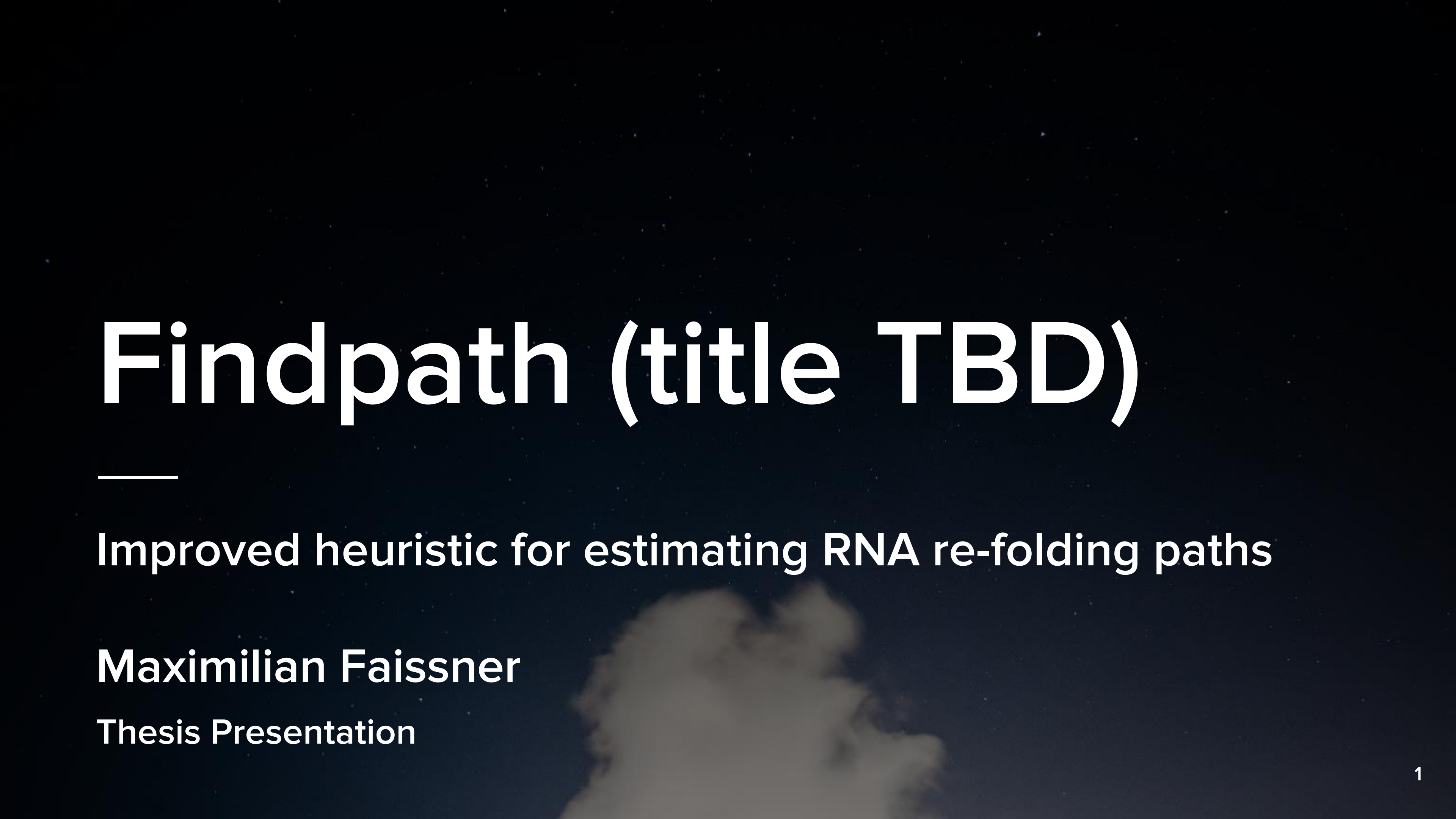


Findpath (title TBD)

Improved heuristic for estimating RNA re-folding paths

Maximilian Faissner

Thesis Presentation

Table of Contents

- RNA kinetics overview & the minimum energy barrier
- The Findpath algorithm
- Project motivation
- Findpath Rewrite & Extensions
- Results and Outlook

Ribonucleic Acids (RNAs)

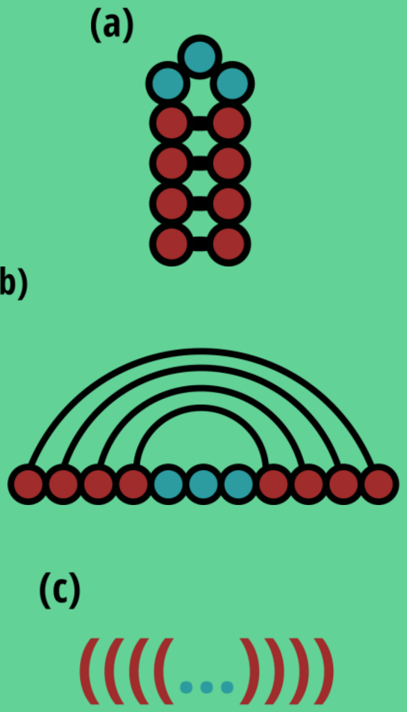
in the context of RNA kinetics & this project

RNA string representation with alphabet $\Sigma := \{A, C, G, U\}$

RNA molecules fold into many confirmations - the secondary structure level is sufficient for RNA folding, kinetics and thermodynamics. Restrictions:

- canonical base pairs (AU, CG, and GU pairs)
- pseudoknot-free secondary structures (no crossing basepairs – simple dot-bracket string notation (c) is sufficient)

