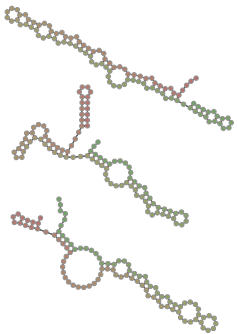
Infrared: A sampling framework for RNA design ... and beyond

Sebastian Will, Yann Ponty, Hua-Ting Yao

École Polytechnique - Institute Polytechnique de Paris

Benasque RNA meeting '22

Once upon a time ... RNARedprint

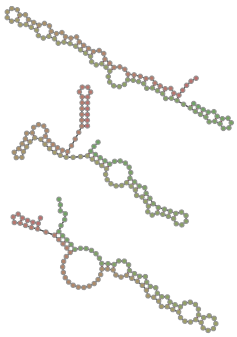


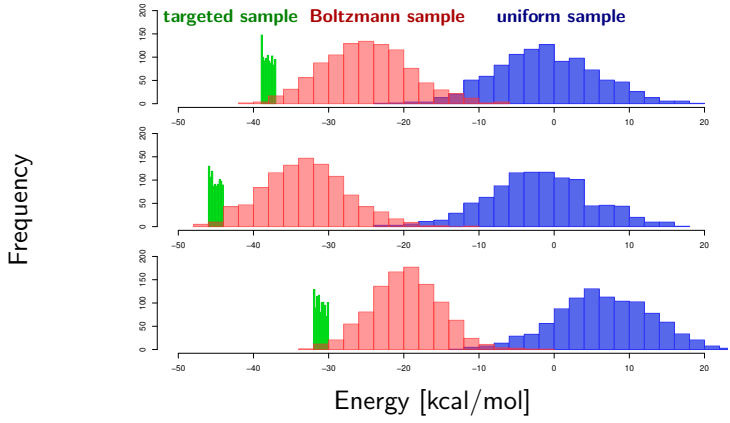
Design for **multiple structural targets**

RNARedPrint [Benasque'18, BMCBioinf 2019]:

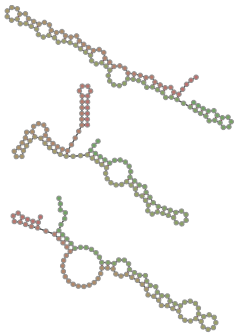
- (Boltzmann) weighted sampling of designs
- exact and highly efficient (FPT, tree-decomposition based)
- target specific properties (GC content, energy of each structure)

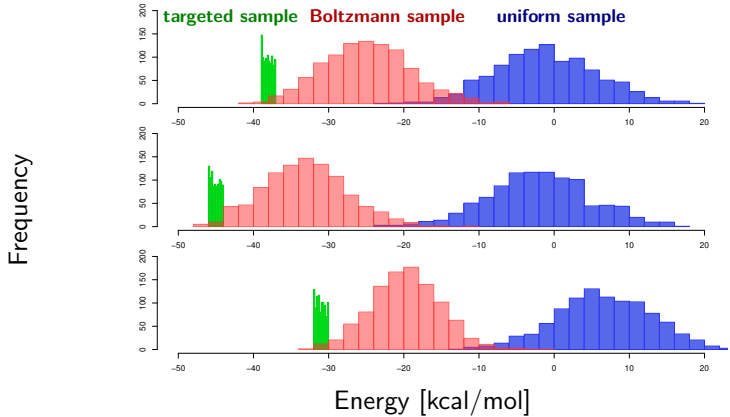
Once upon a time ... RNARedprint





Once upon a time ... RNARedprint





NEW: Infrared

- generalized framework
- support existing (Incarnation, RNARedPrint) and new (RNAPOND...) design approaches
- declarative modeling in Python (rapid prototyping)
- beyond design

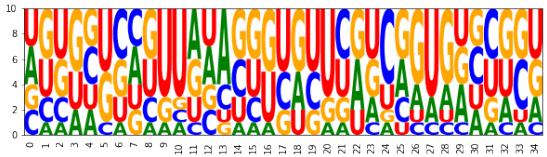
Infrared is ...

