Nussinov-Algorithmus

Peter N. Robinson

Institut für medizinische Genetik
Charité Universitätsmedizin Berlin

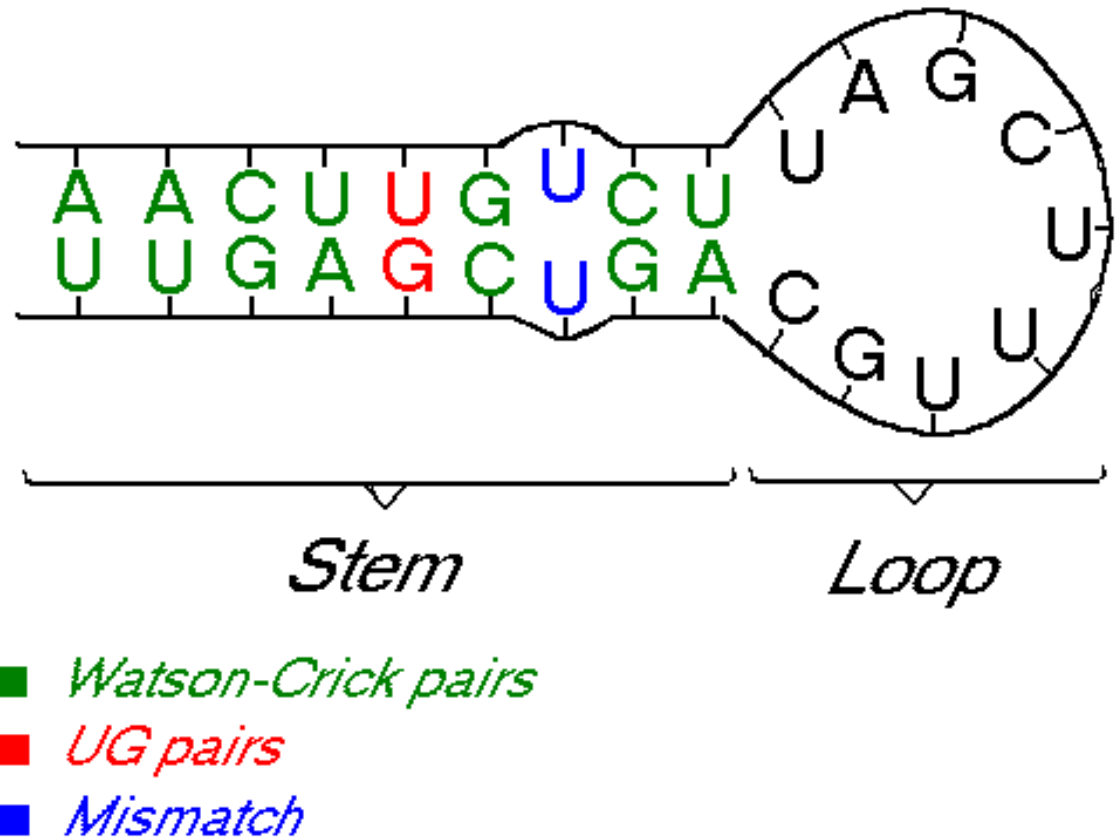
18. Dezember 2015

Bioinformatik der RNA-Faltung

- Zahlreiche Algorithmen
- Dynamic programming
- Freie Energie
- Hier stellen wir einen vereinfachten DP-Algorithmus vor (Nussinov)

Bioinformatik der RNA-Faltung

- Dominiert wird eine RNA Struktur von den Basenpaaren die sich zwischen komplementären Basen bilden.
- Die meisten dieser Basenpaarungen sind Watson-Crick Basenpaare.
- “Palindrome” häufig



Bioinformatik der RNA-Faltung

- Eine vereinfachte Version des Nussinov-Algorithmus versucht, die Anzahl der gepaarten Basen zu maximieren
- Unser Score: +1 für Basenpaar, 0 für alles Andere
- Wir betrachten eine RNA-Sequenz $1, 2, \ldots, n$

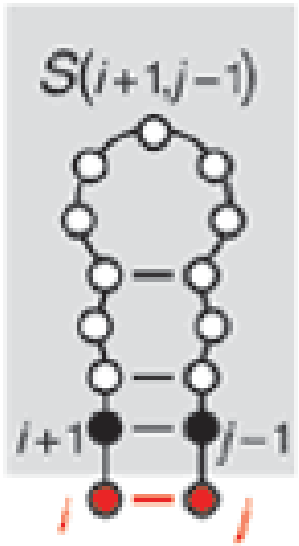
$\boxed{S(i,j)}$ Max. Score für die Subsequenz $i, i+1, \ldots, j$.

