Approximate RNA folding kinetics

Gregor Entzian

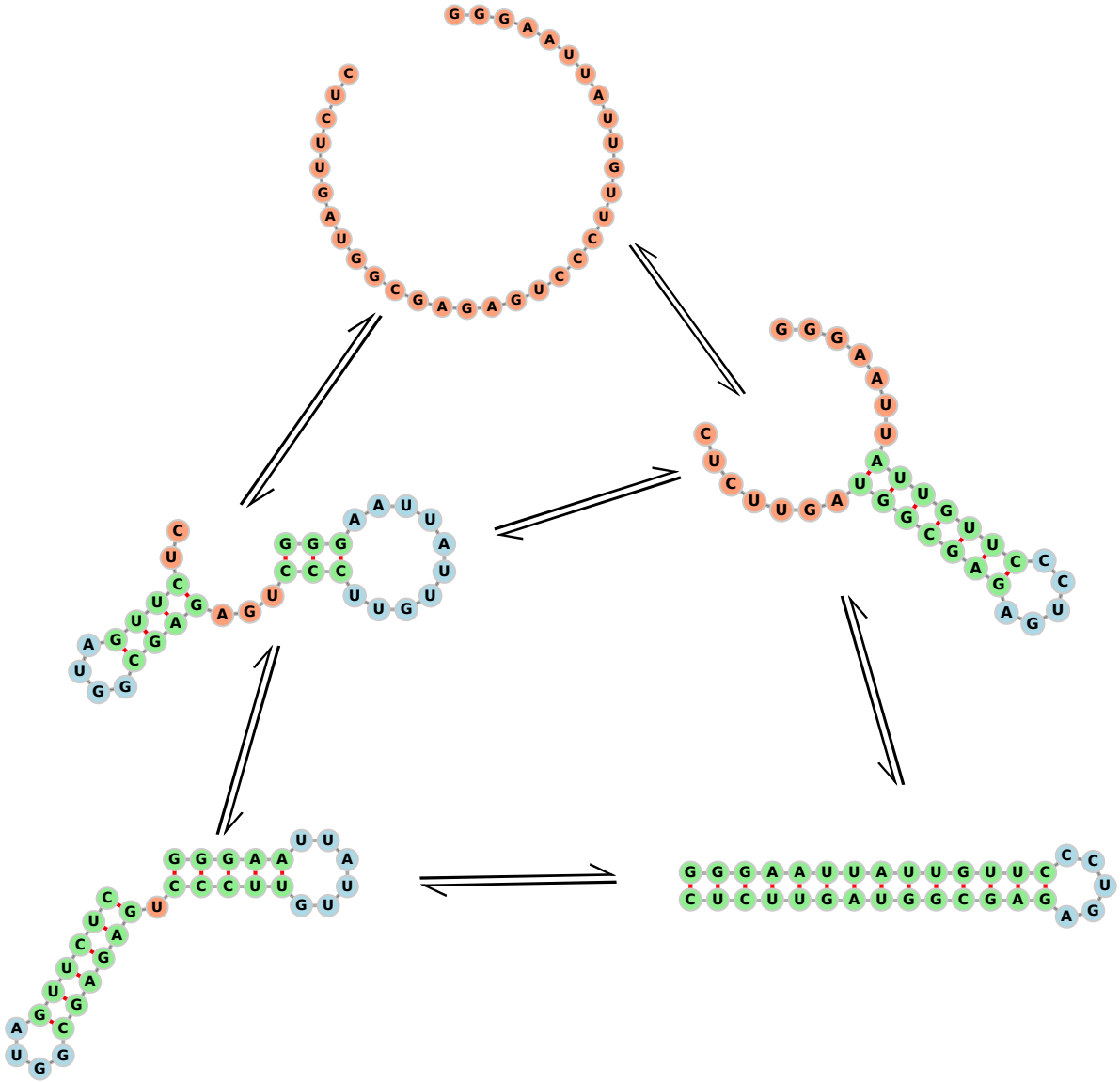
University of Vienna, Faculty of Chemistry, Department of Theoretical Chemistry