- ... a framework for **weighted sampling (and optimization)** of 'objects'...
  - ... that can be modeled in terms of **variables** and **constraints** on variables
  - ... that can be evaluated by **functions** on variables
- ... a **declarative modeling** system, where ...
  - ... tools are implemented by **describing** objects
  - ... the framework **automatically** generates the samples efficiently
- ... a **Python library** (with fast C++ engine)

Toy example of sequence design: Sample sequences that are compatible with a target structure

```
target = "((((((((((...))))(((((....)))))))))))"
model = Model(len(target), 4)
model.add_constraints(BPComp(i,j) for (i,j) in basepairs(target))
sampler = Sampler(model)
samples = [sampler.sample() for _ in range(10)]
```

```
GUGGGGCGCCUCAGGCGUGGGUAGUCCCGUCCUAU
UCCUGUGCCGGGCUGGUGAAGUUUACUUUACGGGG
UUUUGUGUUUAAAAGAUGCAGGGUACUGUACAGAG
UUAUACUUACCACGUAAGGGGCGGCCCUGUAUAA
CCGUUGGUCGUGGUGGUUAGGAAUUUUUGUAAUGG
CUCGUGUACGAAGCGUGACUUAGGUAGGUCGUGGG
UUCUGCUUUAUUUUAGAUUGGGGUACCAGGUAGAG
GGUCAGUUUCAACGAGGUUAGCGCACUGACUGACU
GUGUGUGGGGUCACCCUGGAGCGCAUUCCAUGUGC
UUAUCGGUGUUAGGUAUUUAGCAUCCUGGCGAAAG
```

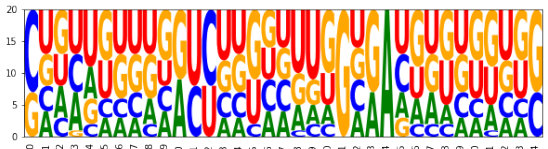


... additionally define IUPAC constraints

```
iupac_sequence = "SNNNNNNNNNRYYNNNNNNNNGNRANNNNNNNNNS"
for i, x in enumerate(iupac_sequence):
    model.add_constraints(ValueIn(i, iupacvalues(x)))

sampler = Sampler(model)
samples = [sampler.sample() for _ in range(20)]
```

```
GUAAUGGUUGGUUCAGCGAUGGUGACAUCUGUUAC
GGAAGAACGGGCCUUGUCAGGGUGACCUGUCUUCC
GGGUUCGUCGGUCUGGUGCUUGCGAGGGUGGAUUC
CUUUGCUAACAUCGUUGCGGGGAAACCCGGCGAGG
CCGAAUUGCGGCUCGUGUGUUGGGAAAUGGUUUGG
CCUUAGCCUAGCCUGGGCGGGGGAAUUUGUUAGGG
GUCAUCGAGUACUACUCAUUAGAAAUGAUGGUGAC
CGGAUAUAUUAUCAGUAGUUGGUAAUGAUUGUCUG
CCCCCUCUAGGUCCUGGUGUUGGGAAACAAGGGGG
CAAAAUGUGGAUCCUGCAGUUGGGAAGUUAUUUUG
...
```