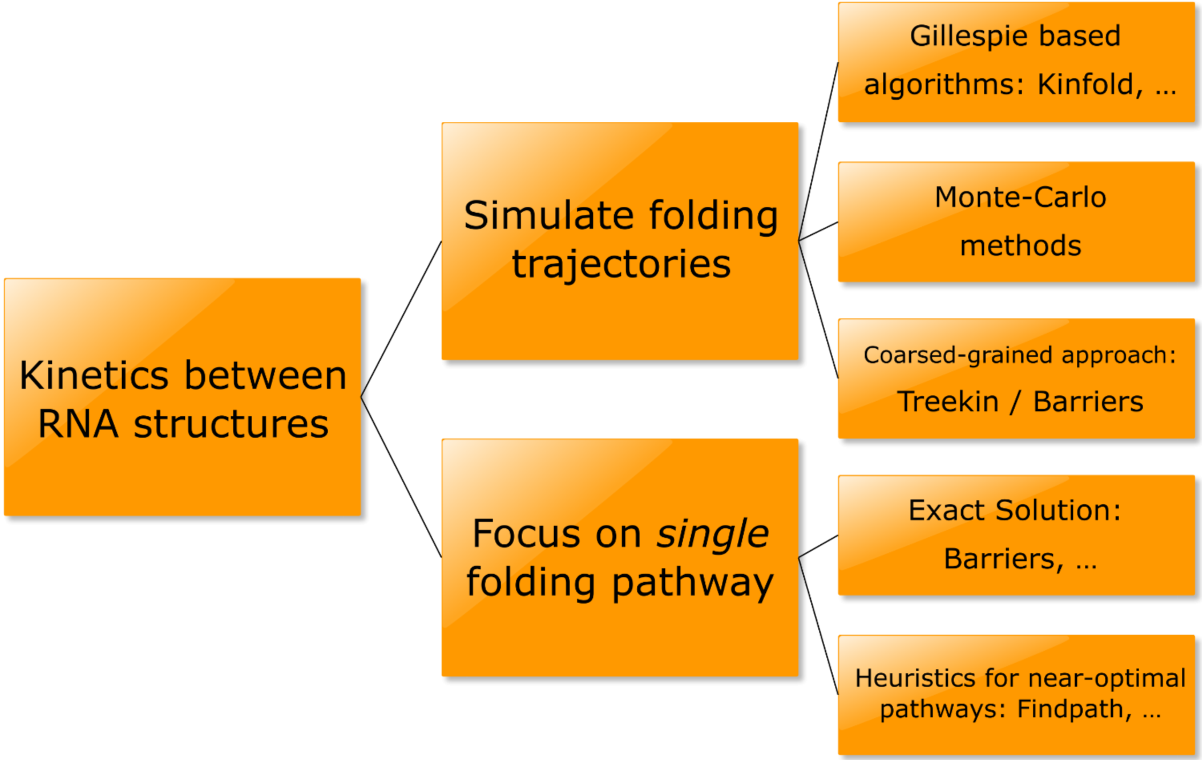
Conformational changes: Only elementary moves $add_{(i,j)}$ or $del_{(i,j)}$



Representations of RNA secondary structure

Source: https://2020.igem.org/Team:Heidelberg/Model/RNA_Combinatorics

RNA kinetic folding landscape



Kinetics between RNA structures

Simulate folding trajectories

Focus on *single* folding pathway

Gillespie based algorithms: Kinfold, …

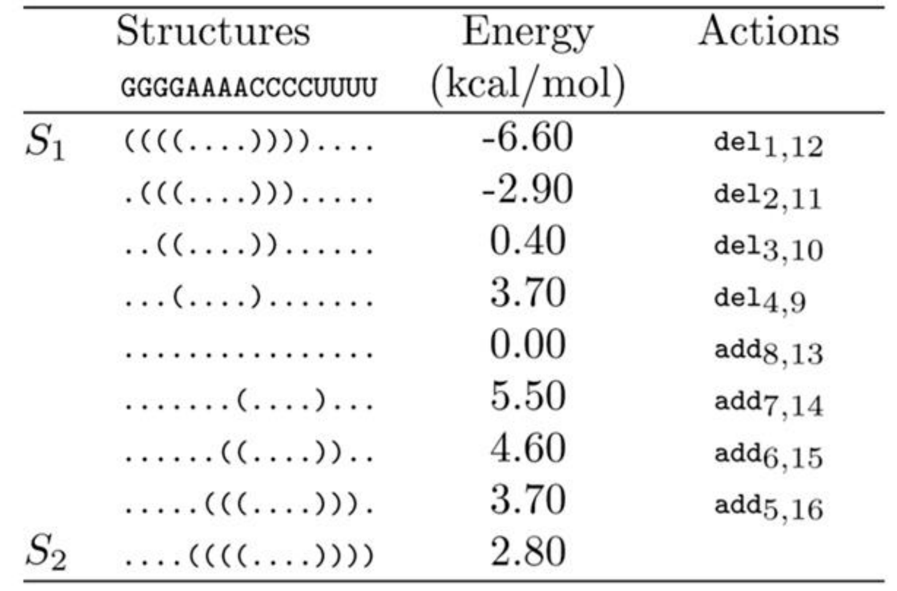
Monte-Carlo methods

Coarsed-grained approach: Treekin / Barriers

Exact Solution: Barriers, …

Heuristics for near-optimal pathways: Findpath, …

Direct RNA folding pathways



	Structures	Energy	Actions
	GGGGAAAACCCCUUUU	(kcal/mol)	
S_1	((((....))))....	-6.60	del$_{1,12}$
	.(((....)))).....	-2.90	del$_{2,11}$
	..((....))......	0.40	del$_{3,10}$
	...(....).......	3.70	del$_{4,9}$
	....	0.00	add$_{8,13}$
	.......(....)...	5.50	add$_{7,14}$
	......((....))..	4.60	add$_{6,15}$
	.....(((....))).	3.70	add$_{5,16}$
S_2	....((((....))))	2.80	

Exemplary Direct Folding Path between initial (S_1) and target structure (S_2)

Source: http://dx.doi.org/10.1186/1471-2105-13-S3-S5

Direct paths:

number of elementary moves is determined by the basepair distance (S_1, S_2).

Gives an upper bound for the “real”, indirect path energy barrier.