entzian@tbi.univie.ac.at

Supervisor: Univ.-Prof. Dipl.-Phys. Dr. Ivo L. Hofacker

February 14, 2018

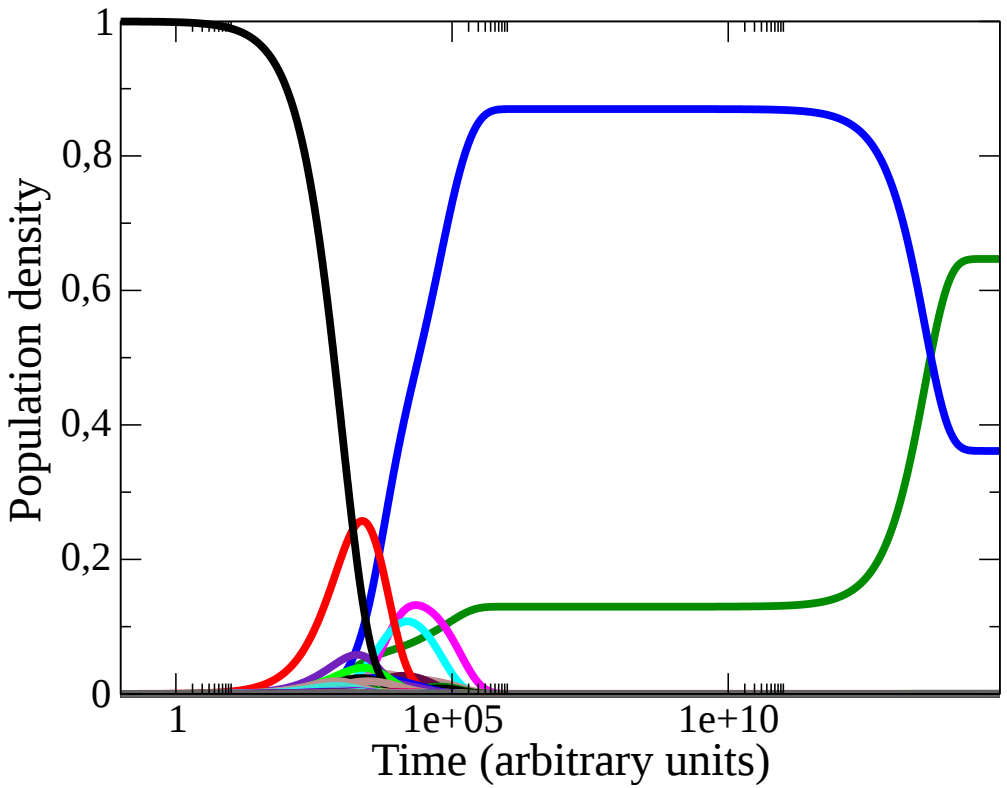
RNA Structure Dynamics

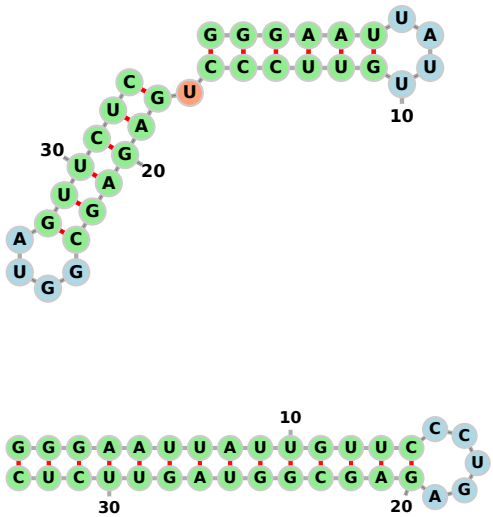


RNA Folding Kinetics

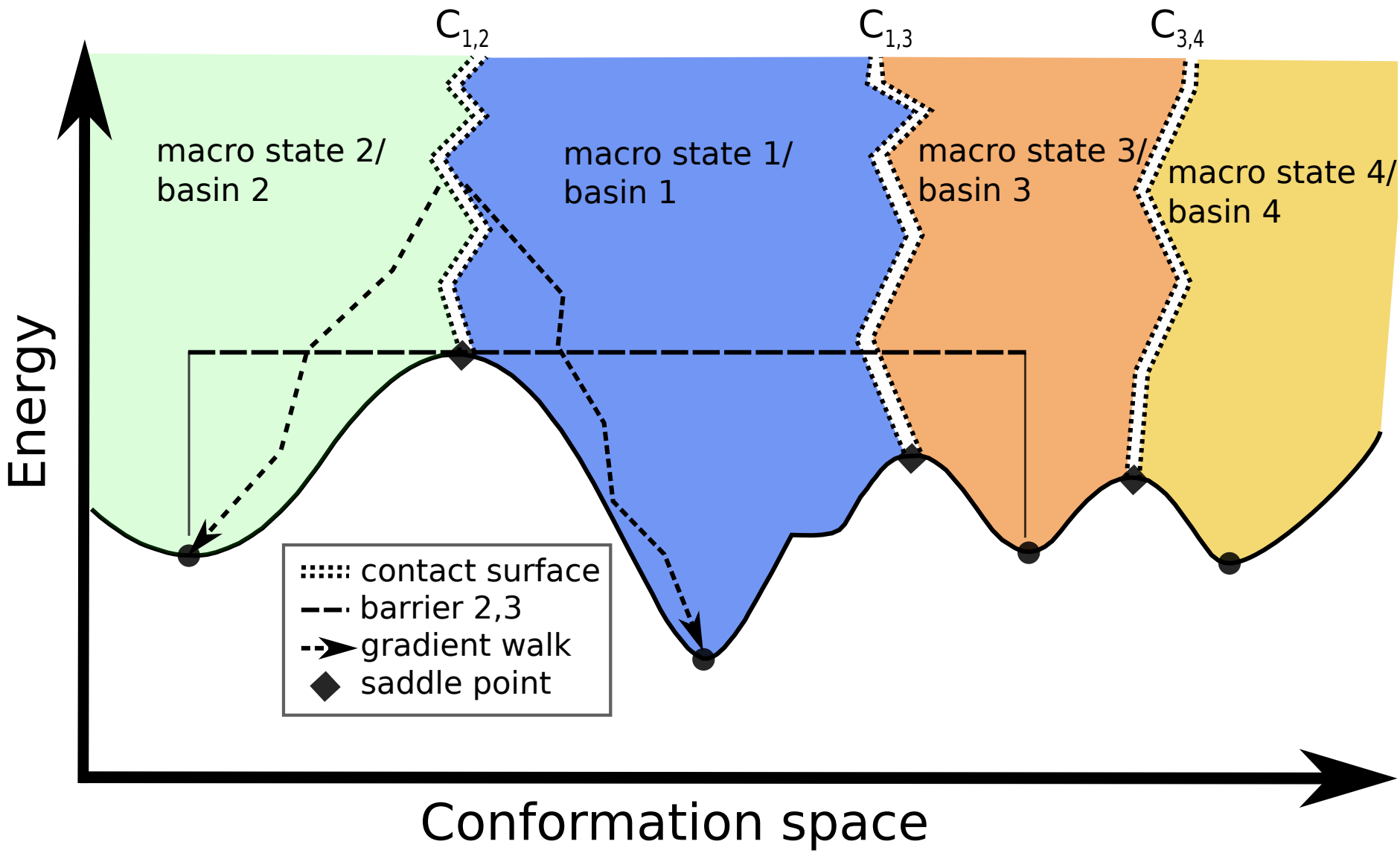
$$\frac{dp_i(t)}{dt} = \sum_{j \neq i} [p_j(t) r_{ji} - p_i(t) r_{ij}]$$

$$\vec{p_t} = e^{Kt} \cdot \vec{p_0}$$

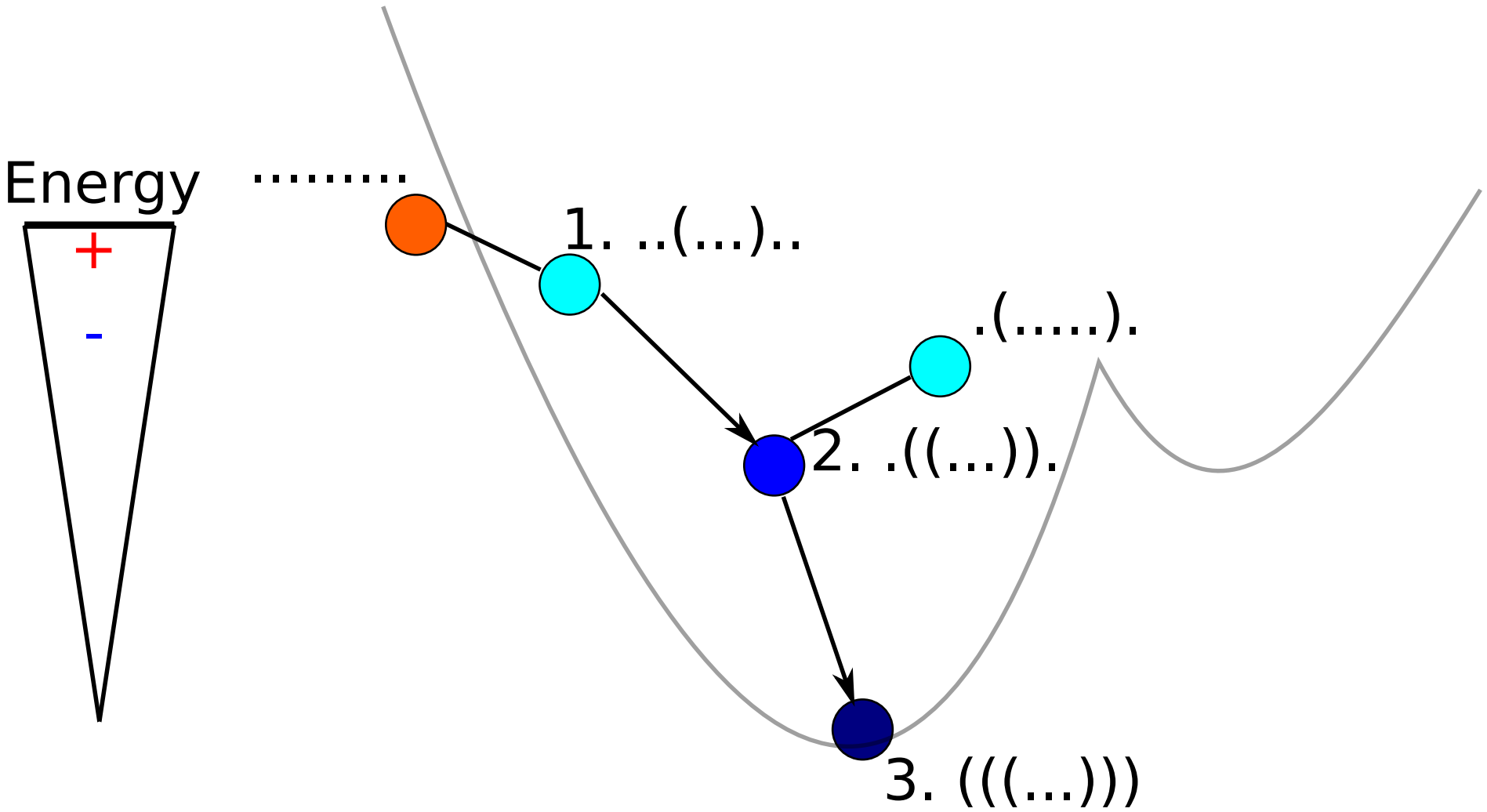




RNA Energy Landscape

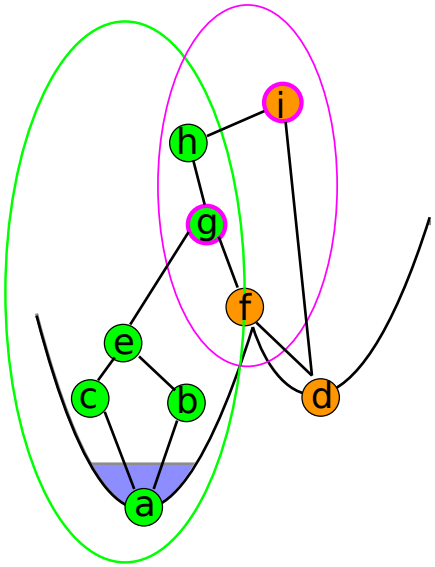


Gradient Walk



$3. = gradientwalk(1.)$

Global Flooding Algorithm - *barriers*



Structures sorted by energy

i
h
g
f
e
d
c
b
a

Basin Hash Table

a --> a

x=a

m = lowest neighbor from neighbors(x)

is E(m) > E(x)?

yes

no

x is a new basin

Z_k += w(x)

hash_write(x, k)

k = hash_lookup(m)

Z_k += w(x)

For all m in neighbors(x) with E(m) < E(x):

o = hash_lookup(m)

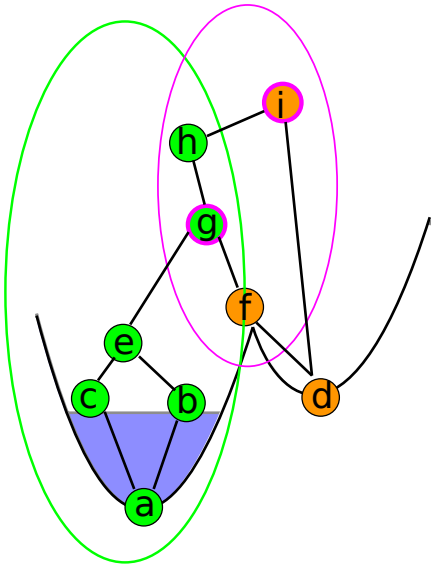
If o is not k: x in contact surface

$Z_{\{o,k\}}$ += w(x)

hash_write(x, k)

(Wolfinger et al. 2004)

Global Flooding Algorithm - *barriers*



Structures sorted by energy

i
h
g
f
e
d
c
b

Basin Hash Table

a --> a
b --> a

x=b

m = lowest neighbor from neighbors(x)

is E(m) > E(x)?

