

Algorithmische Bioinformatik

Hidden-Markov-Modelle
Viterbi - Algorithmus

Ulf Leser

Wissensmanagement in der
Bioinformatik



Ziel dieser Vorlesung

- Unterschied Markov Modelle - HMMs kennenlernen
- Etwas Intuition für HMM Modellierungsspielräume entwickeln
- Viterbi Algorithmus als weiteres DP-basiertes Optimierungsverfahren

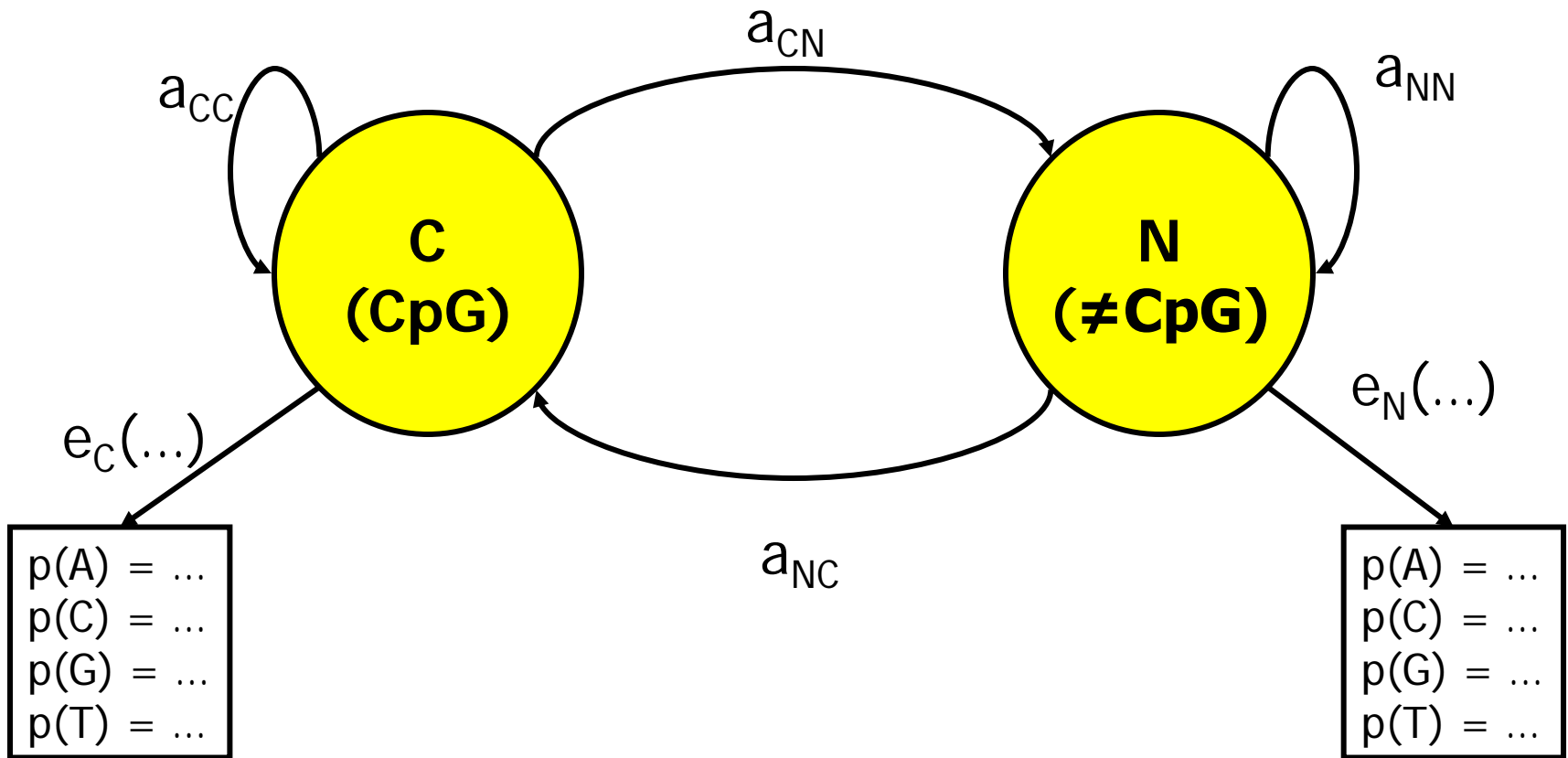
Inhalt der Vorlesung

- Hidden Markov Modelle
 - Baum, L. E. and Petrie, T. (1966). "Statistical Inference for Probabilistic Functions of Finite State Markov Chains." *Annals of Mathematical Statistics* 37: 1554-1563.
- Klassische Probleme
- Dekodierung: Viterbi-Algorithmus