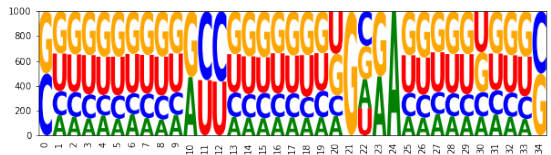


... additionally control GC content

```
model.add_functions([GCCont(i) for i in range(n)], 'gc')
model.set_feature_weight(0.15, 'gc')

sampler = Sampler(model)
samples = [sampler.sample() for _ in range(1000)]
```

```
CGUGGGACUAGUUUGGUCAACGGGAGUUGCCUAUG
GCAUGUUCCUGCUAGGGGUUUGAAAGAGCGCAUGC
GCAGUUCACCGCCGGUGUCUGGAGAUAGAGGUUGC
CGUCGGGGUAAUUUAUUGAUGGGAAUAUCUCGAUG
GGUUCUGGACGCCGUCUAUACGAAAGUGUGGAACC
GGGGCGCGUGACUUACGUUGAGGAAUUAACGCCUC
CCUUUCGUGUAUUGUGCCCAGGUGACUGGGGAAGG
GUCUAUUCUAAUUUGGACCGGGGAAUCGGGUAGAC
GGCCUUCCGUGUCAUGGUGGUGGGAAUCAAGGGUC
CCGGCCGUGUGUCGUGUCUGGGCAAUCGGGGCUGG
...
```



Control of GC content

```
model.add_functions([GCCont(i) for i in range(n)], 'gc')
model.set_feature_weight(0.15, 'gc')
```

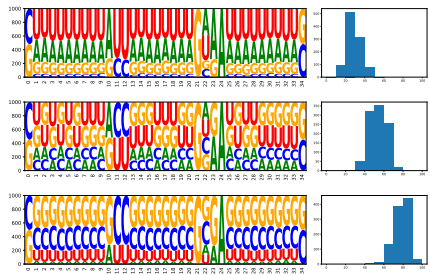
Functions `GCCont` define the **feature**

$$gc : sample \mapsto \#G + \#C$$

Infrared produces samples with

$$\Pr(sample) \propto \exp(w_{\text{gc}} \cdot gc(sample))$$

(satisfying all constraints)



$w_{\text{gc}} = -1$

$w_{\text{gc}} = 0$

$w_{\text{gc}} = 1$

Remarks: exact sampling, stochastic backtracking, requires partition functions at this point, (almost) ready to reimplement IncaRNAtion [Reinharz et al., 2017]

... next: add (arbitrary) functions and constraints

- energy functions

```
model.add_functions([BPEnergy(i,j) for (i,j) in basepairs(target)],'e')
```

- complementarity constraints for multiple target structures

```
for i,target in enumerate(targets):
    model.add_constraints([BPComp(i,j) for (i,j) in basepairs(target)])
```

- energy functions for each target structure ($\rightarrow$ **multiple features** f_i)

$$\Pr(sample) \quad \propto \quad \exp(w_1 f_1(sample) + \cdots + w_k f_k(sample))$$

- define new constraints and functions ...

Remarks: here, we can reimplement RNARedPrint [Hammer et al., 2019]
complex dependencies: how to compute partition functions (efficiently)?

(Automatic) fixed-parameter tractable sampling

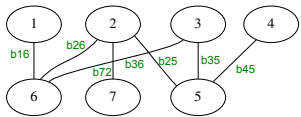
Recipe:

1. **Tree-Decompose** dependency graph
2. Apply **dynamic programming** ↑ (partition functions)
3. **Sample** ↓ (stochastic backtrace)

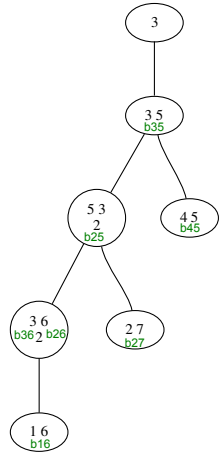
```
1 2 3 4 5 6 7

( ( . . ) ) .
. ( ( ( ) ) )
. ( ( . ) ) .
```

target structures



dependency graph



tree decomposition

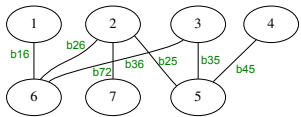
(Automatic) fixed-parameter tractable sampling

Recipe:

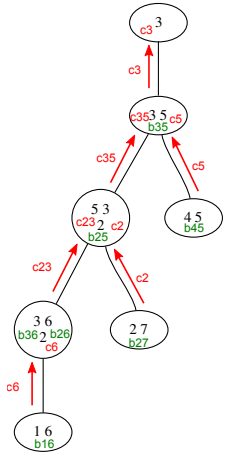
1. **Tree-Decompose** dependency graph
2. Apply **dynamic programming** ↑ (partition functions)
3. **Sample** ↓ (stochastic backtrace)

```
1 2 3 4 5 6 7
( ( . . ) ) .
. ( ( ( ) ) )
. ( ( . ) ) .
```

target structures



dependency graph



tree decomposition

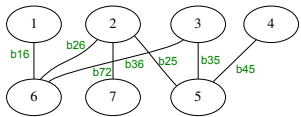
(Automatic) fixed-parameter tractable sampling

Recipe:

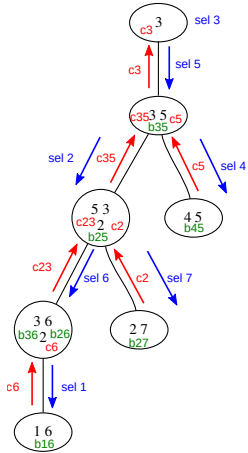
1. **Tree-Decompose** dependency graph
2. Apply **dynamic programming** ↑ (partition functions)
3. **Sample** ↓ (stochastic backtrace)

```
1 2 3 4 5 6 7
( ( . . ) ) .
. ( ( ( ) ) )
. ( ( . ) ) .
```

target structures



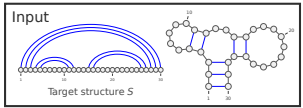
dependency graph



tree decomposition

Theorem: Boltzmann sampling is efficient for fixed tree width w: $\mathcal{O}(n\,k\,\mathbf{4^w} + t\,n\,k)$

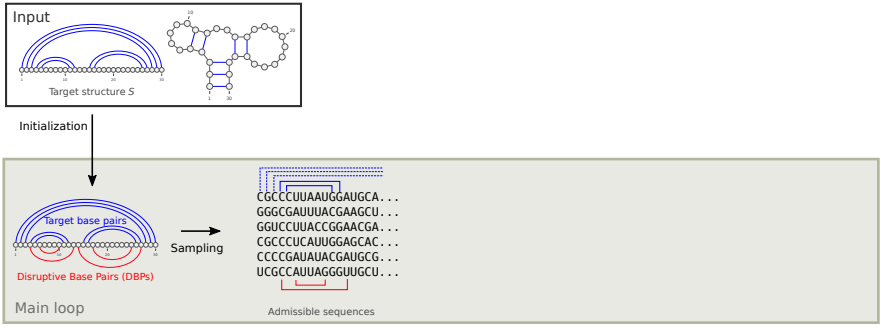
RNAPOND [Yao et al., RECOMB 2021]



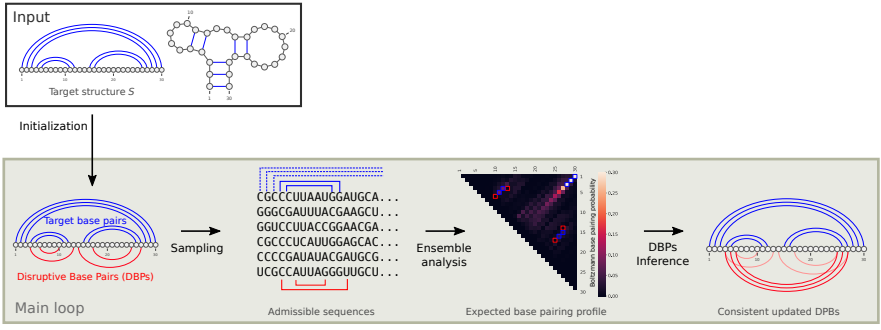
Initialization

Main loop

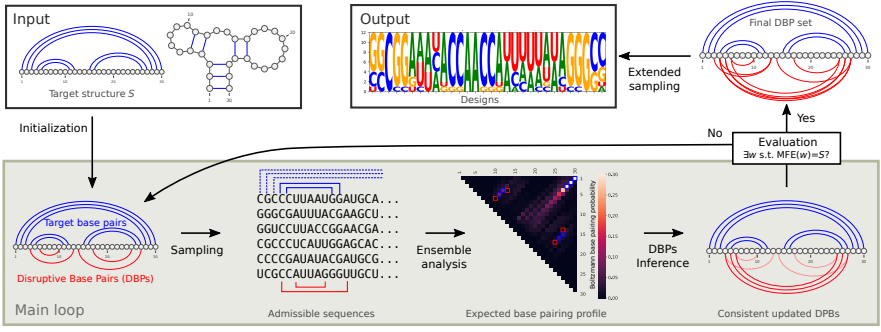
RNAPOND [Yao et al., RECOMB 2021]