Proven NP-hard problem: Heuristics required > 200-300 nt length

The Findpath Algorithm (1)

Bounded breath-first search heuristic

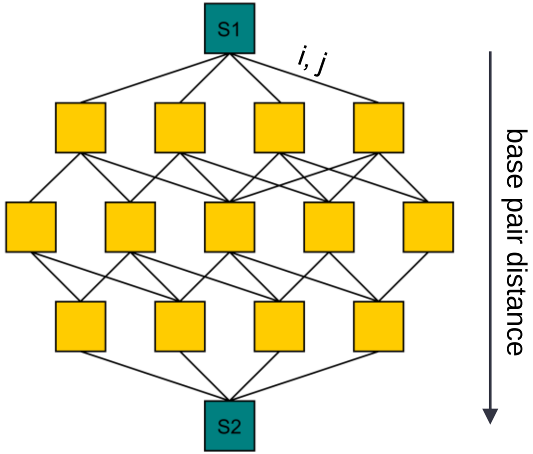
Goal: Find the folding path with the lowest energy barrier, without testing all $n!$ possible paths

2 step process for each distance step:

1. Candidate generation step: verify valid moves, energy evaluation (constant runtime per structure)
2. Sorting step: Keep best k candidates. Remove duplicate structures.

Expected runtime: $O(kn^2)$
As k (search width) tends to ∞, the optimal result will be found.

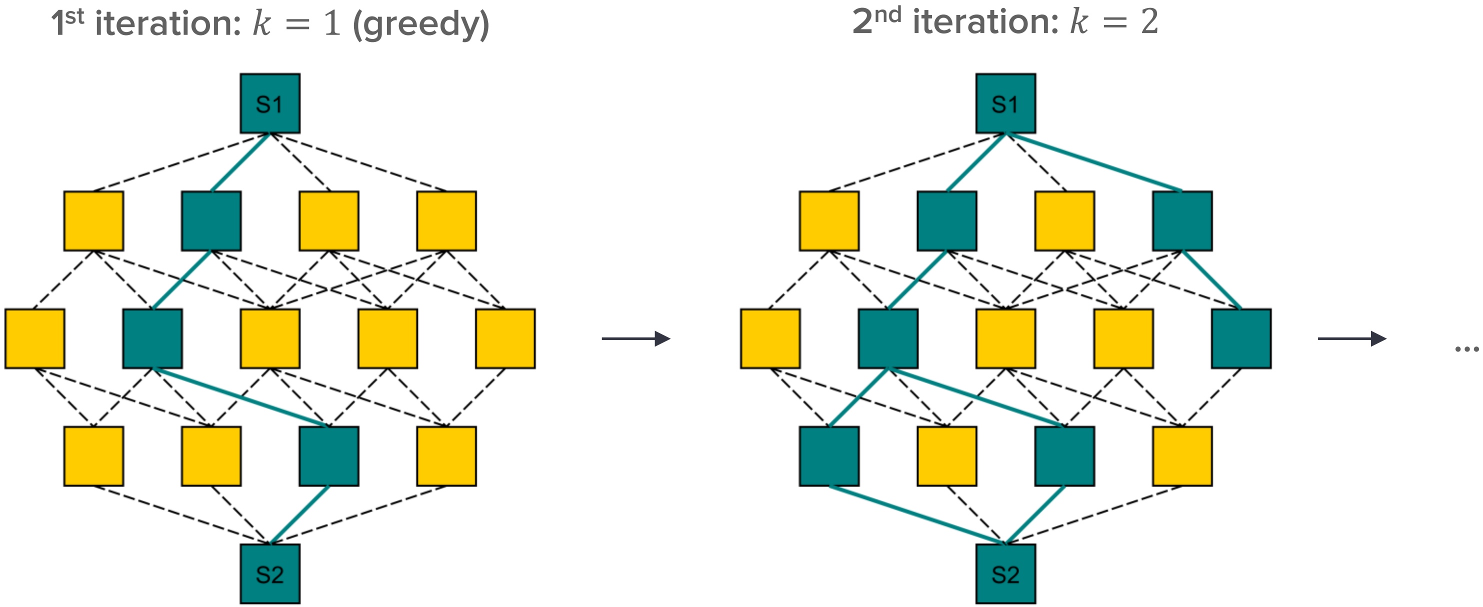
Graph-based representation:



nodes:
unique structures
edges:
valid moves $add_{(i,j)}$ or $del_{(i,j)}$
edge weight:
maximum free energy (energy barrier)

The Findpath Algorithm (2)

Gradual search width (k) increase



Both Forward and Backward directional passes are computed

Project Motivation

Augment direct-path Findpath with old & new ideas

Novel ways for indirect path heuristics

Direct-Path Findpath Extensions:

1. Divide & Conquer Approach
2. Move Restrictions
3. Path-MFE Extension
4. Performance Optimizations

1 Divide & Conquer Extension (1)

Idea: Separate the folding pathway into independently computable sections, then merge resulting pathways recursively. Currently, no implantation of this procedure using the Turner energy model.

Previous citations for this idea:

“Certain base pairs are observed to be present in every ground-state structure. These ‘frozen’ pairs divide the molecule up into mutually inaccessible pieces. All of the separate pieces contribute to determining the distance between structures, but only the largest piece will contribute to the barrier height.” *(Steven R Morgan and Paul G Higgs 1998 J. Phys. A: Math. Gen.* ***31*** *3153)*