Hidden Markov-Modelle (HMM)

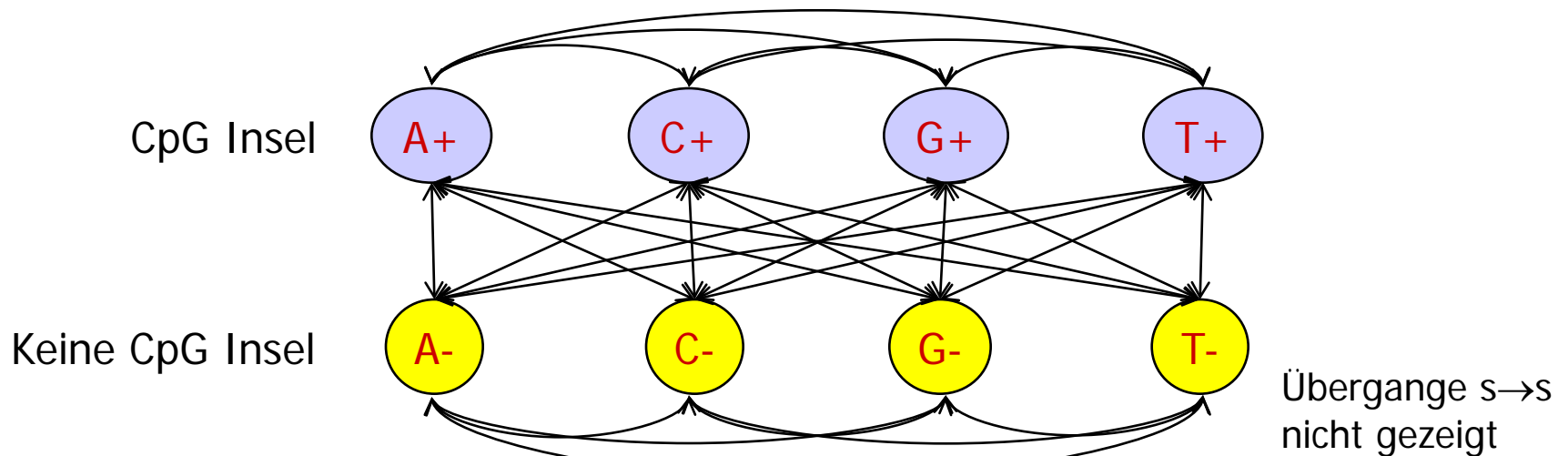
- Idee (am Beispiel CpG Islands)
 - Wir denken uns einen Automaten mit zwei „Meta-Zuständen“: Einer für CpG, einer für nicht-CpG
 - Jeder Zustand kann alle Zeichen des Alphabets mit einer bestimmten Emissions-Wsk emittieren
 - Damit: Zustandsfolge \neq Zeichenfolge
 - Das ist neu im Vergleich zu MM: Zustände sind verborgen (hidden)
 - Emissions-Wsk und Übergangs-Wsk in den beiden Metazuständen sind unterschiedlich und modellieren die Unterschiede zwischen normaler DNA und CpG-Inseln
 - Zustandsübergänge sollen automatisch erkannt werden
 - Keine festen Fenstergrößen mehr

Hidden Markov Modell 1: CpG



Feineres Modell

- Wir **erweitern die Anzahl der Zustände**: $A \rightarrow A+, A- \dots$
 - Zustände $s+$ und $s-$ emittieren jeweils **dasselbe Zeichen**
 - $M+$ Modell: Übergangswsk zwischen $s+$ Zuständen
 - $M-$ Modell: Übergangswsk zwischen $s-$ Zuständen
- Übergang von jedem $s+$ Zustand zu jedem $s-$ Zustand und umgekehrt mit bestimmter (kleiner) Wsk erlaubt



Formale Definition von HMMs

- Definition

Gegeben Σ . Ein Hidden Markov Modell ist ein sequentieller stochastischer Prozess über k Zuständen s_1, \dots, s_k mit