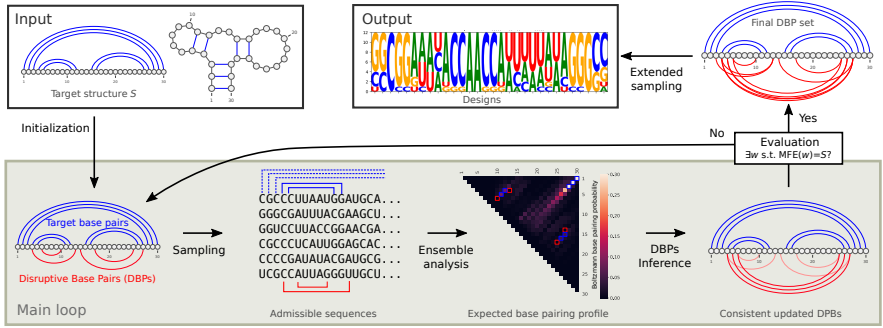
RNAPOND [Yao et al., RECOMB 2021]



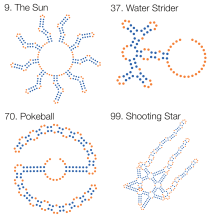
RNAPOND [Yao et al., RECOMB 2021]

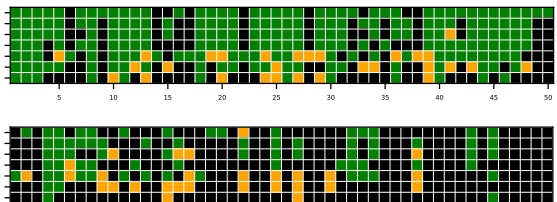


RNAPOND [Yao et al., RECOMB 2021]



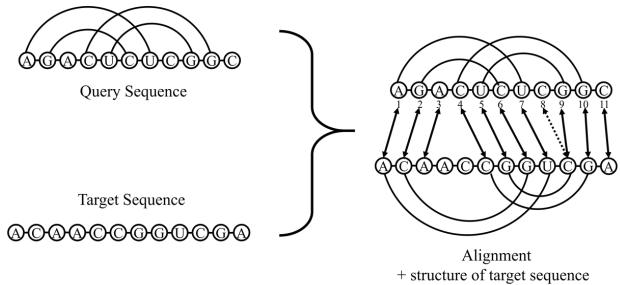
Eterna100 benchmark





LiCoRNA-like alignment of RNA with pseudoknots

IN: query sequence A with (pseudoknotted) structure, target sequence B
OUT: weighted **sample** of alignments of A and B (or **optimal** alignment)



[Rinaudo et al., 2012]

Using Infrared, LiCoRNA [Rinaudo et al., 2012] was (largely) reimplemented, where the (abstract) alignment model is directly encoded as Infrared model ($<<$ 400 LOC)

Take home

- Framework for **efficient sampling** and **multi-dim. Boltzmann sampling**
- **Declarative modeling** of objects and their features → Rapid prototyping
- **Fixed-parameter tractable** (treewidth) → good for sparse dependency graphs
- **RNA design**: RNARedPrint, RNAPOND, ...
- ... **and beyond**, e.g. RNA PK-alignment; building background models
- **Bookchapter/Tutorial** on Design in Infrared: `hal.inria.fr/hal-03711828`
- **Code, docu, and code examples**:
  `www.lix.polytechnique.fr/~will/Software/Infrared/`

You are welcome to discuss more details with Yann, Hua-Ting, me