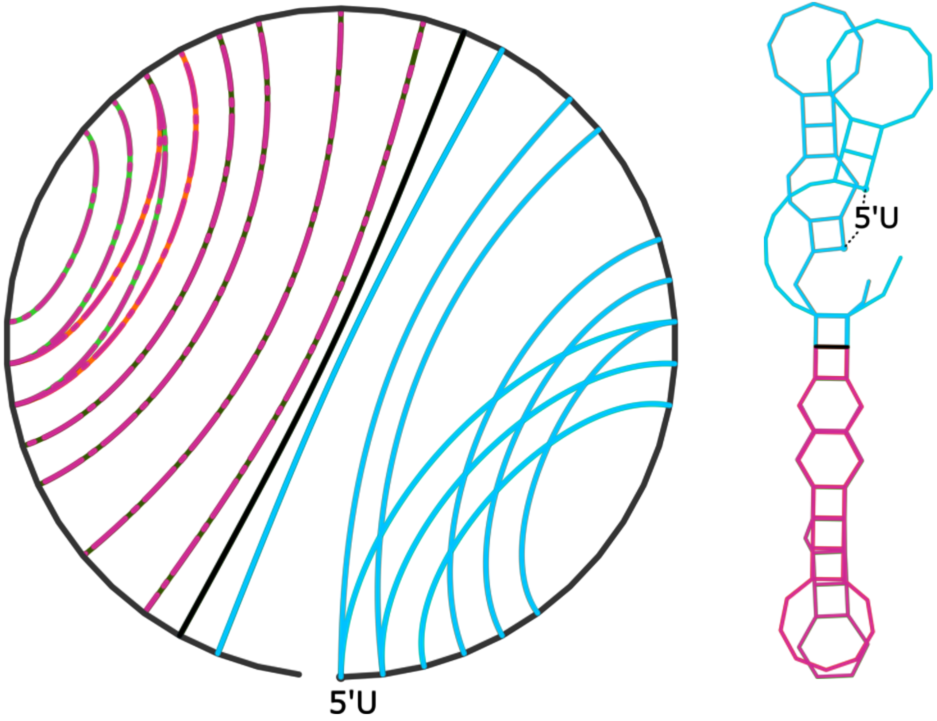
→ incorrect statement for the Turner energy model!

“Our methods exploit elegant algorithms for bipartite graphs to split a problem into independent subproblems where possible. [...] Our algorithms are highly amenable to parallelization and have potential to work with more sophisticated energy models that include Turner parameters for base stacking, for example.” *(Thachuk C et al. An algorithm for the energy barrier problem without pseudoknots and temporary arcs. Pac Symp Biocomput. 2010)*

Correct idea, but the merging process is not straight-forward with the Turner energy model!

1 Divide & Conquer Extension (2)

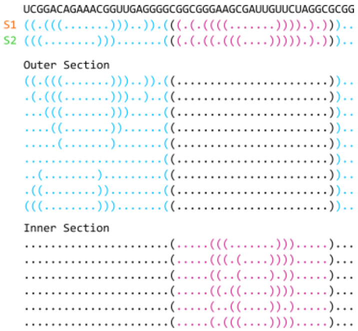
```
   UCGGACAGAAACGGUUGAGGGGCGGCGGGAAGCGAUUGUUCUAGGCGCGG
S1 ((.(((.......)))..)).((.(.((((.......)))).).))..
S2 (((........)))........((.(.((.(((....))))).).))..
```



Split into independent subproblems:

- Find constant basepairs present in both S_1 and S_2
- Recursively start a new section at a constant interior loop basepair (whenever base pair distance > 1)
- Reconstruct final path by recursively merging individual paths

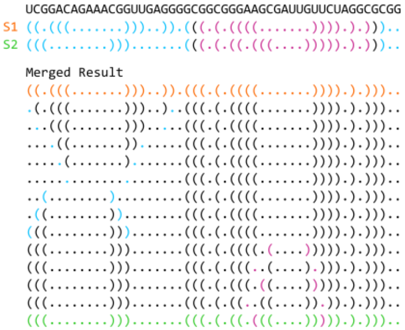
1 Divide & Conquer Extension (3)



```
   UCGGACAGAAACGGUUGAGGGGCGGCGGGAAGCGAUUGUUCUAGGCGCGG
S1 ((.(((.......)))..)).((.(.((((.......)))).).)))..
S2 (((........)))........((.(.((.(((....))))).).)))..

   Outer Section
   ((.(((.......)))..)).((.......................))..
   .(.(((.......)))..)..((.......................))..
   ...(((.......))).....((.......................))..
   ....((.......))......((.......................))..
   .....(.......).......((.......................))..
   .....................((.......................))..
   ..(........).........((.......................))..
   .((........))........((.......................))..
   (((........)))........((.......................))..

   Inner Section
   ......................(.....(((.......))).....)...
   ......................(.....(((.(....)))).....)...
   ......................(.....((..(....).)).....)...
   ......................(.....((.((....)))).....)...
   ......................(.....(..((....)).).....)...
   ......................(.....(.(((....)))).....)...
```



```
   UCGGACAGAAACGGUUGAGGGGCGGCGGGAAGCGAUUGUUCUAGGCGCGG
S1 ((.(((.......)))..)).((.(.((((.......)))).).)))..
S2 (((........)))........((.(.((.(((....))))).).)))..

   Merged Result
   ((.(((.......)))..)).(((.(.((((.......)))).).)))..
   .(.(((.......)))..)..(((.(.((((.......)))).).)))..
   ...(((.......))).....(((.(.((((.......)))).).)))..
   ....((.......))......(((.(.((((.......)))).).)))..
   .....(.......).......(((.(.((((.......)))).).)))..
   .....................(((.(.((((.......)))).).)))..
   ..(........).........(((.(.((((.......)))).).)))..
   .((........))........(((.(.((((.......)))).).)))..
   (((........))).......(((.(.((((.......)))).).)))..
   (((........))).......(((.(.((((.(....))))).).)))..
   (((........))).......(((.(.(((..(....).))).).)))..
   (((........))).......(((.(.(((.((....))))).).)))..
   (((........))).......(((.(.((..((....)).)).).)))..
   (((........))).......(((.(.((.(((....))))).).)))..
```

- Saddle points of subpaths infer no information about the merged energy barrier!

  ➔ merging procedure is necessary

- No optimality criterion exists to select for optimal subpaths: in rare cases, even suboptimal subpaths are required for merging!