yes

x is a new basin

Z_k += w(x)

hash_write(x, k)

no

k = hash_lookup(m)

Z_k += w(x)

For all m in neighbors(x) with E(m) < E(x):

o = hash_lookup(m)

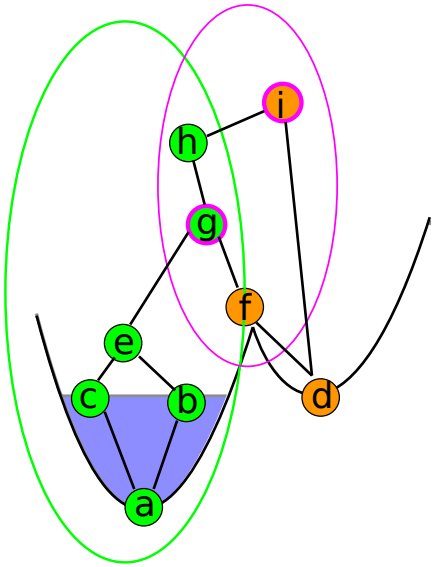
If o is not k: x in contact surface

$Z_{\{o,k\}}$ += w(x)

hash_write(x, k)

(Wolfinger et al. 2004)

Global Flooding Algorithm - *barriers*



Structures sorted by energy

i
h
g
f
e
d
c

Basin Hash Table

a --> a
b --> a
c --> a

x=c

m = lowest neighbor from neighbors(x)

is E(m) > E(x)?

yes

x is a new basin

Z_k += w(x)

hash_write(x, k)

no

k = hash_lookup(m)

Z_k += w(x)

For all m in neighbors(x) with E(m) < E(x):

o = hash_lookup(m)

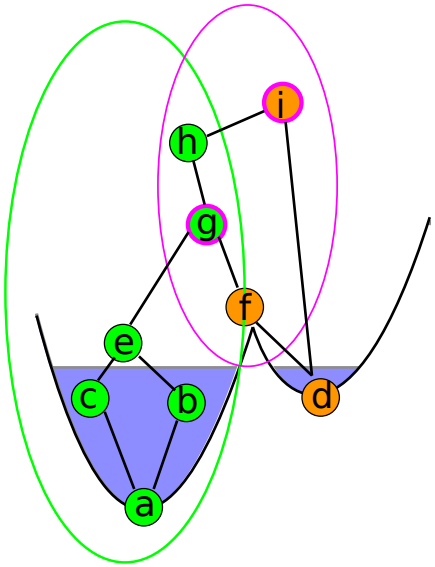
If o is not k: x in contact surface

$Z_{\{o,k\}}$ += w(x)

hash_write(x, k)

(Wolfinger et al. 2004)

Global Flooding Algorithm - *barriers*



Structures sorted by energy

i
h
g
f
e
d

Basin Hash Table

a --> a
b --> a
c --> a
d --> d

x=d

m = lowest neighbor from neighbors(x)

is E(m) > E(x)?

yes

x is a new basin

Z_k += w(x)

hash_write(x, k)

no

k = hash_lookup(m)

Z_k += w(x)

For all m in neighbors(x) with E(m) < E(x):

o = hash_lookup(m)

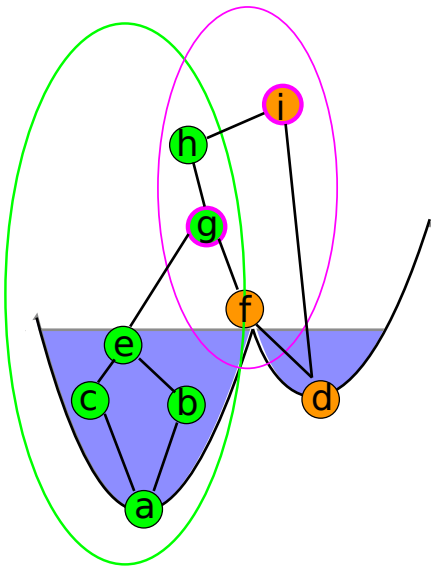
If o is not k: x in contact surface

$Z_{\{o,k\}}$ += w(x)

hash_write(x, k)

(Wolfinger et al. 2004)

Global Flooding Algorithm - *barriers*



Structures sorted by energy

i
h
g
f
e

Basin Hash Table

a --> a
b --> a
c --> a
d --> d
e --> a

x=e

m = lowest neighbor from neighbors(x)

is E(m) > E(x)?

yes

x is a new basin

Z_k += w(x)

hash_write(x, k)

no

k = hash_lookup(m)

Z_k += w(x)

For all m in neighbors(x) with E(m) < E(x):

o = hash_lookup(m)

If o is not k:

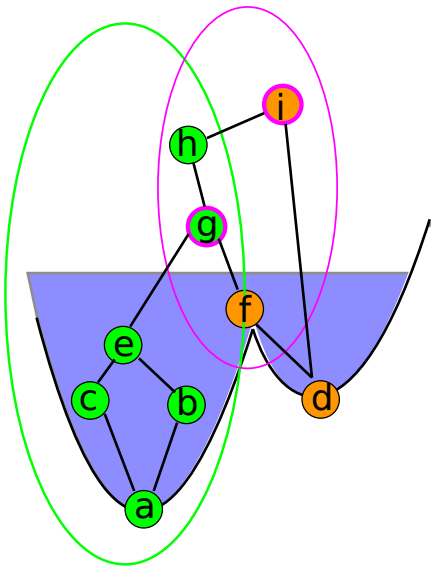
$Z_{\{o,k\}}$ += w(x)

hash_write(x, k)

x in contact surface

(Wolfinger et al. 2004)

Global Flooding Algorithm - *barriers*



Structures sorted by energy

i
h
g
f

Basin Hash Table

a --> a
b --> a
c --> a
d --> d
e --> a
f --> d

x=f

m = lowest neighbor from neighbors(x)

is E(m) > E(x)?

yes

x is a new basin

Z_k += w(x)

hash_write(x, k)

no

k = hash_lookup(m)

Z_k += w(x)

For all m in neighbors(x) with E(m) < E(x):

o = hash_lookup(m)

If o is not k:

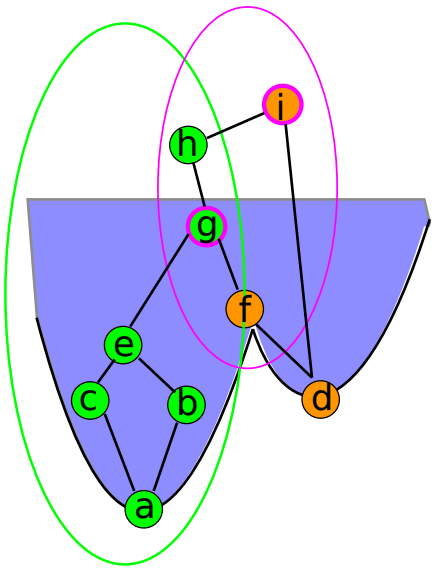
$Z_{\{o,k\}}$ += w(x)

x in contact surface

hash_write(x, k)

(Wolfinger et al. 2004)

Global Flooding Algorithm - *barriers*



Structures sorted by energy

i
h
g

Basin Hash Table

a --> a
b --> a
c --> a
d --> d
e --> a
f --> d
g --> a

x=g

m = lowest neighbor from neighbors(x)

is E(m) > E(x)?

yes

x is a new basin

Z_k += w(x)

hash_write(x, k)

no

k = hash_lookup(m)

Z_k += w(x)

For all m in neighbors(x) with E(m) < E(x):

o = hash_lookup(m)

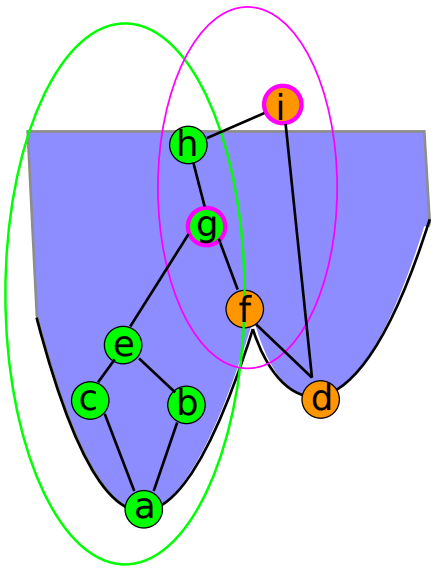
If o is not k: x in contact surface

$Z_{\{o,k\}}$ += w(x)

hash_write(x, k)

(Wolfinger et al. 2004)

Global Flooding Algorithm - *barriers*



Structures sorted by energy

i
h

Basin Hash Table

a --> a
b --> a
c --> a
d --> d
e --> a
f --> d
g --> a
h --> a

x=h

m = lowest neighbor from neighbors(x)

is E(m) > E(x)?

yes

x is a new basin

Z_k += w(x)

hash_write(x, k)

no

k = hash_lookup(m)

Z_k += w(x)

For all m in neighbors(x) with E(m) < E(x):

o = hash_lookup(m)

