute score of each "hole" recusively and combine it with score before left-outside-bounds LEFT$_\varepsilon$

$\longrightarrow$ would lead to a 4-dimensional matrix

Better idea:

1. sort holes according to their size: $h_i \preceq_{\text{HOLES}^1} h_j \Longleftrightarrow (r_i^1 - l_i^1) \leq (r_j^1 - l_j^1)$
2. start computing with the smallest hole $\rightarrow$ no uncomputed holes inside $\rightarrow$ left ends of current hole is fixed $\rightarrow$ only 2-dimensional matrix

Dynamic programming algorithm to solve LCS-EPM:

Complexity

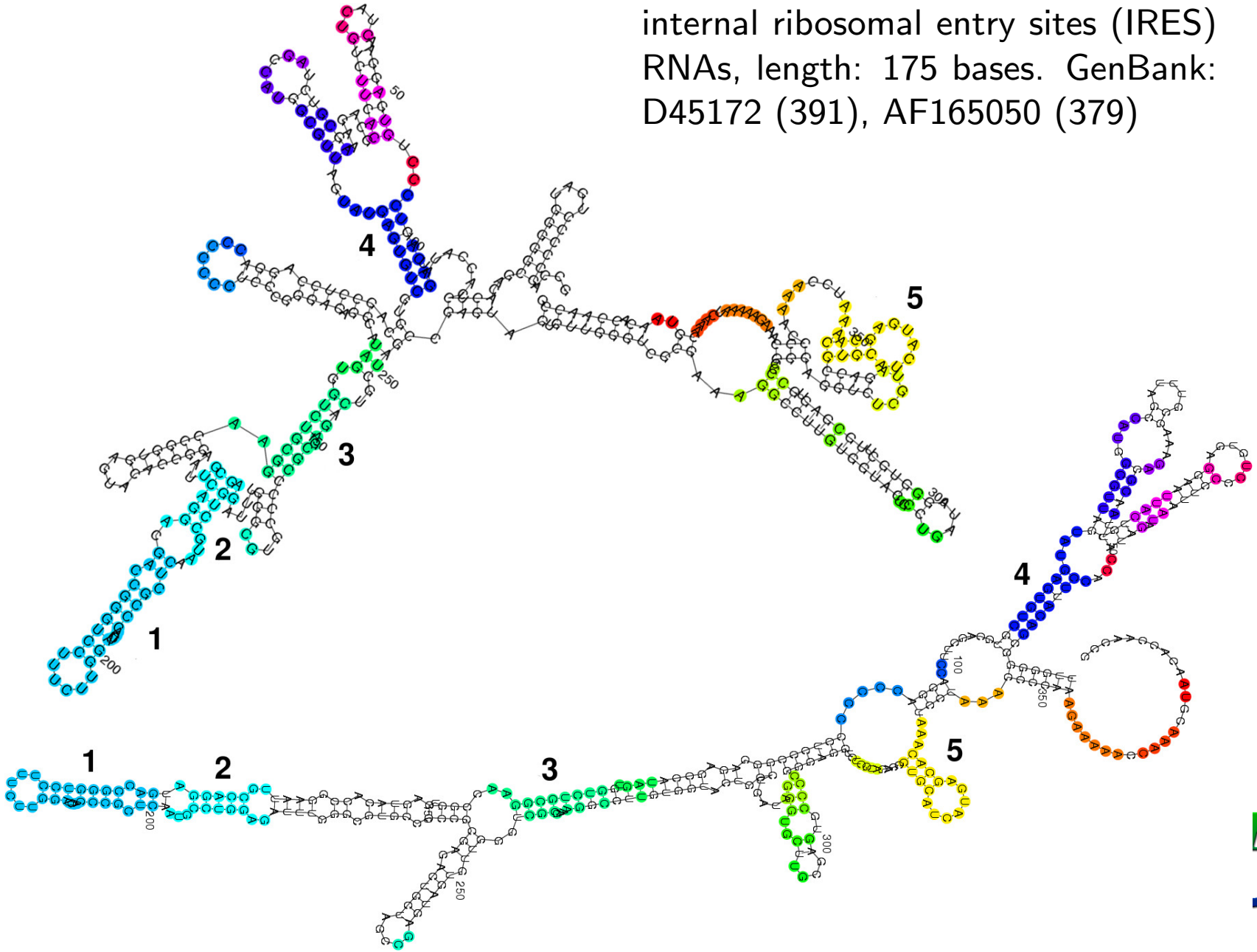
- maximal $O(nm)$ different holes in $\mathbf{E}_{\gamma}^{1,2}$
- $O(nm)$ matrix for each hole
- $\Rightarrow$ worst case complexity: $O(n^2m^2)$ time and $O(nm)$ space
- $\Rightarrow$ "real case": $O(H \cdot nm)$ time since $H \ll n \cdot m$ (H: #holes)