- $S(i,j)$ kann rekursiv berechnet werden (Dynamic programming)

RNA-Faltung: DP (1)

- Falls i, j ein WC-Baasenpaar sind

$$S^1(i,j) = 1 + S(i+1, j-1)$$

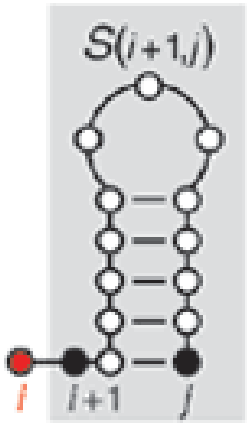


1. *i,j* pair

RNA-Faltung: DP (2)

- Falls i ungepaart bleibt

$$S^2(i,j) = S(i+1,j)$$

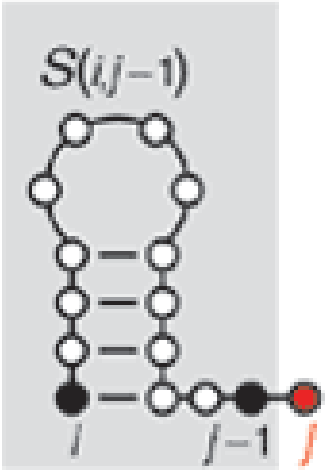


2. i unpaired

RNA-Faltung: DP (3)

- Falls j ungepaart bleibt

$$S^3(i,j) = S(i,j-1)$$

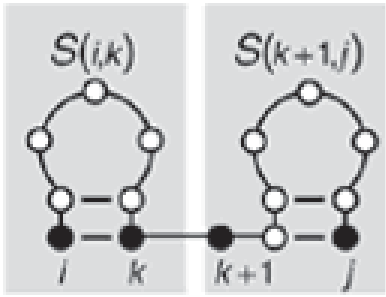


3. j unpaired

RNA-Faltung: DP (4)

- Falls i, j jeweils mit anderen Nukleotiden gepaart sind, handelt es sich um eine Bifurkation, die Struktur $S(i,j)$ besteht dann aus den Strukturen für zwei Subsequenzen $i, \ldots, k$ und $k+1, \ldots, j$:

$$S^4(i,j) = \max_{i<k<j} S(i,k) + S(k+1,j)$$



4. Bifurcation

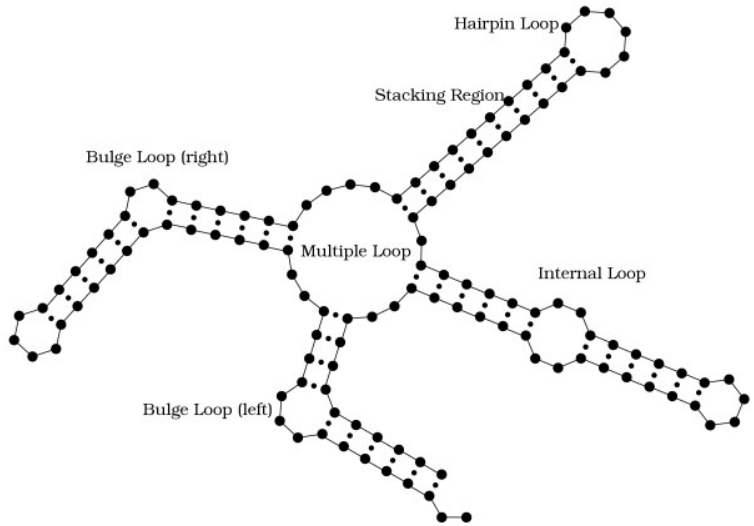
RNA-Faltung: Dynamic programming

- Falls i und j also ein Basenpaar bilden, wird dem Score ein Punkt hinzugefügt, ganz egal was die Struktur der Subsequenz $i+1,\ldots,j-1$ ist
- Daher müssen wir den Score für $S(i+1,j-1)$ nicht neu berechnen
- Ähnliche Argumente gelten für die anderen drei Möglichkeiten
- Der optimale Score $S(i,j)$ ist daher lediglich das Maximum der vier Optionen

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

RNA-Faltung: Dynamic programming

- Realistischere Algorithmen betrachten Stems, Haarnadelstrukturen, Bulges, innere Schleifen, auch Pseudoknoten



Grafik: Steffen P, Giegerich R (2005) *BMC Bioinformatics* **6**:224.