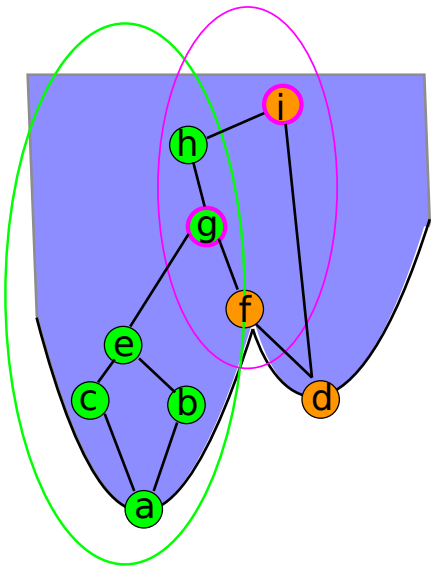
If o is not k: x in contact surface

$Z_{\{o,k\}}$ += w(x)

hash_write(x, k)

(Wolfinger et al. 2004)

Global Flooding Algorithm - *barriers*



Structures sorted by energy

i

Basin Hash Table

a --> a
b --> a
c --> a
d --> d
e --> a
f --> d
g --> a
h --> a
i --> a

x=i

m = lowest neighbor from neighbors(x)

is E(m) > E(x)?

yes

x is a new basin

Z_k += w(x)

hash_write(x, k)

no

k = hash_lookup(m)

Z_k += w(x)

For all m in neighbors(x) with E(m) < E(x):

o = hash_lookup(m)

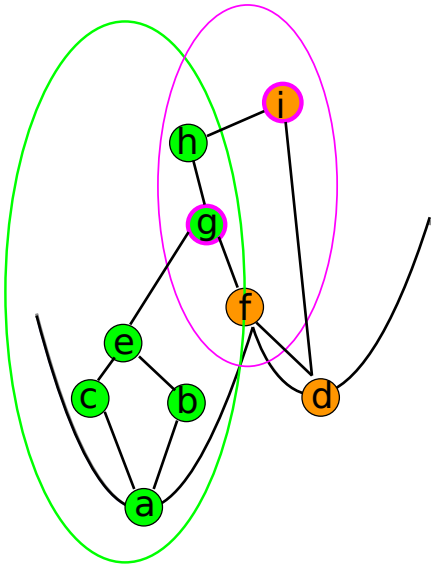
If o is not k: x in contact surface

$Z_{\{o,k\}}$ += w(x)

hash_write(x, k)

(Wolfinger et al. 2004)

Local Flooding Algorithm - *pourRNA*



TodoList (PQ)

DoneList

y = neighbor where E(m)
is minimal for all m in neighbors(x)
y in DoneList?

yes

no

x in Basin
Z_b += w(x)
DoneList += [x]

For all m in neighbors(x):
  If E(m) > E(x):
    TodoList += m

Descending transitions leaving b:

For all m in neighbors(x):
  If E(m) < E(x) and m not in DoneList:
    $Z_{b,\ \text{grad. walk(m)}}$ += w(x)

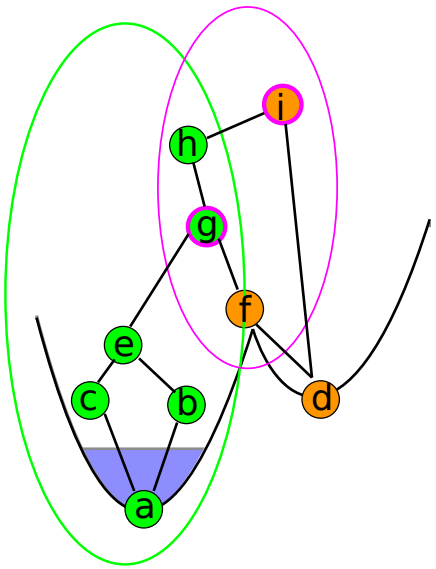
x in Contact Surface

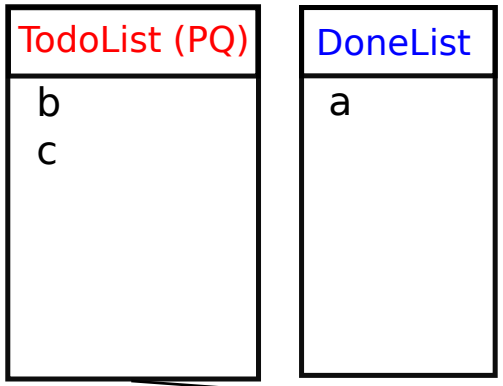
Descending transitions entering b:

For all m in neighbors(x):
  If E(m) < E(x) and m in DoneList:
    $Z_{\text{grad. walk(x)},b}$ += w(x)

(Mann et al. 2014)

Local Flooding Algorithm - *pourRNA*





initalization

x = a

yes

no

x in Basin

Z_b += w(x)

DoneList += [x]

For all m in neighbors(x):

If E(m) > E(x):

TodoList += m

Descending transitions leaving b:

For all m in neighbors(x):

If E(m) < E(x) and m not in DoneList:

$Z_{b,\ \text{grad. walk(m)}}$ += w(x)

x in Contact Surface

Descending transitions entering b:

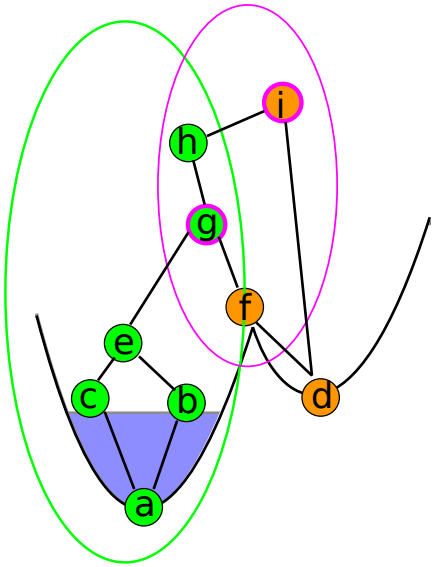
For all m in neighbors(x):

If E(m) < E(x) and m in DoneList:

$Z_{\text{grad. walk(x)},b}$ += w(x)

(Mann et al. 2014)

Local Flooding Algorithm - *pourRNA*



TodoList (PQ)	DoneList
c	a
e	b

x = b

y = neighbor where E(m)
is minimal for all m in neighbors(x)
y in DoneList?

yes

x in Basin
Z_b += w(x)
DoneList += [x]
For all m in neighbors(x):
  If E(m) > E(x):
    TodoList += m
Descending transitions leaving b:
For all m in neighbors(x):
  If E(m) < E(x) and m not in DoneList:
    $Z_{b,\ \text{grad. walk(m)}}$ += w(x)

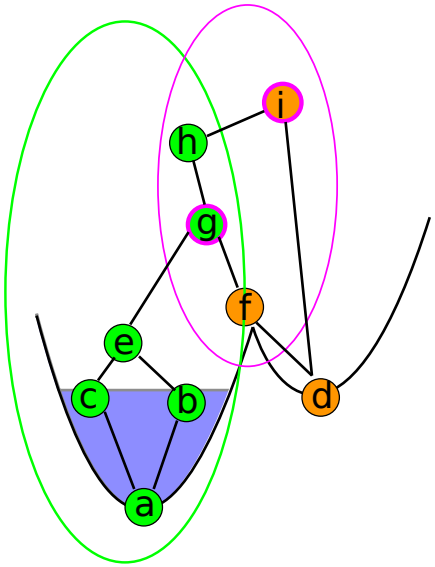
no

x in Contact Surface

Descending transitions entering b:
For all m in neighbors(x):
  If E(m) < E(x) and m in DoneList:
    $Z_{\text{grad. walk(x)},b}$ += w(x)

(Mann et al. 2014)

Local Flooding Algorithm - *pourRNA*



TodoList (PQ)
e

DoneList
a
b
c

x = c

y = neighbor where E(m)
is minimal for all m in neighbors(x)
y in DoneList?

yes

x in Basin
Z_b += w(x)
DoneList += [x]
For all m in neighbors(x):
  If E(m) > E(x):
    TodoList += m
Descending transitions leaving b:
For all m in neighbors(x):
  If E(m) < E(x) and m not in DoneList:
    $Z_{b,\ \text{grad. walk(m)}}$ += w(x)

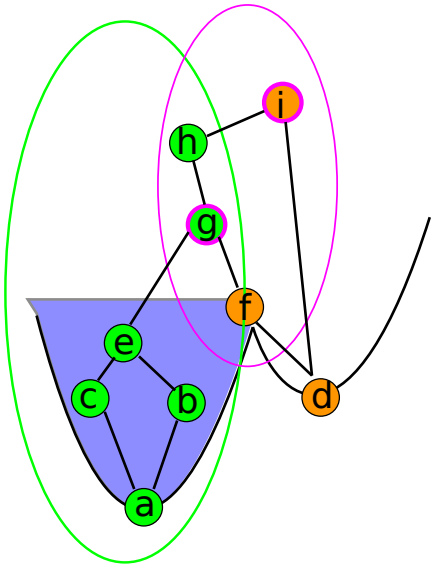
no

x in Contact Surface

Descending transitions entering b:
For all m in neighbors(x):
  If E(m) < E(x) and m in DoneList:
    $Z_{\text{grad. walk(x)},b}$ += w(x)