  ➔ best effort approach for merging

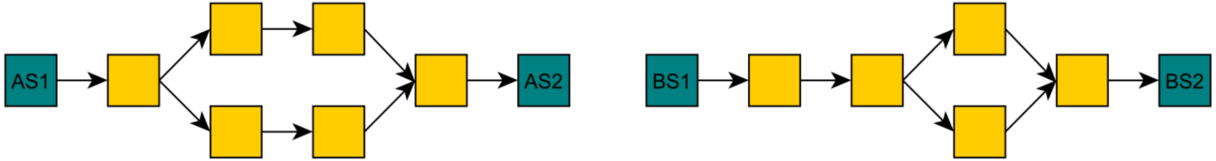
1 Divide & Conquer Extension (4)

Best Effort Merging: Use as many input subpaths as possible

Step 1 – Postprocessing: Reduce redundancy of input paths, generate input graph

```
(0,  0 ) --> (-46, -51) --> (-40, -56) --> (-41, -55) --> (-43, -53) --> (-44, -52) --> ( 43,  52) --> ( 42,  53) --> ( 40,  54)
(0,  0 ) --> (-46, -51) --> (-43, -53) --> ( 42,  53) --> (-44, -52) --> ( 43,  52) --> (-41, -55) --> (-40, -56) --> ( 40,  54)
                                                                     ↓
(0,  0 ) --> (-46, -51) --> (-40, -56) --> (-41, -55) --> (-43, -53) --> (-44, -52) --> ( 43,  52) --> ( 42,  53) --> ( 40,  54)
                        \-> (-43, -53) --> ( 42,  53) --> (-44, -52) --> ( 43,  52) --> (-41, -55) --> (-40, -56) -/
```

Step 2 – Merging: Apply Findpath algorithm to merge 2 graphs, using k_{merge} (merging search width)



2 Move Restrictions

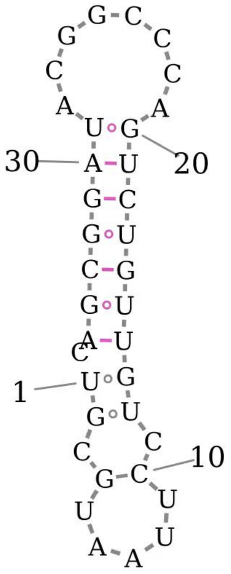
```
   UGCGUAAUUCCUGUUGUCUGACCCGGCAUAGGCGACG
S1 ((.(.....).))(((((((........)))))))..
                  [...]
S2 ((.(.....).))..(((...((.......)).))).
```

Goal: Simplified, helix-based refolding model

Implementation: Reduce possible move choices / possible candidates for the next iteration

Theory:

- *Consecutive moves within a double helix section follow certain rules*
- *Define helix sections, then apply adjacency restrictions & greedy restrictions*
- *Every helix section has to be considered as isolated unit*



2.1 Adjacency Restriction

Findpath (others as well) create unrealistic folding paths, as result of the Turner energy model. Optimal paths don’t require non-adjacent extensions.

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

Correction: Subsequent moves within helices should follow these rules:

- continuously extending a pre-existing helix, or
- removing terminal resides of a helix

Findpath restriction implementation: Filter during candidate generation

In any given state, only 1 or 2 moves within a helix are valid

2.2 Greedy Restriction (add moves)

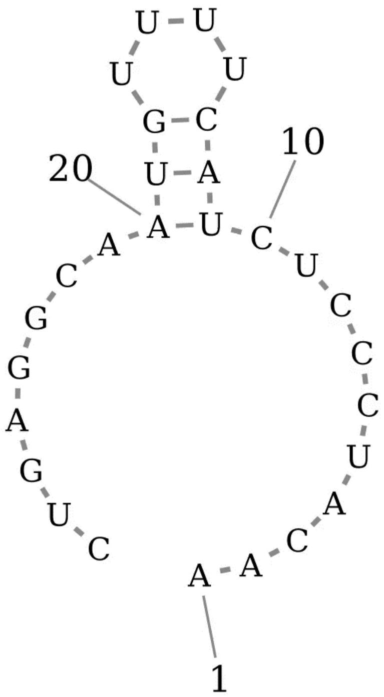
```
   AACAUCCCUCUACUUUUGUAACGGAGUC
S1 ..........(((....)))........
             [...]
S2 .((.(((...(((....)))..))))).
```

The optimal helix assembly permutation should be computed greedily.

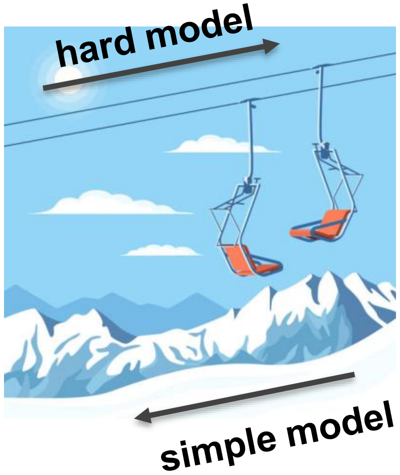
Greedy assembly is expected from the Turner energy model.

Findpath restriction implementation:

- Initial move: check energy contribution of all moves, only 1 candidate remains (sorting step)
- Subsequent moves: apply adjacency restriction first, 1 candidate remains



2.3 Greedy Restriction (del moves)



The optimal helix permutation of disassembly moves should not be computed greedily.

Otherwise, often no convergence to best saddle point.

Reasoning: Greedy only works if a helix operation leads to a lower energy state.

NB: Assembly and disassembly moves are often interleaved (see right). It is (in most cases) not possible to compute delete moves from the opposite direction.

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

3 Direct Path MFE

If a direct path minimum free-energy (MFE) structure exists (en. lower than S_1 and S_2), than this structure has to be on the optimal path.