Initialization

1: $s = newMatrix(n, n)$
2: **for** $i \leftarrow 2$ to n **do**
3: $\quad s[i, i-1] \leftarrow 0$
4: **end for**
5: **for** $i \leftarrow 1$ to n **do**
6: $\quad s[i, i] \leftarrow 0$
7: **end for**

- DP-Matrix initialisieren
- Nur die Hälfte der Matrix wird verwendet
- Wir analysieren die Faltung der Sequence GGGAAAUCC

	G_1	G_2	G_3	A_4	A_5	A_6	U_7	C_8	C_9
G_1	0								
G_2	0	0							
G_3		0	0						
A_4			0	0					
A_5				0	0				
A_6					0	0			
U_7						0	0		
C_8							0	0	
C_9								0	0

Füllen der Matrix (1)

1: **for** $s \leftarrow 2$ to n **do**
2: $\quad i \leftarrow 0$
3: $\quad$ **for** $j \leftarrow s$ to n **do**
4: $\quad\quad i \leftarrow i+1$
5: $\quad\quad s[i,j] \leftarrow \max(a,b,c,d)$
6: $\quad$ **end for**
7: **end for**

- Die For-Schleifen besuchen sukzessive die Nebendiagonalen
- Die DP-Rekursion prüft vier Möglichkeiten (a,b,c,d):

$$S(i,j) = \max \begin{cases} \text{a)} & \delta(i,j) + S(i+1,j-1) \\ \text{b)} & S(i+1,j) \\ \text{c)} & S(i,j-1) \\ \text{d)} & \max_{i<k<j} S(i,k) + S(k+1,j) \end{cases}$$

	G_1	G_2	G_3	A_4	A_5	A_6	U_7	C_8	C_9
G_1	0	?							
G_2	0	0							
G_3		0	0						
A_4			0	0					
A_5				0	0				
A_6					0	0			
U_7						0	0		
C_8							0	0	
C_9								0	0

Füllen der Matrix (2)

1: **for** $s \leftarrow 2$ to n **do**
2: $\quad i \leftarrow 0$
3: $\quad$ **for** $j \leftarrow s$ to n **do**
4: $\quad\quad i \leftarrow i+1$
5: $\quad\quad s[i,j] \leftarrow \max(a,b,c,d)$
6: $\quad$ **end for**
7: **end for**

- Die DP-Rekursion prüft vier Möglichkeiten (a,b,c,d)

$$S(i,j) = \max \begin{cases} \text{a)} & \delta(i,j) + S(i+1,j-1) \\ \text{b)} & S(i+1,j) \\ \text{c)} & S(i,j-1) \\ \text{d)} & \max_{i<k<j} S(i,k) + S(k+1,j) \end{cases}$$

	G_1	G_2	G_3	A_4	A_5	A_6	U_7	C_8	C_9
G_1	0	?							
G_2	0	0							
G_3		0	0						
A_4			0	0					
A_5				0	0				
A_6					0	0			
U_7						0	0		
C_8							0	0	
C_9								0	0

- Option a) prüft, ob eine Basenpaarung der Nukleotiden i und j gibt ($\delta(i,j)$).
- Prüfe, ob die Basenpaarung $i.j$ zusammen mit dem vorausgegangen Alignment in $s[i+1,j-1]$ (in blau) einen maximalen Score hat
- $\delta(2,1)$ ergibt 0, weil G_2 und G_1 nicht paaren können.
- $\delta(i,j) + s[i+1,j-1] = \delta(1,2) + s[2,1] = 0+0 = 0$.