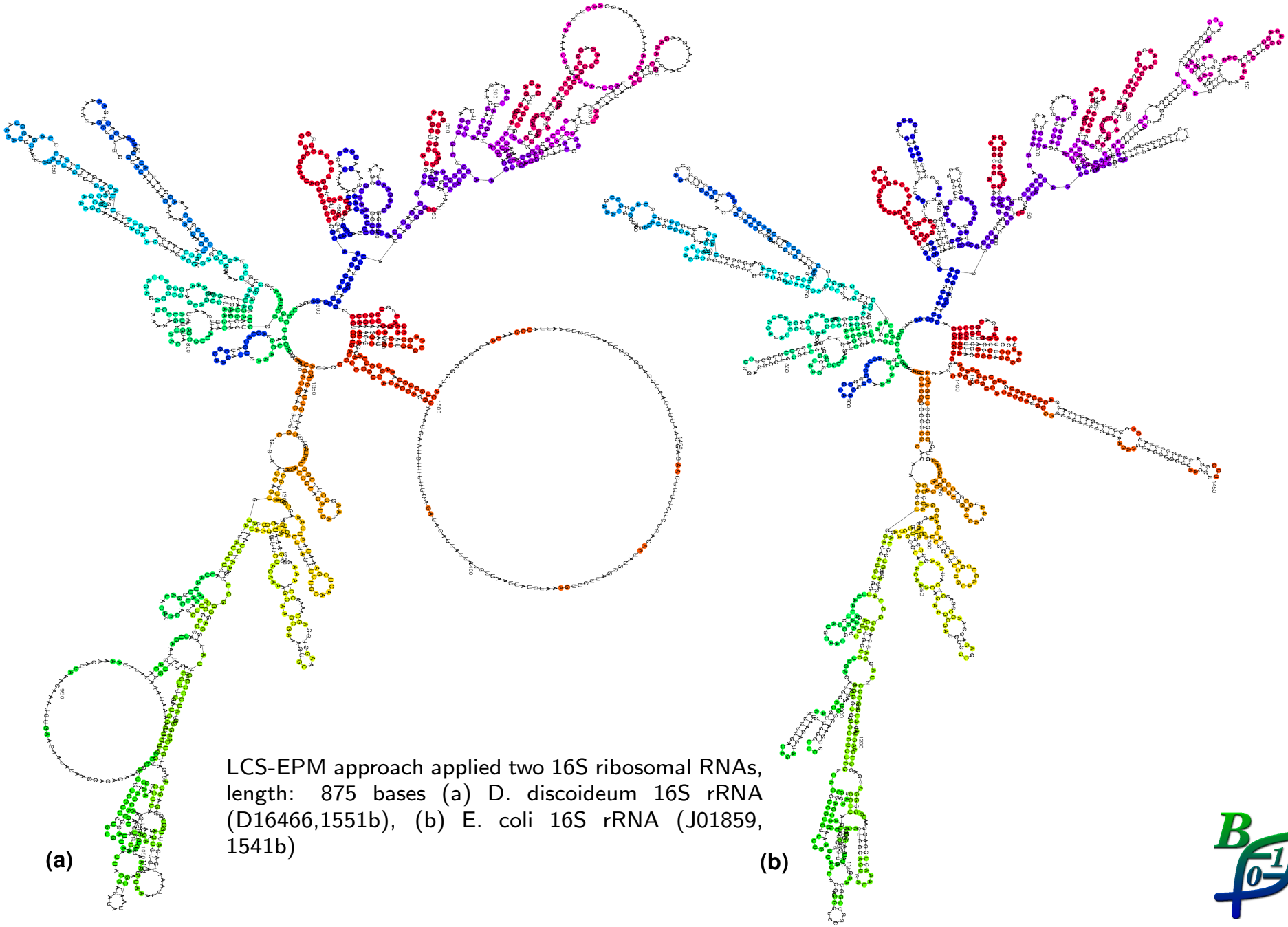
Results

Application scenarios:

1. Comparative RNA sequence analysis
2. Filtering/ speedup method for expensive Sankoff-style RNA alignments
   $\Rightarrow$ here common subtructures from LCS-EPM are used as "structural" anchor constraints

LCS-ERP for two Hepatitis C virus internal ribosomal entry sites (IRES) RNAs, length: 175 bases. GenBank: D45172 (391), AF165050 (379)