(Mann et al. 2014)

Local Flooding Algorithm - *pourRNA*



TodoList (PQ)

g

DoneList

a
b
c
e

x = e

y = neighbor where E(m)
is minimal for all m in neighbors(x)
y in DoneList?

yes

no

x in Basin

Z_b += w(x)

DoneList += [x]

For all m in neighbors(x):

If E(m) > E(x):

TodoList += m

Descending transitions leaving b:

For all m in neighbors(x):

If E(m) < E(x) and m not in DoneList:

$Z_{b,\ \text{grad. walk(m)}}$ += w(x)

x in Contact Surface

Descending transitions entering b:

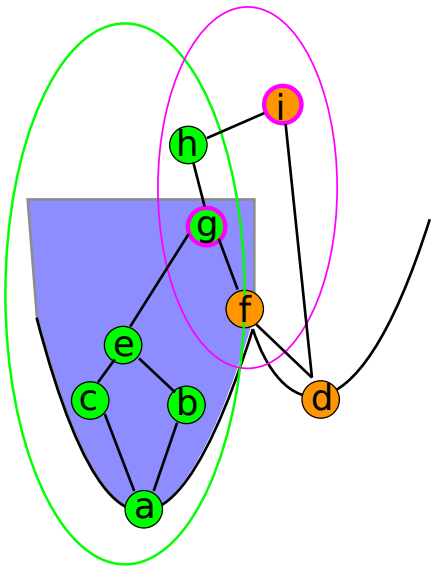
For all m in neighbors(x):

If E(m) < E(x) and m in DoneList:

$Z_{\text{grad. walk(x)},b}$ += w(x)

(Mann et al. 2014)

Local Flooding Algorithm - *pourRNA*



TodoList (PQ)	DoneList
h	a b c e g

x = g

y = neighbor where E(m)
is minimal for all m in neighbors(x)
y in DoneList?

yes

x in Basin

Z_b += w(x)

DoneList += [x]

For all m in neighbors(x):

If E(m) > E(x):

TodoList += m

Descending transitions leaving b:

For all m in neighbors(x):

If E(m) < E(x) and m not in DoneList:

$Z_{b,\ \text{grad. walk(m)}}$ += w(x)

no

x in Contact Surface

Descending transitions entering b:

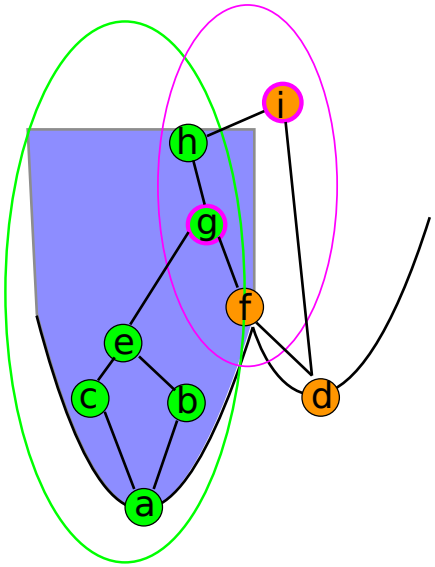
For all m in neighbors(x):

If E(m) < E(x) and m in DoneList:

$Z_{\text{grad. walk(x)},b}$ += w(x)

(Mann et al. 2014)

Local Flooding Algorithm - *pourRNA*



TodoList (PQ)	DoneList
i	a
	b
	c
	e
	g
	h

x = h

y = neighbor where E(m)
is minimal for all m in neighbors(x)
y in DoneList?

yes

x in Basin

Z_b += w(x)

DoneList += [x]

For all m in neighbors(x):

If E(m) > E(x):

TodoList += m

Descending transitions leaving b:

For all m in neighbors(x):

If E(m) < E(x) and m not in DoneList:

$Z_{b,\ \text{grad. walk(m)}}$ += w(x)

no

x in Contact Surface

Descending transitions entering b:

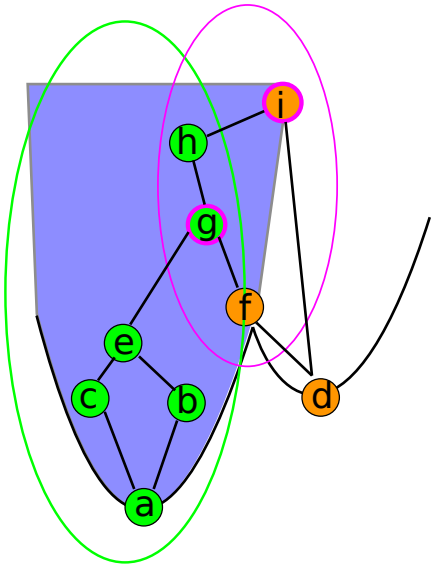
For all m in neighbors(x):

If E(m) < E(x) and m in DoneList:

$Z_{\text{grad. walk(x)},b}$ += w(x)

(Mann et al. 2014)

Local Flooding Algorithm - *pourRNA*



TodoList (PQ)

DoneList
a
b
c
e
g
h

x = i

y = neighbor where E(m)
is minimal for all m in neighbors(x)
y in DoneList?

yes

x in Basin
Z_b += w(x)
DoneList += [x]

For all m in neighbors(x):
  If E(m) > E(x):
    TodoList += m

Descending transitions leaving b:

For all m in neighbors(x):
  If E(m) < E(x) and m not in DoneList:
    $Z_{b,\ \text{grad. walk(m)}}$ += w(x)

no

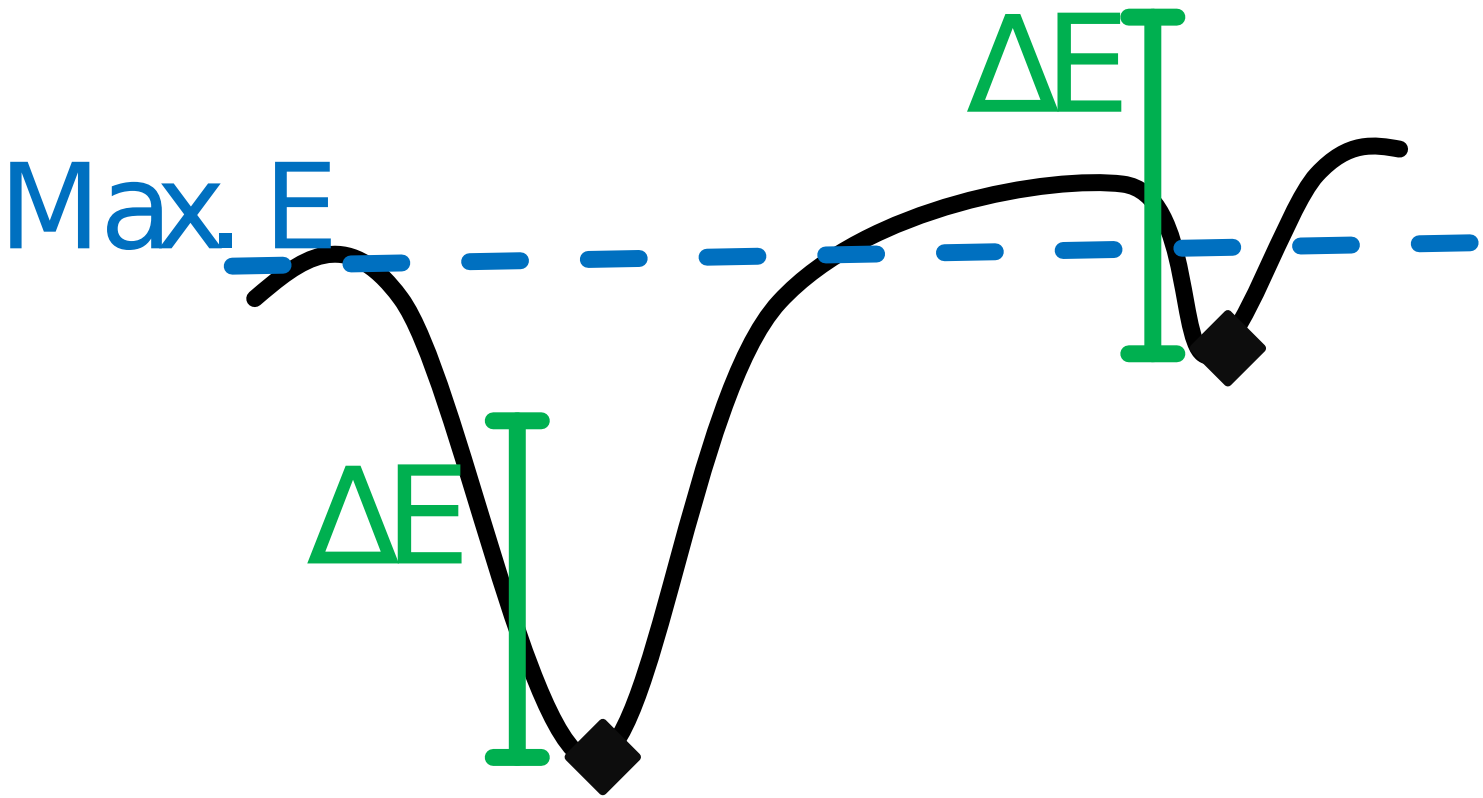
x in Contact Surface

Descending transitions entering b:

For all m in neighbors(x):
  If E(m) < E(x) and m in DoneList:
    $Z_{\text{grad. walk(x)},b}$ += w(x)

(Mann et al. 2014)

Filter - Maximum and Delta Energy

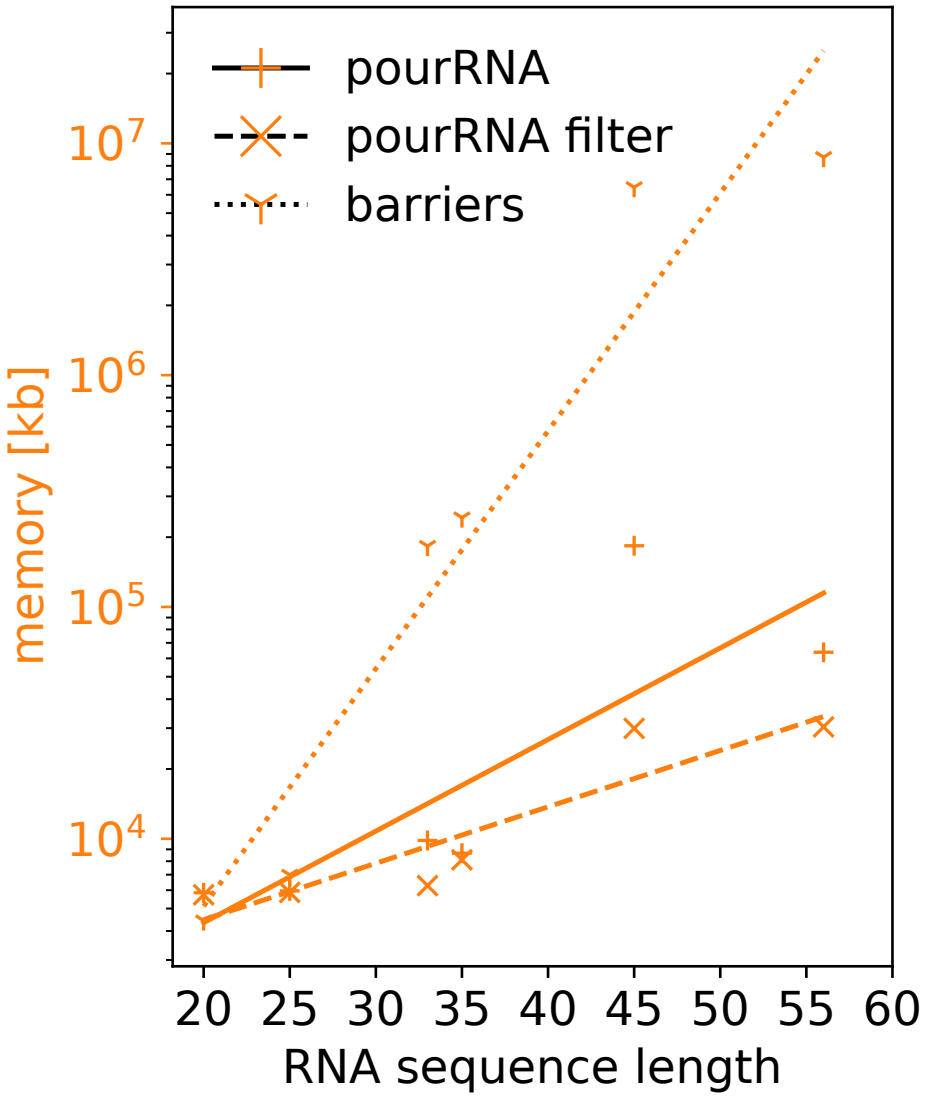


Memory Trend Lines

9 sequences

sequence	length
xbix	20
boris1	20
boris2	20
d25	25
d33	33
ire	35
bhg33	33
d45	45
SL (stem 1)	56