Füllen der Matrix (3)

1: **for** $s \leftarrow 2$ to n **do**
2: $\quad i \leftarrow 0$
3: $\quad$ **for** $j \leftarrow s$ to n **do**
4: $\quad\quad i \leftarrow i+1$
5: $\quad\quad s[i,j] \leftarrow \max(a,b,c,d)$
6: $\quad$ **end for**
7: **end for**

- Die DP-Rekursion prüft vier Möglichkeiten (a,b,c,d)

$$S(i,j) = \max \begin{cases} \text{a)} & \delta(i,j) + S(i+1,j-1) \\ \text{b)} & S(i+1,j) \\ \text{c)} & S(i,j-1) \\ \text{d)} & \max_{i<k<j} S(i,k) + S(k+1,j) \end{cases}$$

	G_1	G_2	G_3	A_4	A_5	A_6	U_7	C_8	C_9
G_1	0	?							
G_2	0	0							
G_3		0	0						
A_4			0	0					
A_5				0	0				
A_6					0	0			
U_7						0	0		
C_8							0	0	
C_9								0	0

- Option b) prüft, ob das vorausgegangene Alignment $s[i+1,j]$ (in blau) zusammen mit einem ungepaarten Nukleotid i einen maximalen Score hat
- $s[i+1,j] = s[2,2] = 0$.

Füllen der Matrix (4)

```
1: for s ← 2 to n do
2:     i ← 0
3:     for j ← s to n do
4:         i ← i + 1
5:         s[i, j] ← max(a, b, c, d)
6:     end for
7: end for
```

- Die DP-Rekursion prüft vier Möglichkeiten (a,b,c,d)

$$S(i,j) = \max \begin{cases} \text{a)} & \delta(i,j) + S(i+1,j-1) \\ \text{b)} & S(i+1,j) \\ \text{c)} & S(i,j-1) \\ \text{d)} & \max_{i<k<j} S(i,k) + S(k+1,j) \end{cases}$$

	G_1	G_2	G_3	A_4	A_5	A_6	U_7	C_8	C_9
G_1	0	?							
G_2	0	0							
G_3		0	0						
A_4			0	0					
A_5				0	0				
A_6					0	0			
U_7						0	0		
C_8							0	0	
C_9								0	0

- Option c) prüft, ob das vorausgegangene Alignment $s[i,j-1]$ (in blau) zusammen mit einem ungepaarten Nukleotid j einen maximalen Score hat
- $s[i,j-1] = s[1,1] = 0$.

Füllen der Matrix (5)

1: **for** $s \leftarrow 2$ to n **do**
2: $\quad i \leftarrow 0$
3: $\quad$ **for** $j \leftarrow s$ to n **do**
4: $\quad\quad i \leftarrow i+1$
5: $\quad\quad s[i,j] \leftarrow \max(a,b,c,d)$
6: $\quad$ **end for**
7: **end for**

- Die DP-Rekursion prüft vier Möglichkeiten (a,b,c,d)

$$S(i,j) = \max \begin{cases} \text{a)} & \delta(i,j) + S(i+1,j-1) \\ \text{b)} & S(i+1,j) \\ \text{c)} & S(i,j-1) \\ \text{d)} & \max_{i<k<j} S(i,k) + S(k+1,j) \end{cases}$$

	G_1	G_2	G_3	A_4	A_5	A_6	U_7	C_8	C_9
G_1	0	0							
G_2	0	0							
G_3		0	0						
A_4			0	0					
A_5				0	0				
A_6					0	0			
U_7						0	0		
C_8							0	0	
C_9								0	0

- Option d) ist in diesem Schritt nicht möglich, weil es bei $i = 1, j = 2$ keine Ganzzahl k gibt mit $i < k < j$.
- $s[2,1] = \max(a,b,c) = \max(0,0,0) = 0$

Füllen der Matrix (6)

```
1: for s ← 2 to n do
2:     i ← 0
3:     for j ← s to n do
4:         i ← i + 1
5:         s[i, j] ← max(a, b, c, d)
6:     end for
7: end for
```

$$S(i,j) = \max \begin{cases} \text{a)} & \delta(i,j) + S(i+1, j-1) \\ \text{b)} & S(i+1,j) \\ \text{c)} & S(i,j-1) \\ \text{d)} & \max_{i<k<j} S(i,k) + S(k+1,j) \end{cases}$$

	G_1	G_2	G_3	A_4	A_5	A_6	U_7	C_8	C_9
G_1	0	0							
G_2	0	0	0						
G_3		0	0	0					
A_4			0	0	0				
A_5				0	0	0			
A_6					0	0	?		
U_7						0	0		
C_8							0	0	
C_9								0	0