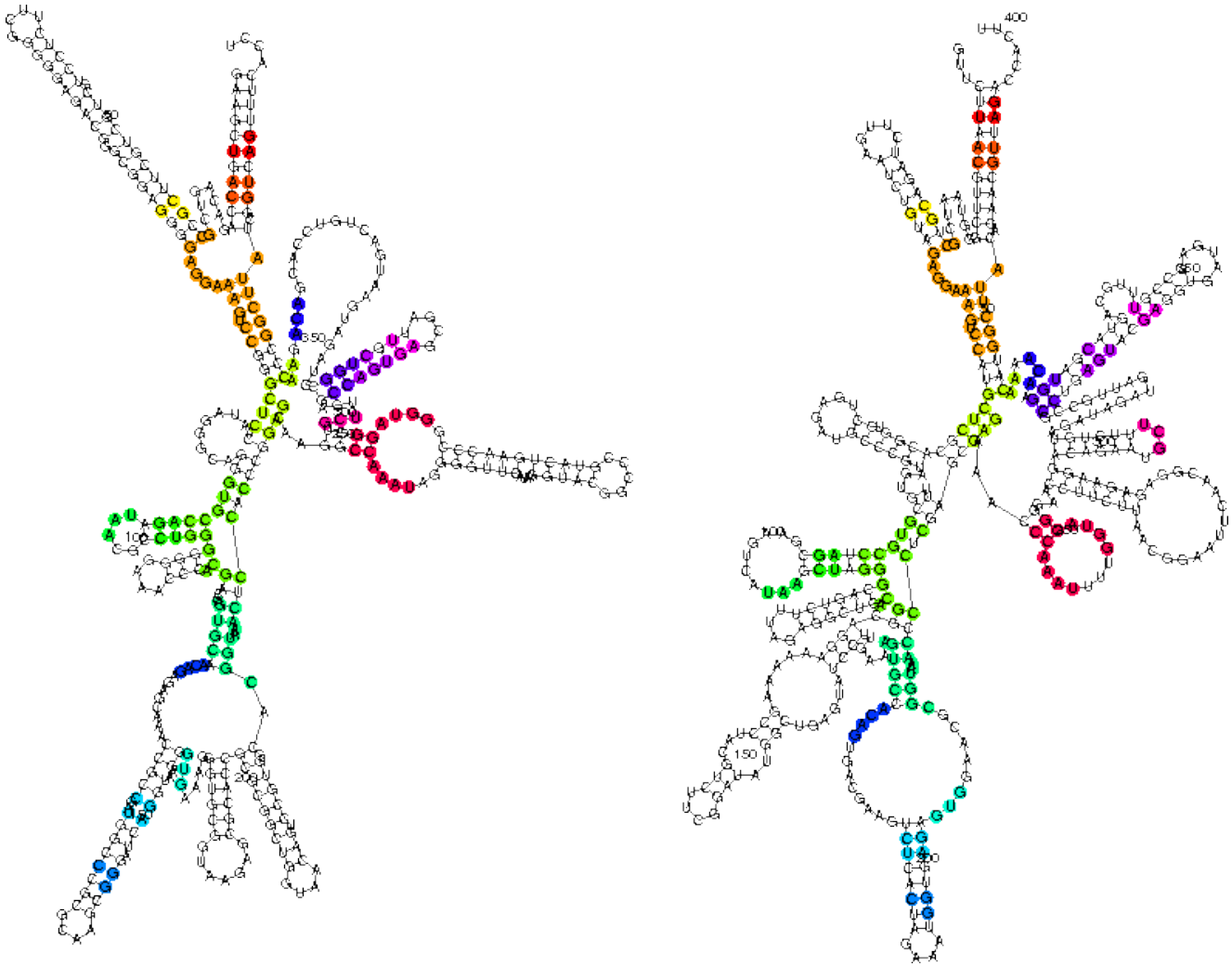




LCS-EPM approach applied two 16S ribosomal RNAs, length: 875 bases (a) D. discoideum 16S rRNA (D16466,1551b), (b) E. coli 16S rRNA (J01859, 1541b)

(a)

(b)



LCS-EPM for two RNAse P RNAs, length: 117, $\gamma = 3$

left: E.coli (A-type, 378nt), right B. subtilis (B-type, 402nt)

Comparison with other alignment methods:

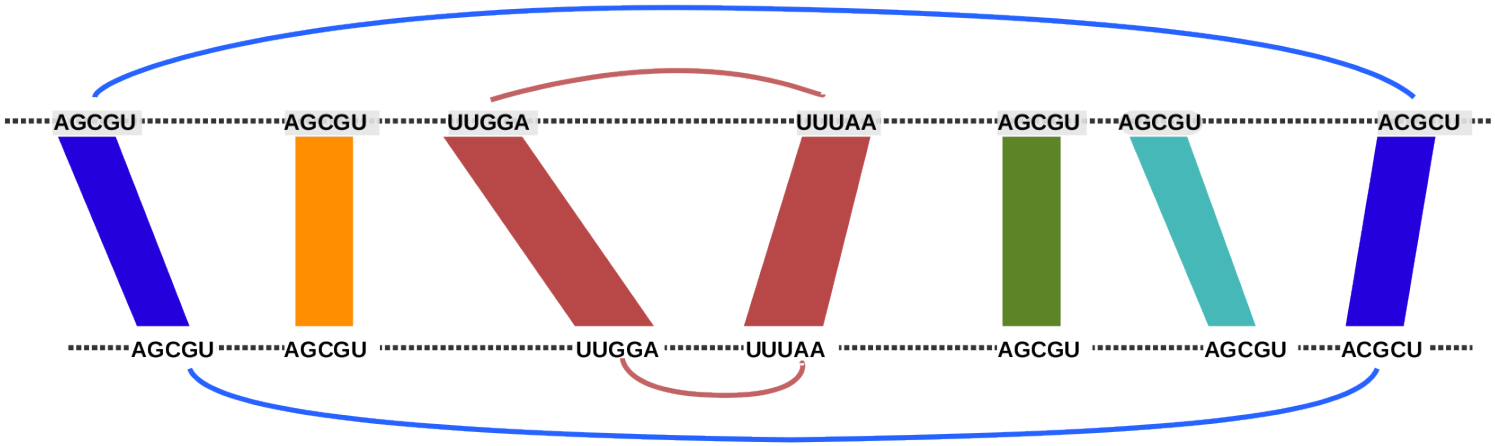
	IRES RNAs			16S rRNAs		
	#matches	coverage	time	#matches	coverage	time
ExpaRNA	175	45%	0.97*s*	875	57%	16.9*s*
RNA_align	192	50%	62.1*s*	861	56%	1*h* 35*m*
RNAforester	128	33%	5.41*s*	847	55%	7*m* 25*s*

comparison	IRES RNAs #common matches	16S rRNAs #common matches
ExpaRNA & RNA_align	159 (82.8%)	688(79.9%)
ExpaRNA & RNAforester	103 (80.5%)	700(82.6%)