filter: $\Delta E = 5 kcal/mol$

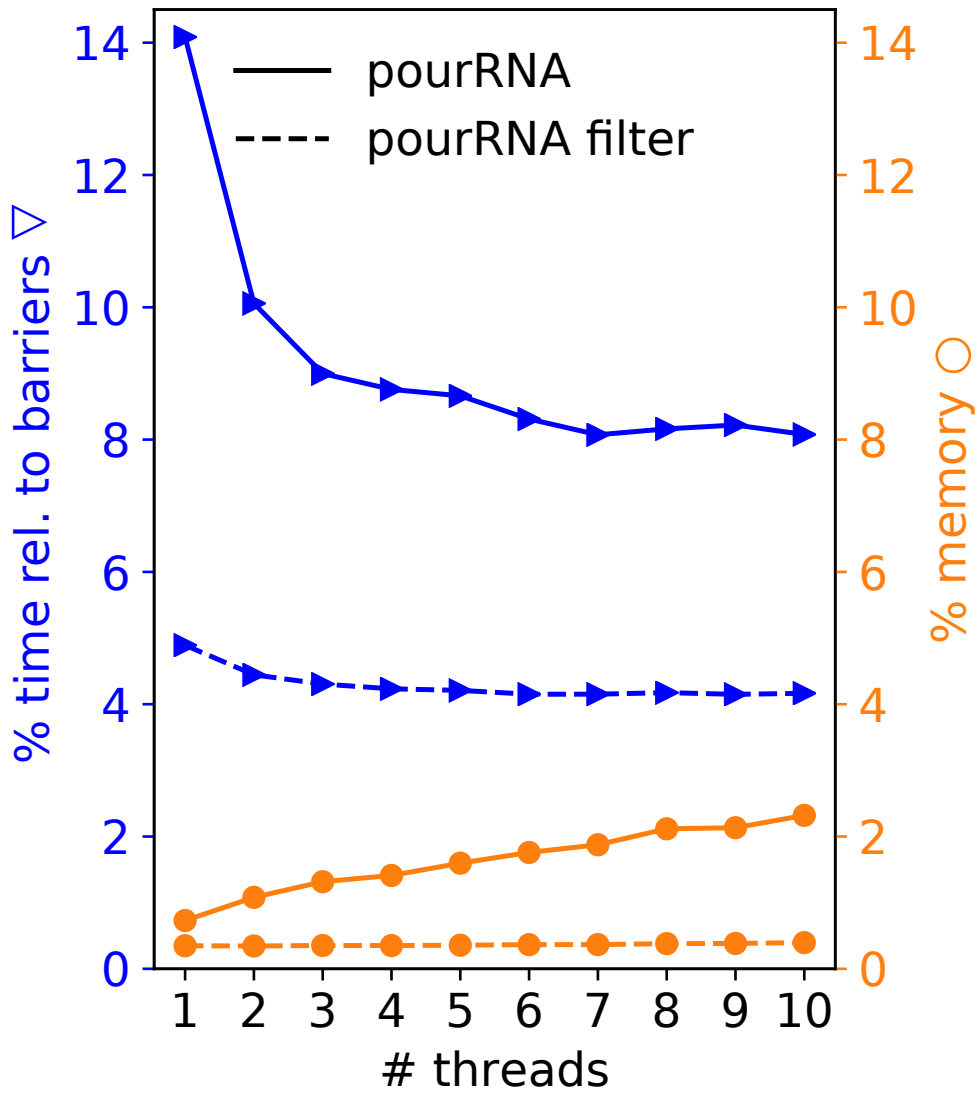


Memory peak per sequence length

More Threads: Time and Memory

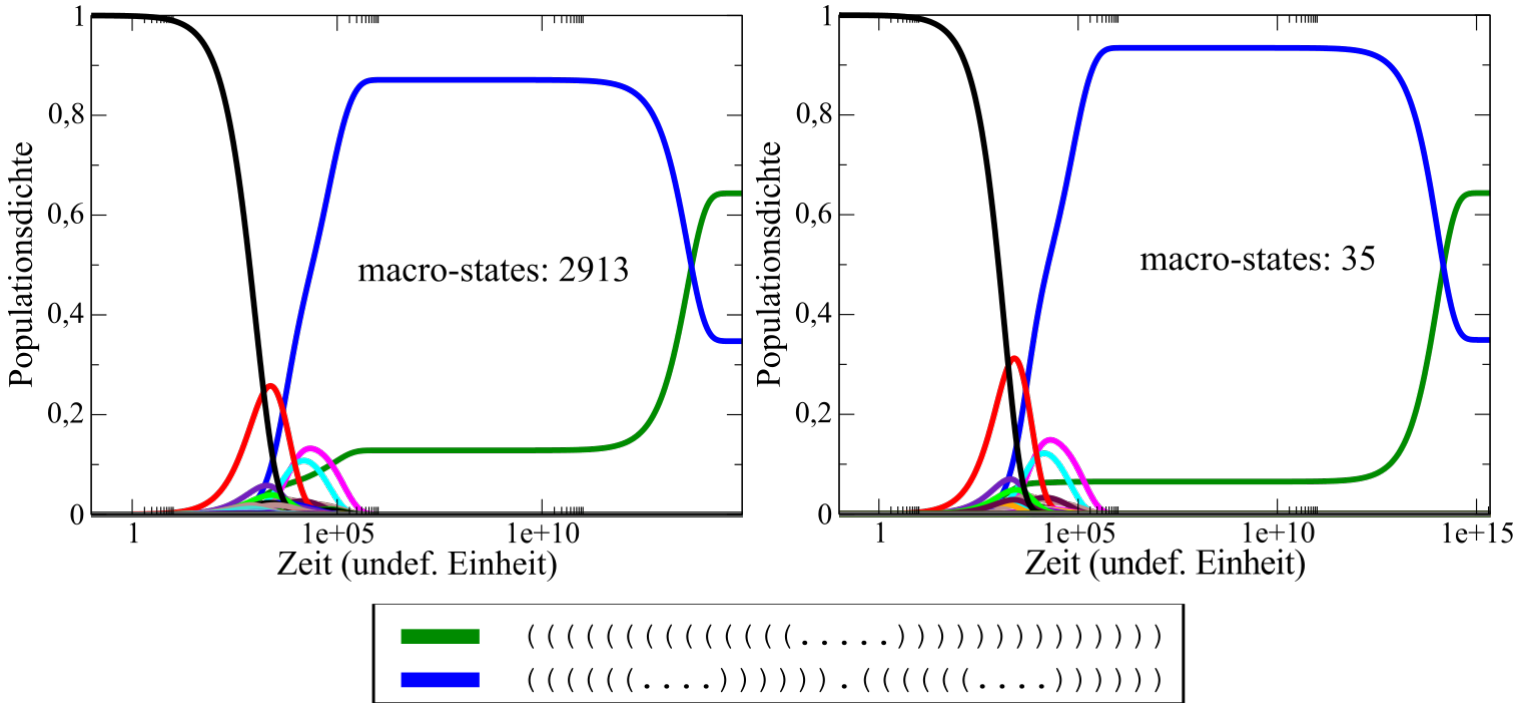
Sequence: Spliced Leader
Nucleotides: 56
1 Thread,
max. Energy = 5kcal/mol:

	barriers	pourRNA
memory [GB]	8.31	0.06
time [m]	44.9	6.3



filter: $\Delta E = 5kcal/mol$

Approximate Kinetics Quality



unfiltered

Computing time: 0.5h

Delta-Energie filter with 10 kcal/mol ,
K-Best filter = 4 ,
DeltaMinE filter = 1

Computing time: 0.5m

Advantages and Disadvantages

	barriers	**pourRNA**
optimization focus	runtime	memory
runtime	$\mathcal{O}(N * n)$ † ‡	$\mathcal{O}(N * n * (grad.walks))$†
parallelization	difficult	easy
bottleneck	memory	time for gradient walks, contact surfaces
local filters	difficult	easy
detects missing sublandscape	yes	no
connected component only	no	yes
many start structures	no	yes

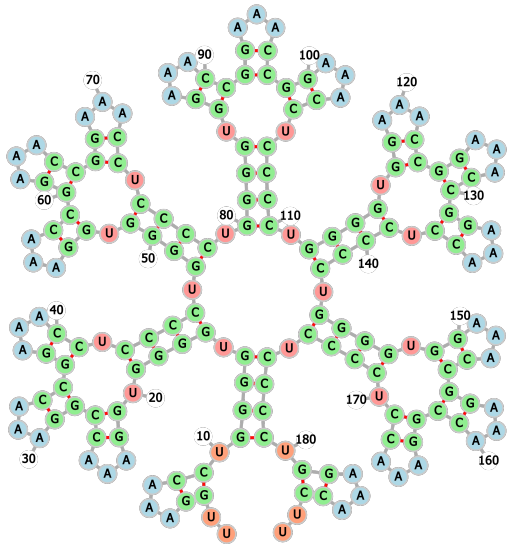
†N = all secondary structures, n = number of neighbors

‡RNAsubopt $\mathcal{O}(m^3 + mL + L \log L)$

m = sequence length, L = number of lowest structures

Acknowledgements

- Martin Raden
- Team Freiburg
- Team Vienna



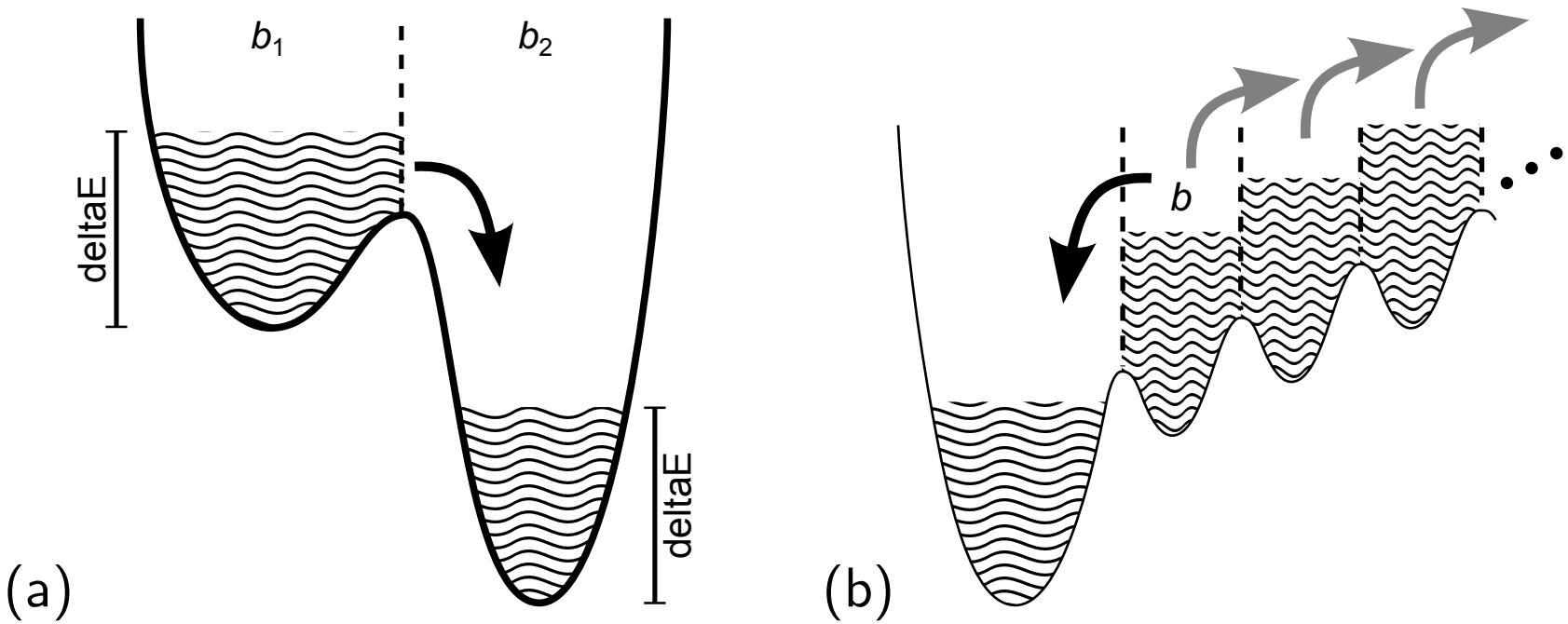
Thank you!

pourRNA

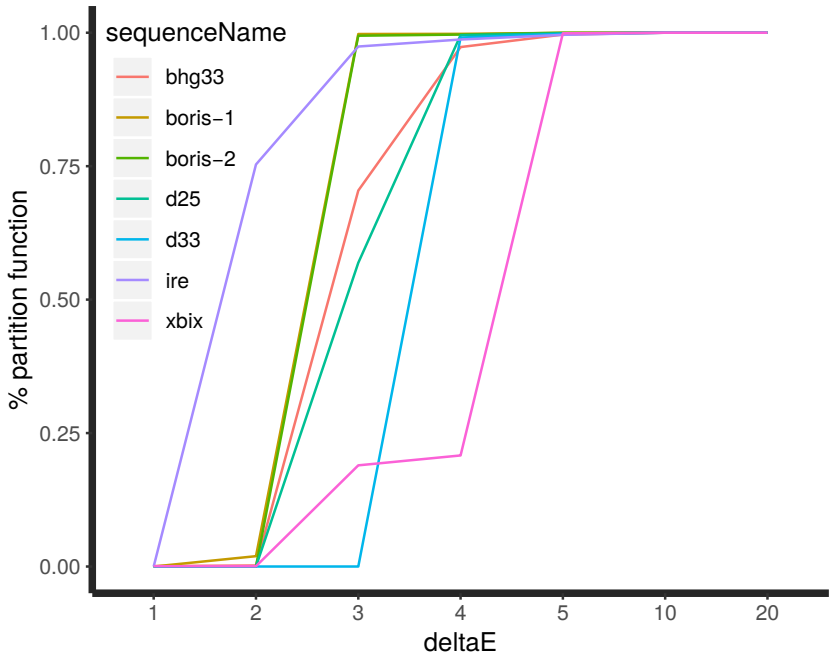
`https://github.com/ViennaRNA/pourRNA`

tbi BOKU FWF Der Wissenschaftsfonds.