- Die folgenden Schritte sind ähnlich, bis wir Zelle $(6,7)$ erreichen
- $\delta(6,7) = 1$, weil A_6 mit U_7 eine Basenpaarung bilden kann
- Daher erreicht $\delta(6,7) + s[7,6]$ den maximalen Score von 1

Füllen der Matrix (7)

1: **for** $s \leftarrow 2$ to n **do**
2: $\quad i \leftarrow 0$
3: $\quad$ **for** $j \leftarrow s$ to n **do**
4: $\quad\quad i \leftarrow i+1$
5: $\quad\quad s[i,j] \leftarrow \max(a,b,c,d)$
6: $\quad$ **end for**
7: **end for**

$$S(i,j) = \max \begin{cases} \text{a)} & \delta(i,j) + S(i+1,j-1) \\ \text{b)} & S(i+1,j) \\ \text{c)} & S(i,j-1) \\ \text{d)} & \max_{i<k<j} S(i,k) + S(k+1,j) \end{cases}$$

	G_1	G_2	G_3	A_4	A_5	A_6	U_7	C_8	C_9
G_1	0	0	0						
G_2	0	0	0	0					
G_3		0	0	0	0				
A_4			0	0	0	0			
A_5				0	0	0	?		
A_6					0	0	1		
U_7						0	0	0	
C_8							0	0	0
C_9								0	0

- Zelle $(6,7)$ bekommt den Wert 1 und wir fahren analog fort
- Die nächste "interessante" Zelle ist $(5,7)$
- Es gibt zwei Möglichkeiten, den maximalen Score für $s[5,7]$ zu erreichen
  1. Neue Basenpaarung A_5 und U_7 und Score von $s[6,6]$ übernehmen (Option a)
  2. das A_5 ungepaart lassen und Score von Zelle $s[6,7]$ übernehmen (Option b)
- Da $s[i,j]$ nur den maximalen Score einer optimalen Struktur der Subsequence $s_{i\ldots j}$ speichert, ist es in diesem Schritt unwesentlich, von welcher Option der besten Alignment-Score kam

Füllen der Matrix (8)

1: **for** $s \leftarrow 2$ to n **do**
2: $\quad i \leftarrow 0$
3: $\quad$ **for** $j \leftarrow s$ to n **do**
4: $\quad\quad i \leftarrow i+1$
5: $\quad\quad s[i,j] \leftarrow \max(a,b,c,d)$
6: $\quad$ **end for**
7: **end for**

$$S(i,j) = \max \begin{cases} \text{a)} & \delta(i,j) + S(i+1,j-1) \\ \text{b)} & S(i+1,j) \\ \text{c)} & S(i,j-1) \\ \text{d)} & \max_{i<k<j} S(i,k) + S(k+1,j) \end{cases}$$

	G_1	G_2	G_3	A_4	A_5	A_6	U_7	C_8	C_9
G_1	0	0	0	0	0	0			
G_2	0	0	0	0	0	0	1		
G_3		0	0	0	0	0	1	?	
A_4			0	0	0	0	1	1	
A_5				0	0	0	1	1	1
A_6					0	0	1	1	1
U_7						0	0	0	0
C_8							0	0	0
C_9								0	0

- Hier sieht man die bis zur Zelle $(3,8)$ befüllte Matrix
- Option a: Paarung A_3 und U_8 zusammen mit Score für beste Substruktur der Subsequenz $s_{4\ldots7}$ (1+1).
- Option b: Base i (A_3) ungepaart zusammen mit Score für beste Substruktur der Subsequenz $s_{4\ldots8}$ (1).
- Option c: Base j (U_8) ungepaart zusammen mit Score für beste Substruktur der Subsequenz $s_{3\ldots7}$ (1).
- Option d: Summe der besten Scores der Subseuqenzen $s_{3\ldots4}$ und $s_{5\ldots8}$ (0+1) bzw. $s_{3\ldots5}$ und $s_{6\ldots8}$ (0+1) oder $s_{3\ldots6}$ und $s_{7\ldots8}$ (0+0)
- Die Wahl fällt auf Option a und wir setzen $s[3,8] = 2$.