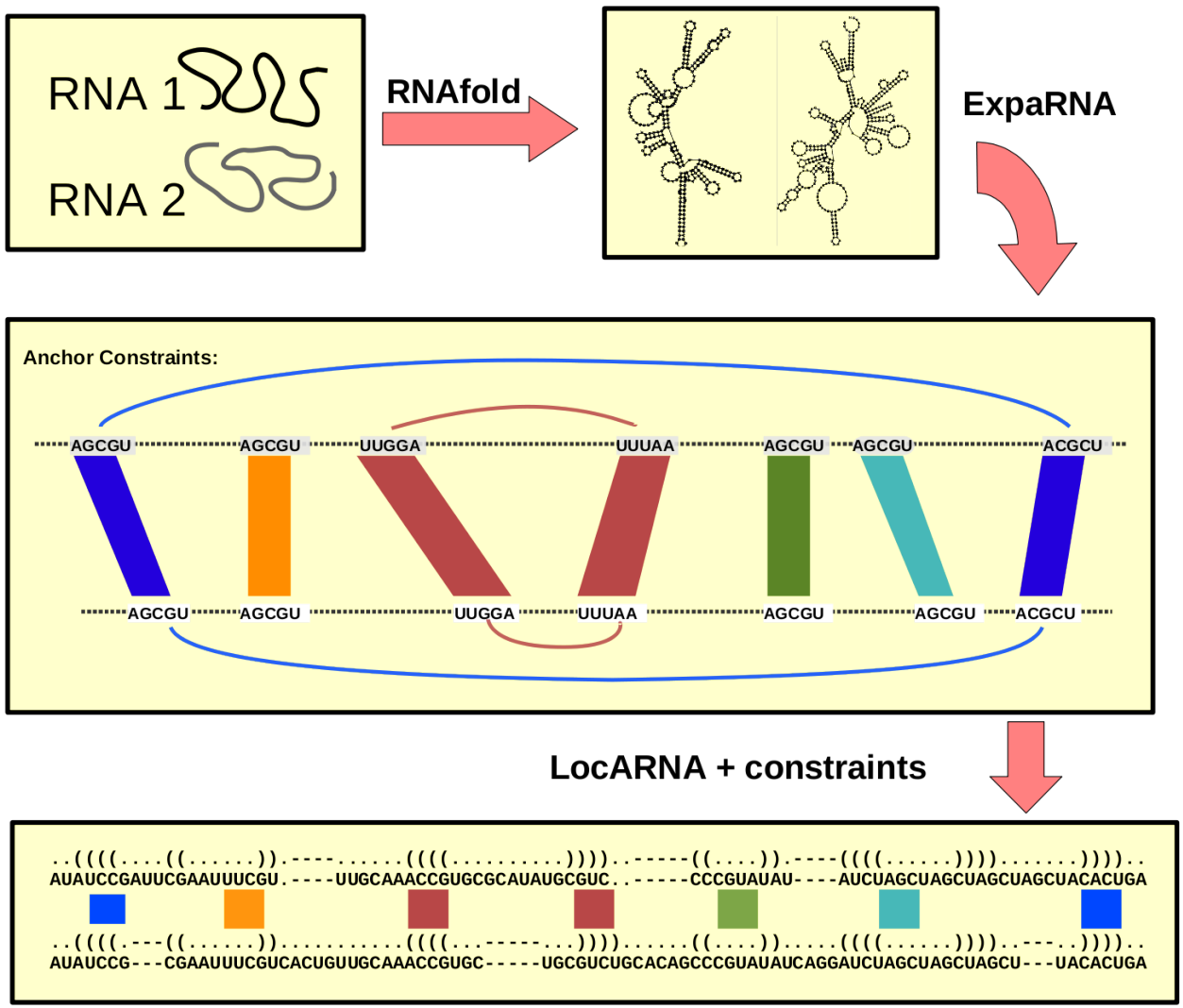
Table: Comparison of the number of found exact matchings by LCS-EPM and two alignment methods. *#common matches* defines the number of identical matched nucleotides of ExpaRNA and the other methods. (AMD Opteron 2.2 GHz 20GB)

Speeding-up Sankoff-style alignment methods with anchor contraints predicted by ExpaRNA