Source & Idea:

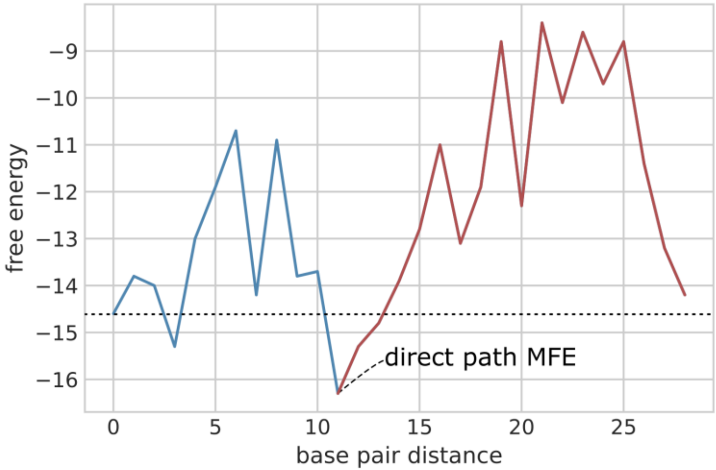
Laurent Bulteau, Bertrand Marchand, Yann Ponty.

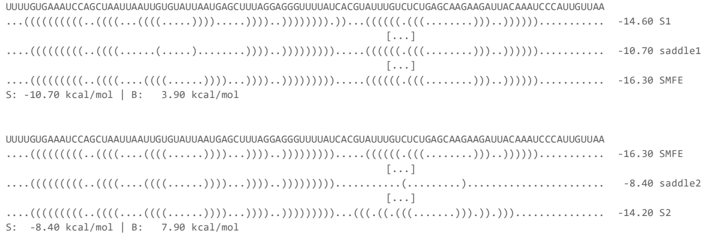
A new parametrization for independent set reconfiguration and applications to RNA kinetics. IPEC 2021 - 16th International Symposium on Parameterized and Exact Computation, Sep 2021, Lisbon, Portugal.

Implementation:

MFE calculation: Restrict base pairs, run global MFE on sequence.

Run Findpath S_1 to S_{MFE} and S_{MFE} to S_2

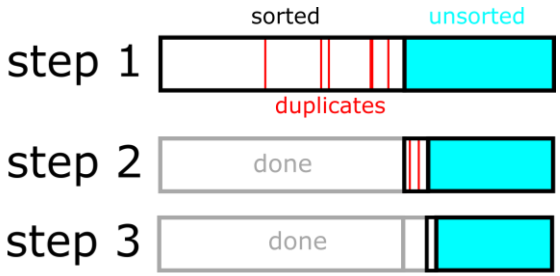




```
UUUUGUGAAAUCCAGCUAAUUAAUUGUGUAUUAAUGAGCUUUAGGAGGGUUUUAUCACGUAUUUGUCUCUGAGCAAGAAGAUUACAAAUCCCAUUGUUAA
...((((((((((..((((...((((.....)))).....))))..)))))))).))...((((((.(((........)))..))))))...........  -14.60 S1
                                                          [...]
....(((((((((..((((......(.....)........))))..))))))))).....((((((.(((........)))..))))))...........  -10.70 saddle1
                                                          [...]
....(((((((((..((((....((((......))))...))))..))))))))).....((((((.(((........)))..))))))...........  -16.30 SMFE
S: -10.70 kcal/mol | B:   3.90 kcal/mol


UUUUGUGAAAUCCAGCUAAUUAAUUGUGUAUUAAUGAGCUUUAGGAGGGUUUUAUCACGUAUUUGUCUCUGAGCAAGAAGAUUACAAAUCCCAUUGUUAA
....(((((((((..((((....((((......))))...))))..))))))))).....((((((.(((........)))..))))))...........  -16.30 SMFE
                                                          [...]
....(((((((((..((((....((((......))))...))))..)))))))))...........(.........).......................   -8.40 saddle2
                                                          [...]
....(((((((((..((((....((((......))))...))))..)))))))))...(((.((.(((.......))).)).)))...............  -14.20 S2
S:  -8.40 kcal/mol | B:   7.90 kcal/mol
```

4 Other Performance Improvements



Partial energy sorting of best candidates (fig.)