Füllen der Matrix (9)

1: **for** $s \leftarrow 2$ to n **do**
2: $\quad i \leftarrow 0$
3: $\quad$ **for** $j \leftarrow s$ to n **do**
4: $\quad\quad i \leftarrow i+1$
5: $\quad\quad s[i,j] \leftarrow \max(a,b,c,d)$
6: $\quad$ **end for**
7: **end for**

$$S(i,j) = \max \begin{cases} \text{a)} & \delta(i,j) + S(i+1,j-1) \\ \text{b)} & S(i+1,j) \\ \text{c)} & S(i,j-1) \\ \text{d)} & \max_{i<k<j} S(i,k) + S(k+1,j) \end{cases}$$

	G_1	G_2	G_3	A_4	A_5	A_6	U_7	C_8	C_9
G_1	0	0	0	0	0	0	1	2	
G_2	0	0	0	0	0	0	1	2	?
G_3		0	0	0	0	0	1	2	2
A_4			0	0	0	0	1	1	1
A_5				0	0	0	1	1	1
A_6					0	0	1	1	1
U_7						0	0	0	0
C_8							0	0	0
C_9								0	0

- Wir fahren bis Zelle $(2,9)$ fort
- Der optimale Score wird durch eine Basenpaarung G_2 mit C_9 zusammen mit Score für beste Substruktur der Subsequenz $s_{3\ldots8}$ (1+2) erreicht.

Füllen der Matrix (10)

```
1: for s ← 2 to n do
2:     i ← 0
3:     for j ← s to n do
4:         i ← i + 1
5:         s[i, j] ← max(a, b, c, d)
6:     end for
7: end for
```

$$S(i,j) = \max \begin{cases} \text{a)} & \delta(i,j) + S(i+1,j-1) \\ \text{b)} & S(i+1,j) \\ \text{c)} & S(i,j-1) \\ \text{d)} & \max_{i<k<j} S(i,k) + S(k+1,j) \end{cases}$$

	G_1	G_2	G_3	A_4	A_5	A_6	U_7	C_8	C_9
G_1	0	0	0	0	0	0	1	2	?
G_2	0	0	0	0	0	0	1	2	3
G_3		0	0	0	0	0	1	2	2
A_4			0	0	0	0	1	1	1
A_5				0	0	0	1	1	1
A_6					0	0	1	1	1
U_7						0	0	0	0
C_8							0	0	0
C_9								0	0

- Die letzte Zell $(1,9)$ gibt den Score einer optimalen Struktur für die ganze Sequenz $s_{1\ldots9}$ an
- Der optimale Score kann hier durch Option a oder b erreicht werden (Überprüfen Sie!)

Traceback

- Der Traceback beginnt immer in der rechten oberen Ecke der Matrix und verfolgt den Pfad einer optimalen Struktur
- In diesem Fall ist der Traceback relativ einfach, weil keine verzweigte RNA-Strutkur vorliegt
- Im einfachen Fall kann man beginnend mit Zelle $(1, N)$ die Scores der drei umgebenden Zellen (links, schräg links unten und unten) vergleichen und so schlussfolgern, wie der Score in der aktuellen Zelle zustande gekommen sein muss und so eine optimale Struktur finden.
- Oft (wie hier) sind viele Pfade mit demselben Score möglich
- Siehe das Durbin-Buch für Einzelheiten zum allgemeinen Traceback-Algorithmus.

	G_1	G_2	G_3	A_4	A_5	A_6	U_7	C_8	C_9
G_1	0	0	0	0	0	0	1	2	(3)
G_2	0	0	0	0	0	0	1	2	(3)
G_3		0	0	0	0	0	1	(2)	2
A_4			0	0	0	0	(1)	1	1
A_5				0	0	(0)	1	1	1
A_6					0	(0)	1	1	1
U_7						0	0	0	0
C_8							0	0	0
C_9								0	0

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

Aufgabe

Bestimmen Sie nun eine optimale Sekundärstruktur für die RNA-Sequenz `AACGCUUGA` mit dem Nussinov-Algorithmus

zum Schluss

- Email: peter.robinson@charite.de

weiterführende Literatur

- Filipowicz W, Bhattacharyya SN, Sonenberg N. (2008) Mechanisms of post-transcriptional regulation by microRNAs: are the answers in sight?*Nat Rev Genet.* **9**:102-14.
- Eddy SR (2004) How do RNA folding algorithms work? *Nature Biotech*. **22**:1457-1458
- Durbin R, Eddy SR, Krogh A, Mitchison G (1998) Biological Sequence Analysis Probabilistic Models of Proteins and Nucleic Acids. Cambridge University Press (Kapitel 10)