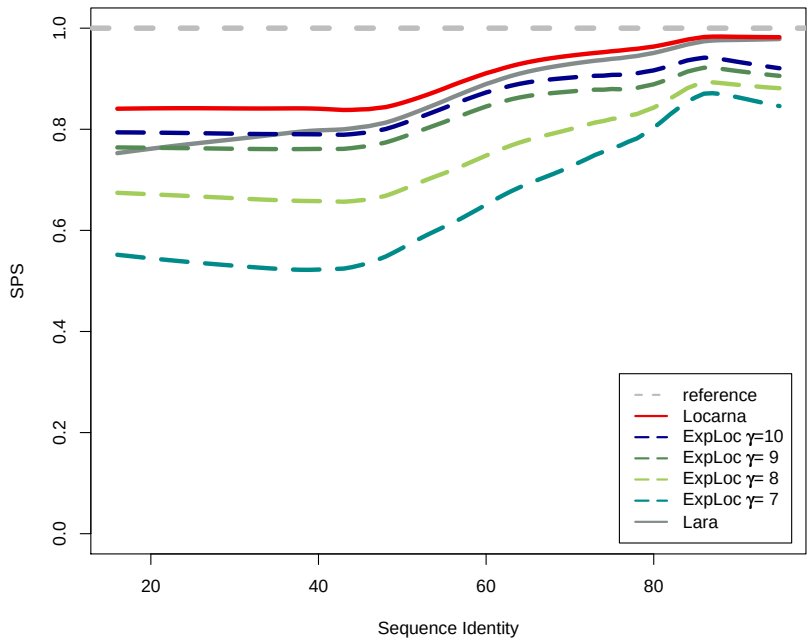
- Idea: use LCS-EPM as "structral" anchor constraints
  $\Rightarrow$ predicted anchors restrict search space of a full alignment algorithm



Speeding-up LocARNA with anchor contraints predicted by ExpaRNA



Speeding-up LocARNA with anchor contraints predicted by ExpaRNA



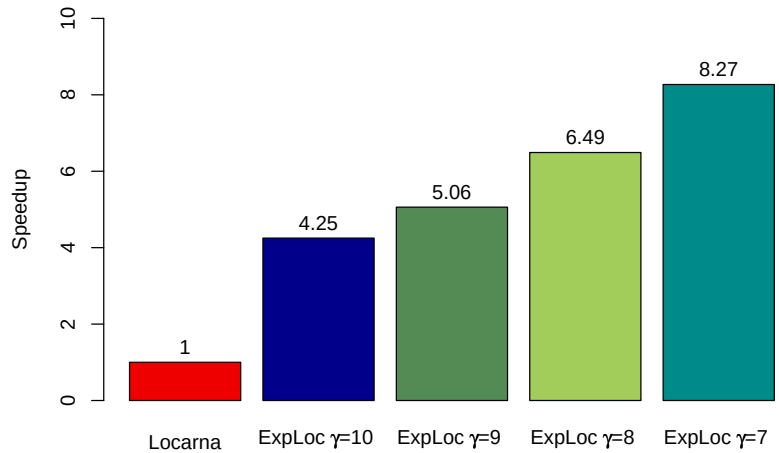
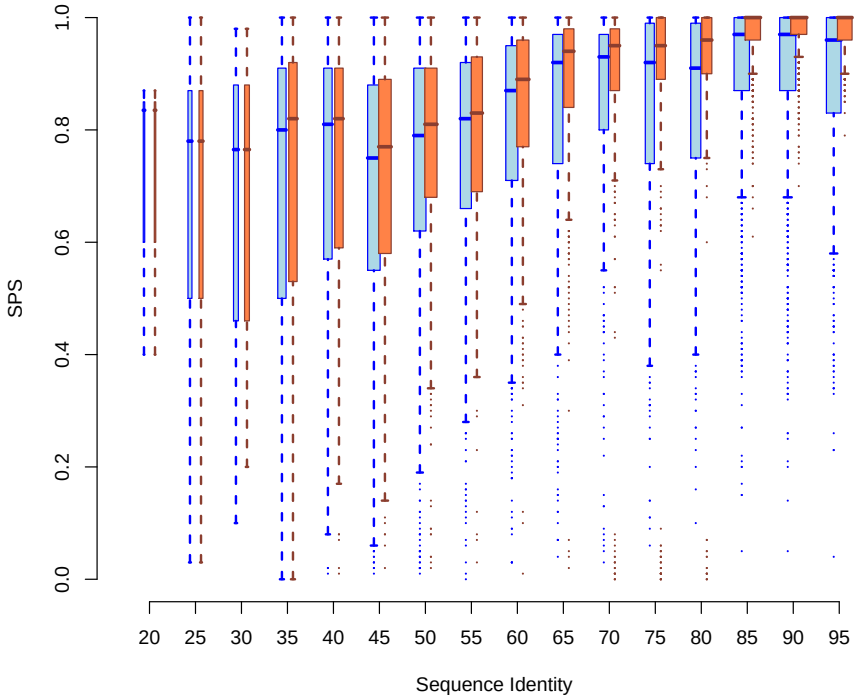
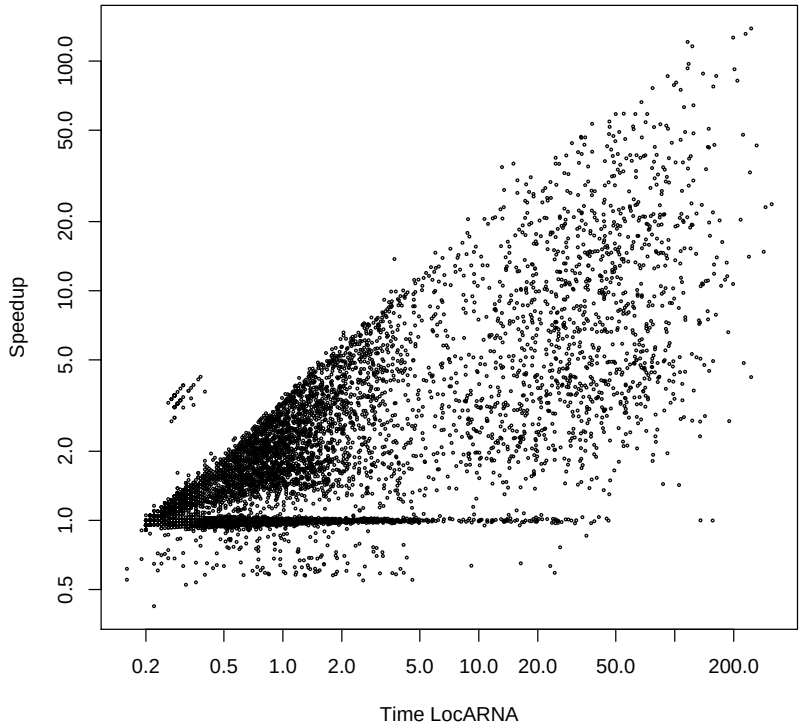