- *Zustand s emittiert Zeichen $x \in \Sigma$ mit Wahrscheinlichkeit $p(x/s)$*
- *Folge der Zustände ist eine Markov-Kette (erster Ordnung), d.h.:*

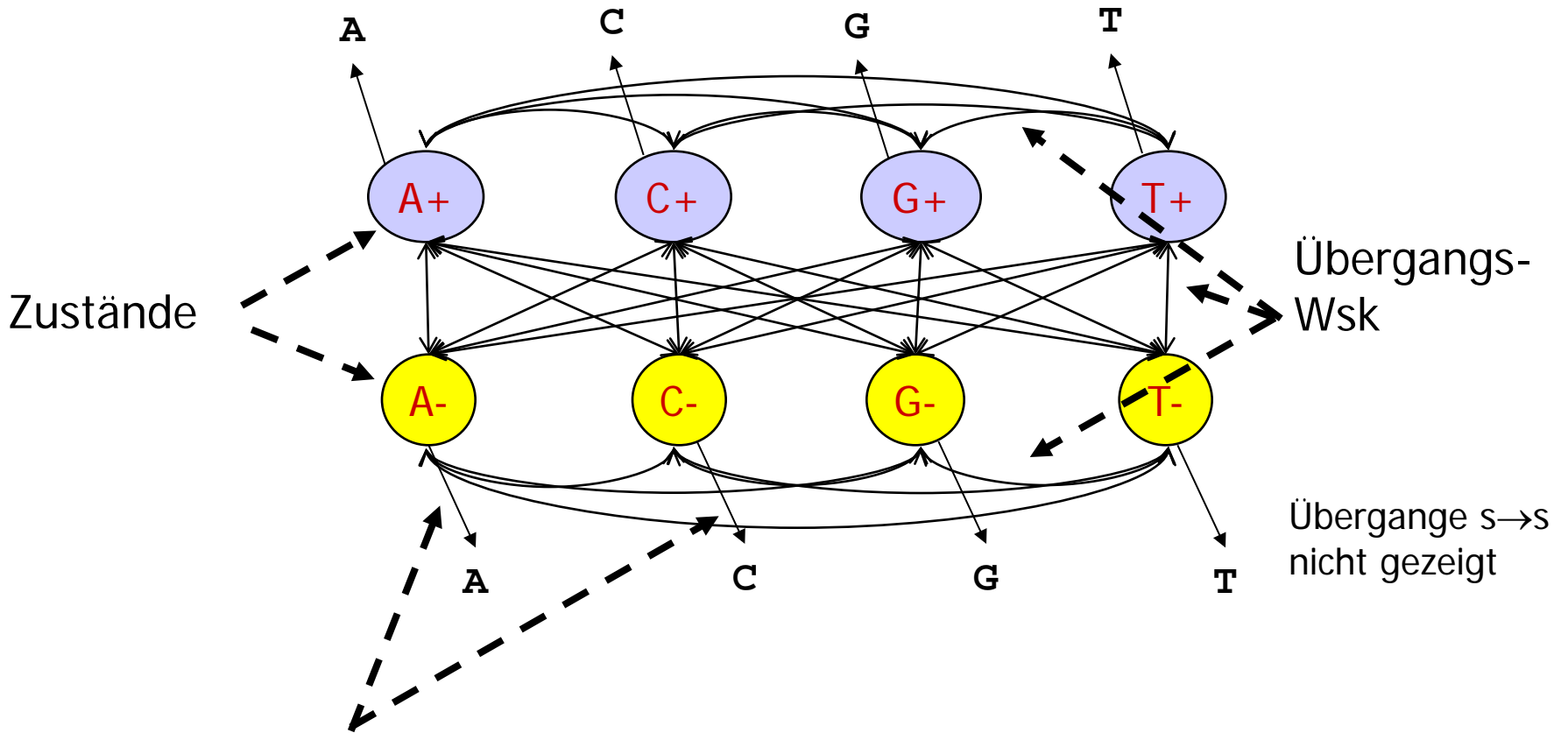
$$p(z_t=s_t/z_{t-1}=s_{t-1}, z_{t-2}=s_{t-2}, \dots, z_0=s_0) = p(z_t=s_t/z_{t-1}=s_{t-1}) = a_{t-1,t}$$