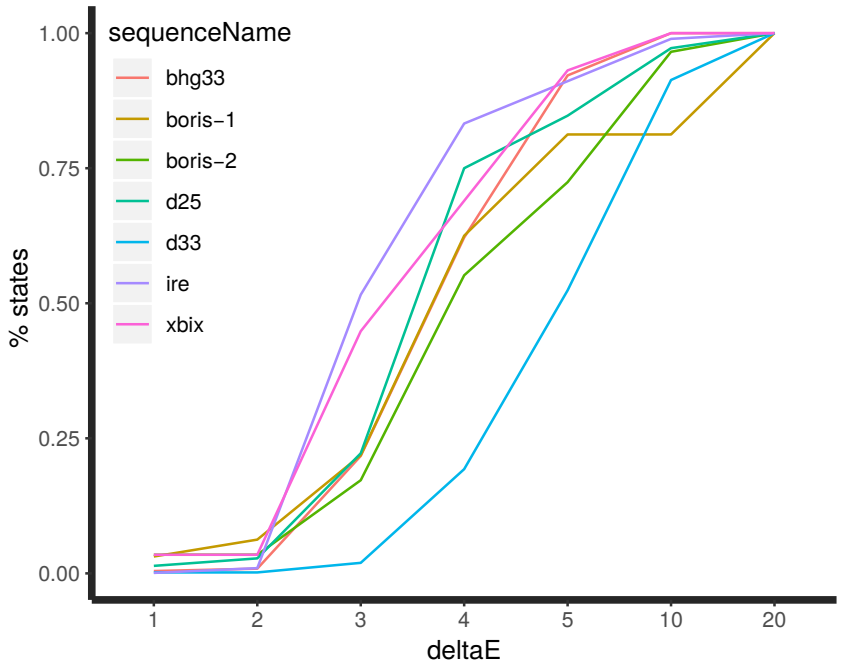
Delta Energy Stair Climb Effect



Delta Energy Filter - Partition Function and Number of States

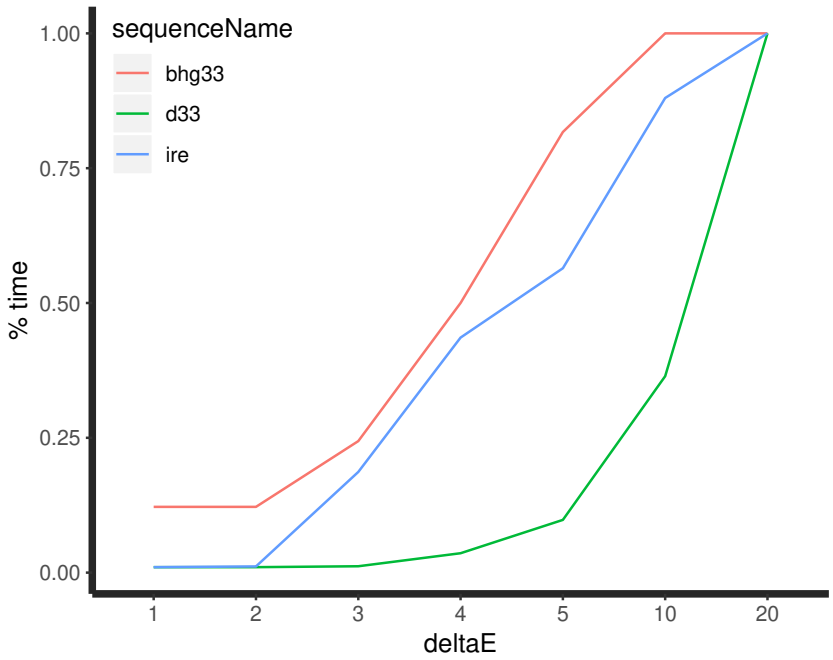


(d) Partition Function

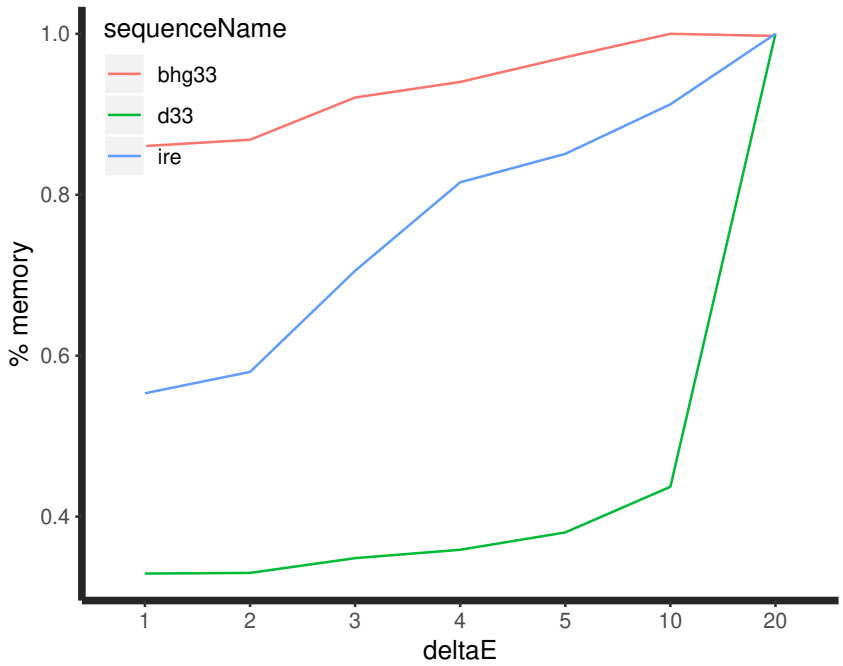


(e) Number of States

Delta Energy Filter - Time and Memory

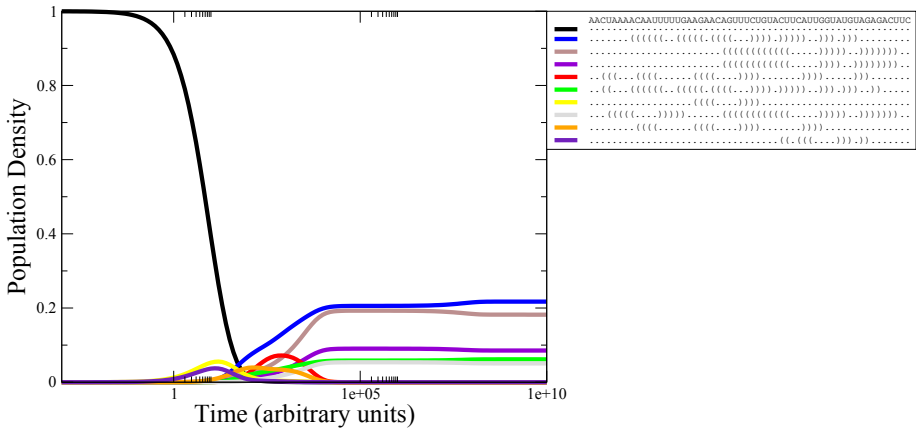


(f) Partition Function

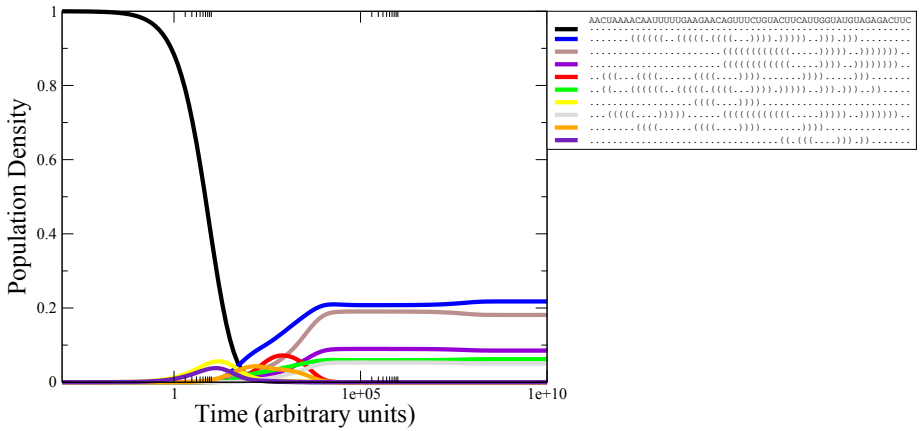


(g) Number of States

SL - DeltaE - Quality



(h) Exact kinetics



(i) Approximate kinetics using deltaE=5

Figure: Folding kinetics of SL when using (a) exact macroscopic transition rates and (b) approximate rates resulting from restricted local flooding with deltaE=5$kcal/mol$.