Figure: (left) Obtained alignment qualities for different minimal EPM sizes γ in comparison to LocARNA and Lara on Bralibase 2.1 k2 dataset (8976 alignments). (right) Achieved speedup of ExpLoc with respect to LocARNA running time when using different minimal EPMsizes γ. Total times were measured for both methods when applied to all alignments of the Bralibase 2.1 k2 dataset.

Speeding-up LocARNA with anchor contraints predicted by ExpaRNA





Summary:

- Common substructures useful for comparative RNA analysis
- Prefiltering for time-consuming sequence-structure comparison methods (use LCS-EPM as anchor constraints for a gapped global alignment)

Drawbacks:

- `ExpaRNA` is based on a fixed secondary structure

Improvements and future work:

- improved scoring function (ensemble probabilities/ accuracy)
- Gap cost function, local version

Tool available: `ExpaRNA` (www.bioinf.uni-freiburg.de/Software)

Publ.: Heyne et al., Bioinformatics, 2009

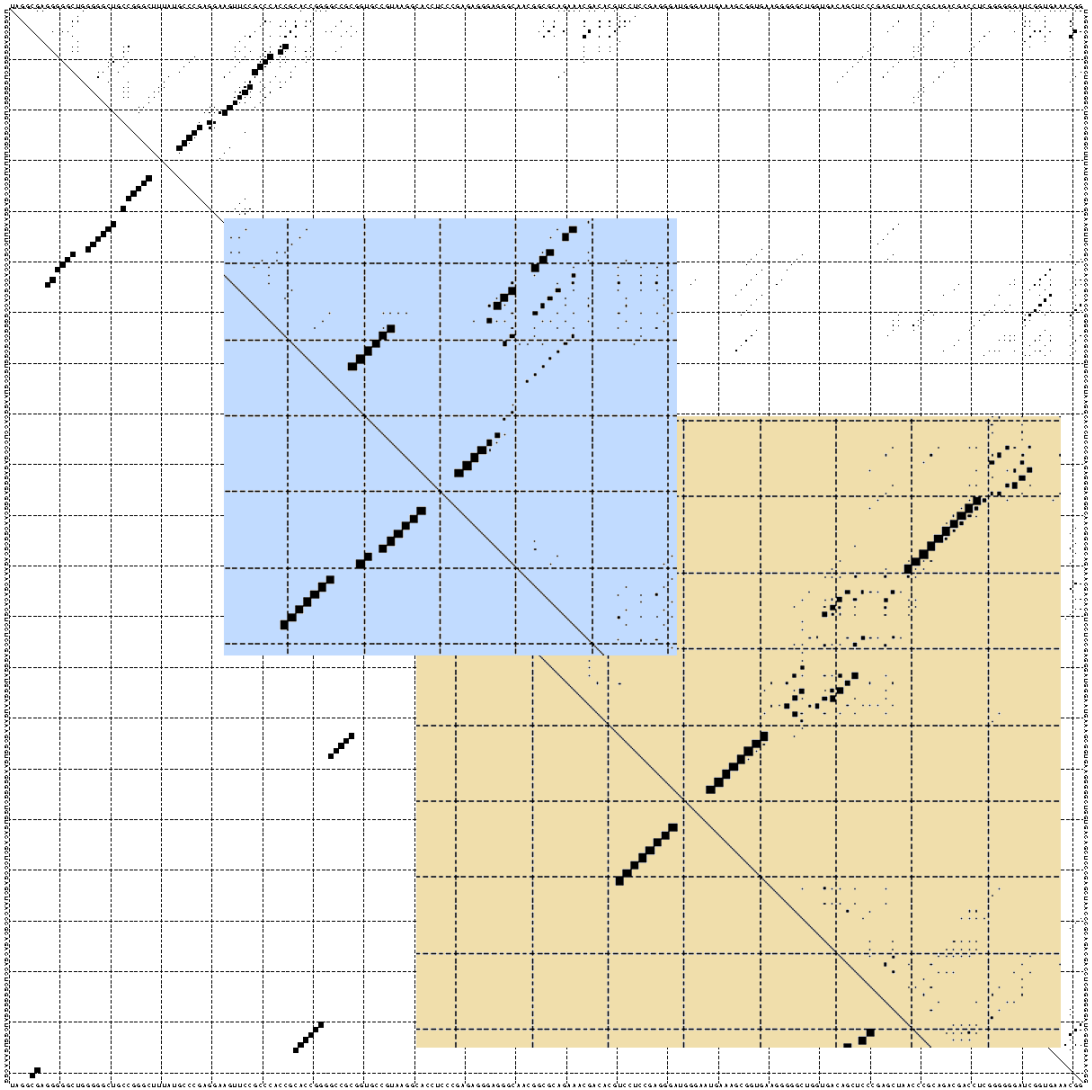
Finding local RNA Motifs

Motivation:

- RNA sequence-structure alignment methods not applicable in a genome wide scale
- Local RNA motifs usful as anchor constraints, but a fixed (mfe) secodary structure is not realistic (mRNAs, ...)
- In mRNAs probably exist many different RNA motifs for protein binding, colocalization etc.

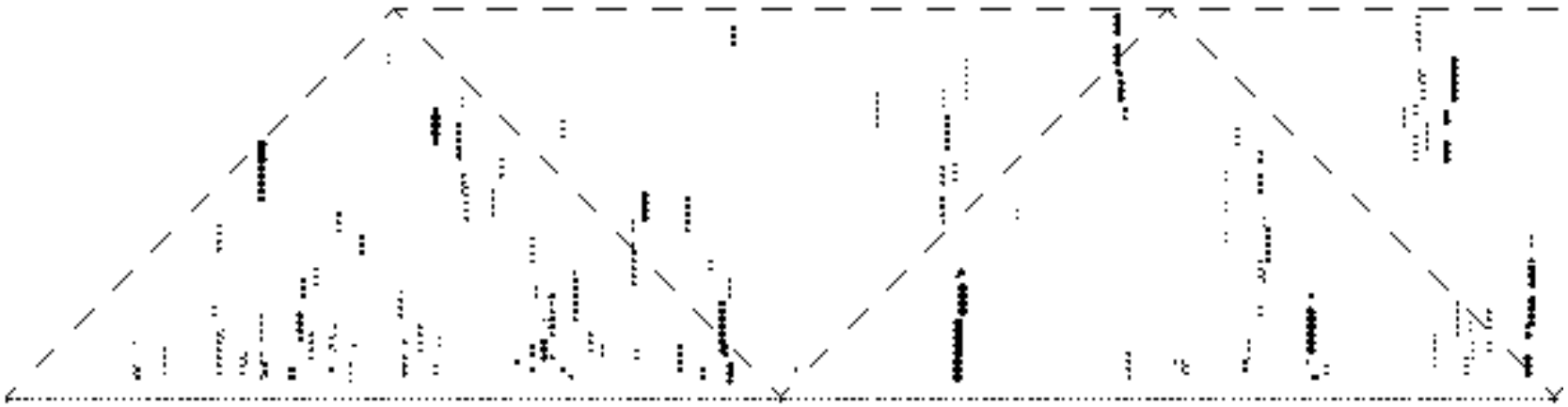
$\Rightarrow$ **How to find local, but highly probable RNA motifs?**

Problem: alternative structures



Pyrococcus furiosus RNase P RNA

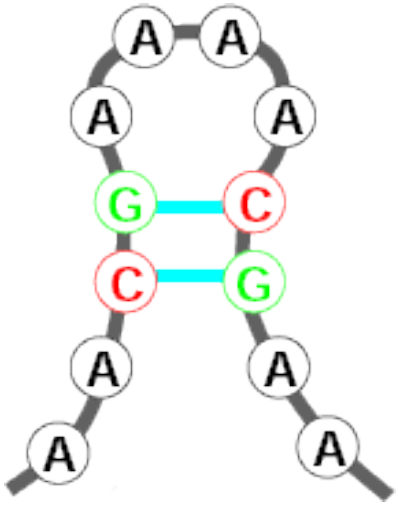
Problem: genome wide RNA analysis



RNAplfold dot-plot, part of Synechococcus PCC7002 genome

Probability of Substructures:

- Product of probability of its elements
- Example: 2 base pairs with $P(i,j) = 0.8, P(i',j') = 0.7$ $\Rightarrow P((i,j) \wedge (i',j')) = 0.56$
- Problem: the more elements you add, the lower is the probability of the substructure



Idea:

- Sum up probabilities of a substructure with recursion:

$$S_{i,j} = \max \begin{cases} S_{i-1,j-1} & + P^{bp}(i,j) \\ S_{i+1,j} & + \alpha \cdot P^{ss}(i) \\ S_{i,j-1} & + \alpha \cdot P^{ss}(j) \end{cases}$$