Not all candidates have to be sorted & checked for duplicates

Optimized structure hashing

2 different hashing methods: string hash & move tuple hash

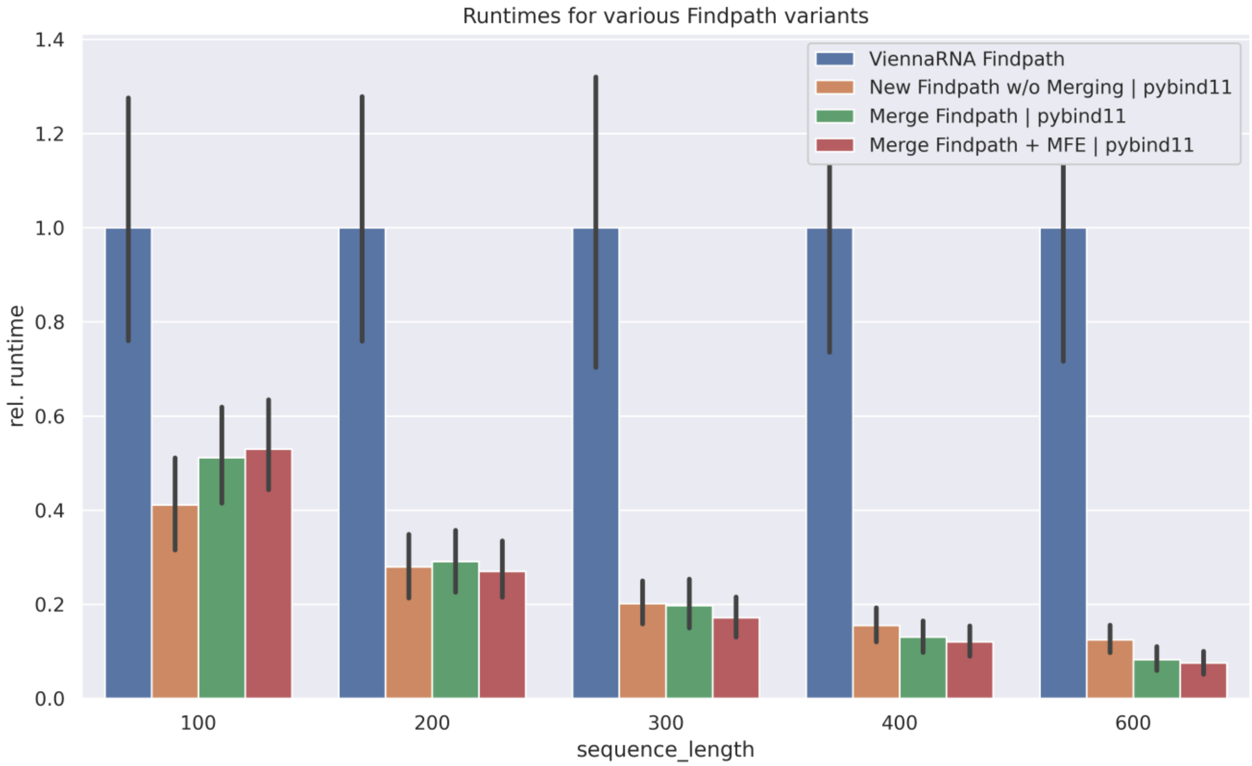
Memory allocation optimization

Reduced memory consumption & alignment (better cache locality)

Multithreading & optimized FWD/BWD passes

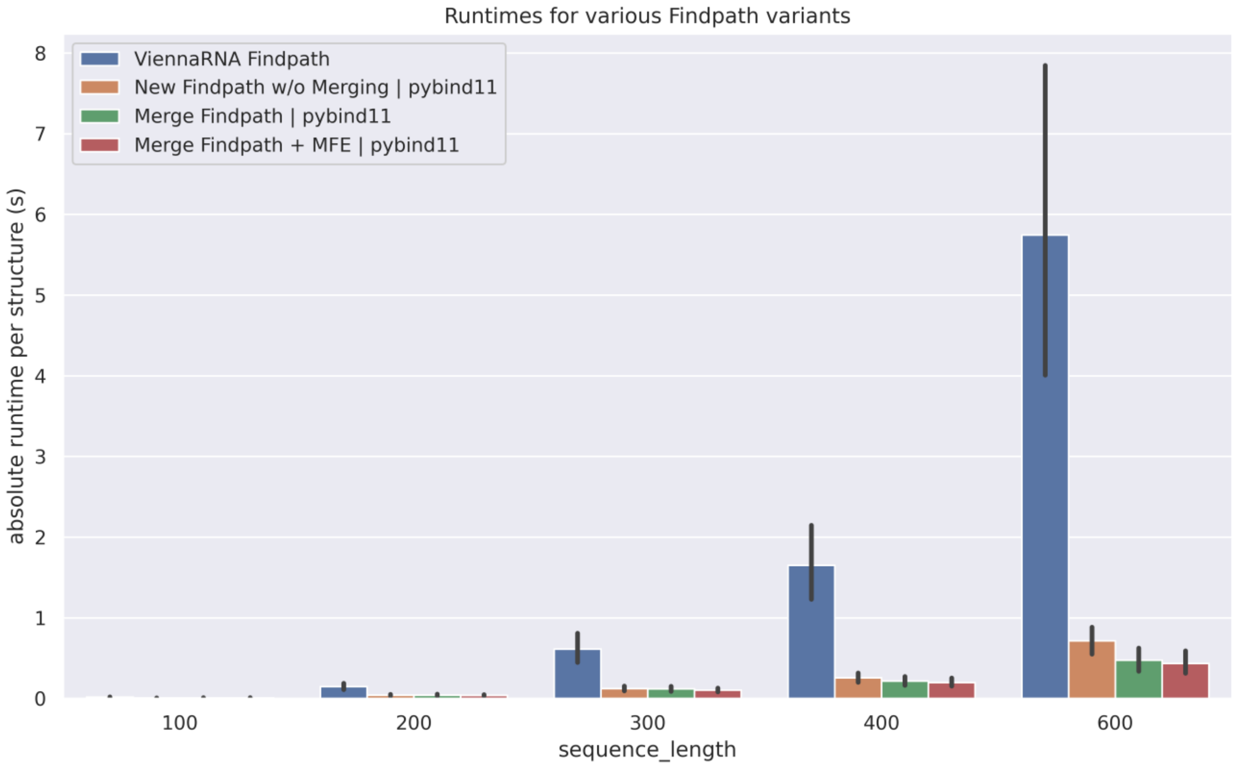
Compute FWD & BWD passes simultaneously / better sw. ramp-up

Results: Comparison relative Runtimes



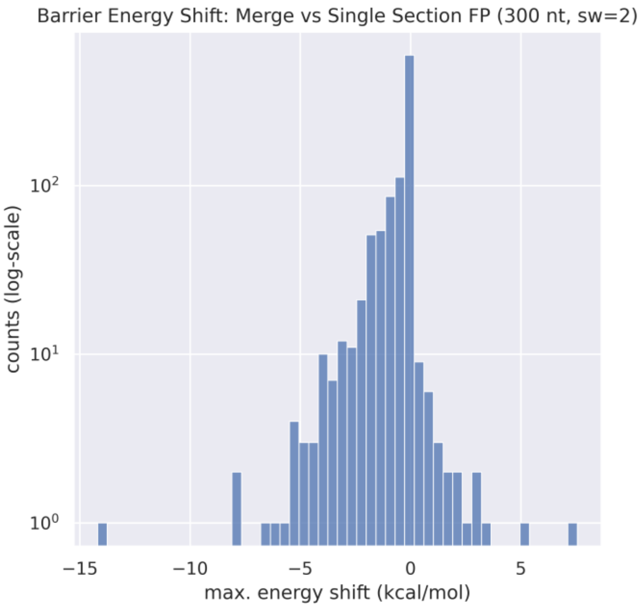
Findpath settings: search width = bp_dist*2; merge search width = bp_dist*1