- Overall score of motif $S_{i,j}$ is defined via its accuracy:

$$Acc = \frac{S_{i,j}}{\delta + length(S_{i,j})^{deg}}$$

- Start value: δ, degression: deg
- Traceback returns best and suboptimal motifs
- Branching motifs are possible

Current status:

- Draft implementation in perl
- Works with RNAfold, RNAalifold and RNAplfold dotplots
- Structural local motifs also possible (incomplete subsequences)

Index-Based Fast Search of RNAs

Motivation:

- Massively increasing sequence data (deep sequencing)
- Matching/Searching within large sequence databases: BLAST
  $\Rightarrow$ but nothing similar for RNAs
- Finding similar RNAs below 75% APSI: sequence-structure alignment methods necessary
- Sophisticated methods exists (CompAlign, CMfinder, LocARNA, ...), but they are computationally very expensive

Idea:

- Index-based fast search engine
- Data structure: **Affix-Array**
  $\Rightarrow$ index data structure that allows efficient solving of bidirectional search problems (extension to the left AND right)
  $\longrightarrow$ suffix-based index structure allows only left-to-right pattern matching

[2] "The affix array data structure and its applications to RNA secondary structure analysis.", Theor. Comput. Sci. 389(1-2): 278-294, 2007)

Idea:

- Index-based fast search engine
- Data structure: **Affix-Array**
  $\Rightarrow$ index data structure that allows efficient solving of bidirectional search problems (extension to the left AND right)
  $\longrightarrow$ suffix-based index structure allows only left-to-right pattern matching
- Affix array: Suffix array + reverse prefix array + link table
- Affix arrays are as powerful as affix trees, but consume less memory (16-18 bytes per input symbol)

[2] "The affix array data structure and its applications to RNA secondary structure analysis.", Theor. Comput. Sci. 389(1-2): 278-294, 2007)

Idea:

- Index-based fast search engine
- Data structure: **Affix-Array**
  $\Rightarrow$ index data structure that allows efficient solving of bidirectional search problems (extension to the left AND right)
  $\longrightarrow$ suffix-based index structure allows only left-to-right pattern matching
- Affix array: Suffix array + reverse prefix array + link table
- Affix arrays are as powerful as affix trees, but consume less memory (16-18 bytes per input symbol)
- Matching engine: first available implementation of theoretical ideas presented by Dirk Strothmann[2], C++ library implemented by Michael Beckstette and Fernando Meyer

[2] "The affix array data structure and its applications to RNA secondary structure analysis.", Theor. Comput. Sci. 389(1-2): 278-294, 2007)

Approach:

1. Extract conserved motifs from an RNA family (alignment)

$$\Downarrow$$

2. Generate descriptors for search engine (wildcards allowed, 2-3 bases specified)

Approach:

1. Extract conserved motifs from an RNA family (alignment)

$$\Downarrow$$

2. Generate descriptors for search engine (wildcards allowed, 2-3 bases specified)

$$\Downarrow$$

3. Get matchings in database from index-based search engine

Approach:

1. Extract conserved motifs from an RNA family (alignment)

$$\Downarrow$$

2. Generate descriptors for search engine (wildcards allowed, 2-3 bases specified)

$$\Downarrow$$

3. Get matchings in database from index-based search engine

$$\Downarrow$$

4. Take all regions where mathings occur in similar ordering (chaining)

Approach:

1. Extract conserved motifs from an RNA family (alignment)

$$\Downarrow$$

2. Generate descriptors for search engine (wildcards allowed, 2-3 bases specified)

$$\Downarrow$$

3. Get matchings in database from index-based search engine

$$\Downarrow$$

4. Take all regions where mathings occur in similar ordering (chaining)

$$\Downarrow$$

5. Align region against original RNA family with expensive methods like CompAlign, LocARNA, ...

Examples:

Descriptor	Time in ms (393MB)* Index-based	Naïve
********A**CUG*C******** ((((.(((.....))).))))	197,3209	5.296,7504
GG**A**CUG*C*******C ((((.(((.....))).))))	131,8251	5.299,6662
GG*GGGACUG*CCCU*CC*C ((((.(((.....))).))))	13,8771	5.297,3132
GUGGCGGGAAACUGGCCCUCCCAC ((((.(((.....))).))))	0,2655	4.709,6669

Database: ftp://ftp.ncbi.nih.gov/genomes/H_sapiens/RNA/rna.fa.gz and random subsets, AMD Opteron 2,6 GHz, 64 GB

Open Questions

- How to define good descriptors? What are they based on (Rfam seed alignments, RNAalifold consensus structure/ dot-plot, ...)?
- Probably several descriptors needed for one RNA family as well as alternative descriptors for a specific region
- Incorporate Hamming distance matching (insertions/ deletions)

Thank you for your attention!

Acknowledgment

Sebastian Will
Michael Beckstette
Rolf Backofen

...the BLED organizers!

...and all other members of the bioinformatics group in Freiburg!

This work has been supported by the Federal Ministry of Education and Research (BMBF grant 0313921 FORSYS/FRISYS) and the German Research Foundation (DFG grant BA 2168/2-1 SPP 1258).