Results: Comparison absolute Runtimes



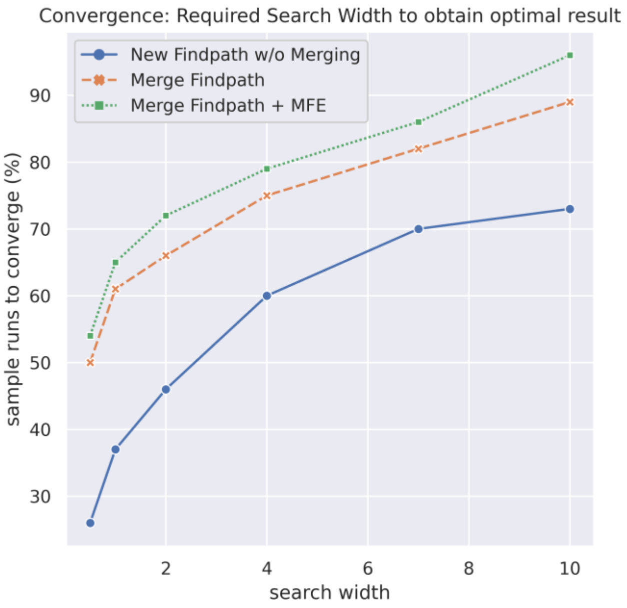
Findpath settings: search width = bp_dist*2; merge search width = bp_dist*1

Results: Barrier Shift (300 nt samples)



Mean shift at 300 nt: -0.53 kcal/mol

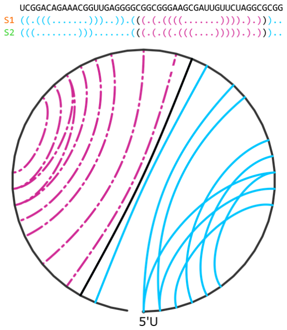
Mean shift at 600 nt: -2.89 kcal/mol



Convergence Tests (optimal result = best found barrier height)

Findpath settings: search width = bp_dist*2; merge search width = bp_dist*1

Empirical Results


UCGGACAGAAACGGUUGAGGGGCGGCGGGAAGCGAUUGUUCUAGGCGCGG
S1 ((.(((.......)))..)).(((.(.((((.......)))).).)))..
S2 (((........)))........((((.(.((.(((....))))).).)))..
5'U

Divide & Conquer Split:

- *better for sequences > 300 nt*
- *no convergence guarantee, merging has to be a best-effort approach (optional feature?)*

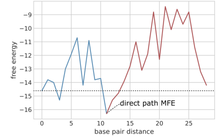
Adjacency & Greedy Restrictions:

- *Performance gains between 14% (100 nt) to 19.5%(400 nt), insignificant change of barrier energies*
- *In rare cases (0.1%), no convergence (related to Turner energy model)*

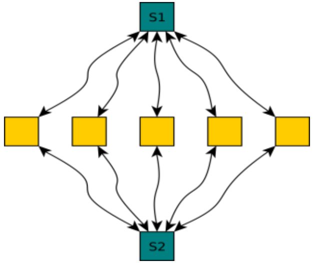
```
....(.....)....
...((.....))...
(..((.....))..)
(.(((.....))).)
(((((.....)))))
```

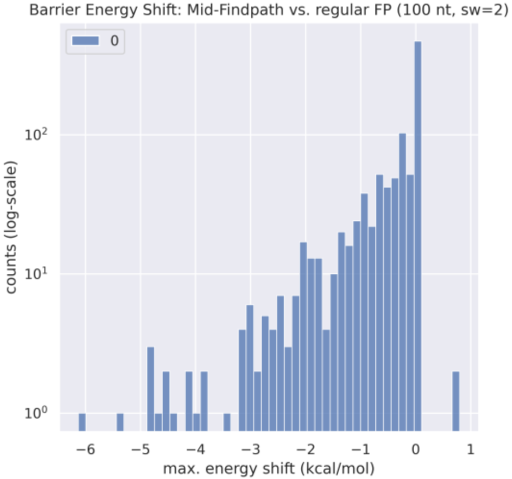
Direct-path MFE midpoint:

- *No disadvantages, but usability depends on dataset*



Midpoint Experiments





Hypothesis:

One of the biggest weak points is the static forward/ backward search. Certain sections (helices) are much easier to compute in one direction.

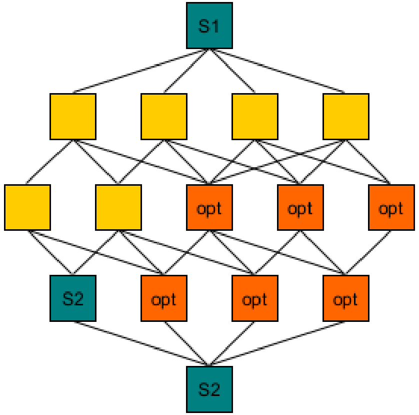
Experiment: (100 nt samples, sw. multiplier = 1.0)

Stop at basepair distance/2, calculate fwd. and bwd. passes from midpoint to S_1 & S_2.

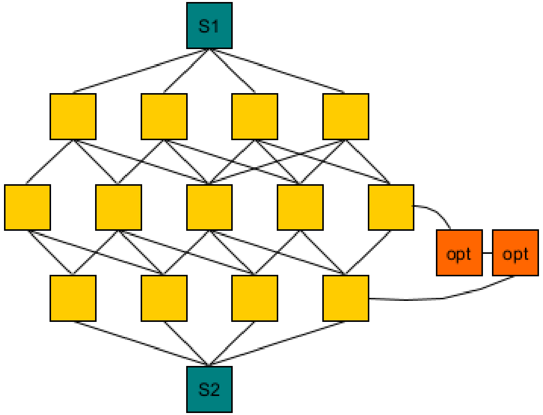
Results:

Barrier energies via saddle points are much lower. Ideally, an optimized pathfinding algorithm should dynamically switch direction & allocate computing resources more thoughtfully.

Indirect Variants



Open-ended approach
(variable basepair distance)



Detour approach
(strict basepair distance,
References S2)

- Reliance on “oracle” to provide extra moves as input
- Only works with indirect moves without repeats.

Thanks!

Acknowledgments etc.

Uppsala universitet 1852–1916
Det goda universitetet. Festskrift till Anders Hallberg