- *Die $a_{0,1}$ heißen Startwahrscheinlichkeiten*
- *Die $a_{t-1,t}$ heißen Übergangswahrscheinlichkeiten*
- *Die $e_s(x) = p(x/s)$ heißen Emissionswahrscheinlichkeiten*

- Bemerkung

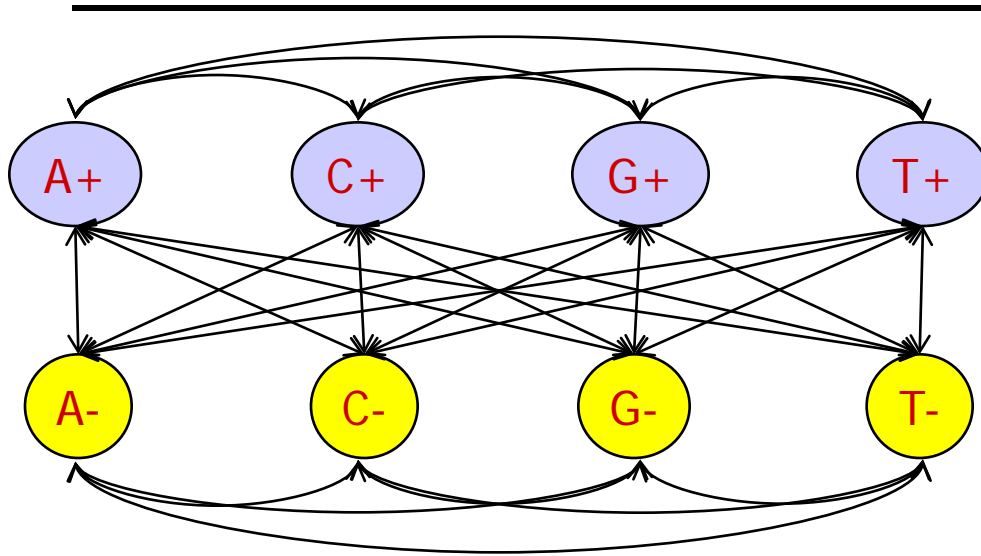
- Die existierenden Zustände in M sind s_1, \dots, s_k
- Eine konkrete Zustandsfolge ist z_0, z_1, \dots, z_n

Beispiel



Emissions-Wsk (hier: Pro Zustand ist eine Wsk=1 und alle anderen=0)

Sequenzen und Zustandsfolgen



M+	A	C	G	T
A	.180	.274	.426	.120
C	.171	.368	.274	.188
G	.161	.339	.375	.125
T	.079	.355	.384	.182
M-	A	C	G	T
A	.300	.205	.285	.210
C	.233	.298	.078	.302
G	.248	.246	.298	.208
T	.177	.239	.292	.292

A C T G A C
A+C+T-G-A-C-

A C T G A C
A+C+T+G+A+C+

A C T G A C
A-C+T+G+A-C+

- Alle möglich
- Aber **nicht** alle **gleich wahrscheinlich**
 - Start-Wsk alle gleich, $p(M+ \leftrightarrow M-)$: 0.01
 - **A+C+T-G-A-C-**: 0.0000406...
 - **A+C+T+G+A+C+**: 0.0008726...
 - **A-C+T+G+A-C+**: 0.00000000...

Inhalt der Vorlesung

- Hidden Markov Modelle
- **Klassische Probleme**
- Dekodierung: Viterbi-Algorithmus

Die drei klassischen HMM Probleme

- **Dekodierung/Parse**: Gegeben eine Sequenz S und ein HMM M ; durch welche Zustandsfolge wurde S wahrscheinlich erzeugt?
 - Lösung: Viterbi Algorithmus
- **Evaluation**: Gegeben eine Sequenz S und ein HMM M ; mit welcher Wahrscheinlichkeit wurde S durch M erzeugt?
 - Muss alle Zustandssequenzen berücksichtigen, die S erzeugen
 - Lösung: Forward/Backward Algorithmus
- **Lernen/Trainieren**: Gegeben eine Sequenz S ; welches HMM M (mit gegebener Zustandsmenge) erzeugt S mit der größten Wsk?
 - Lernen der Übergangs- und Emissionswahrscheinlichkeiten aus S
 - Lösung: Baum-Welch Algorithmus

Example: The Dishonest Casino

A casino has two dice:

- Fair die

$$P(1)=P(2)=P(3)=P(4)=P(5)=P(6) = 1/6$$

- Loaded die

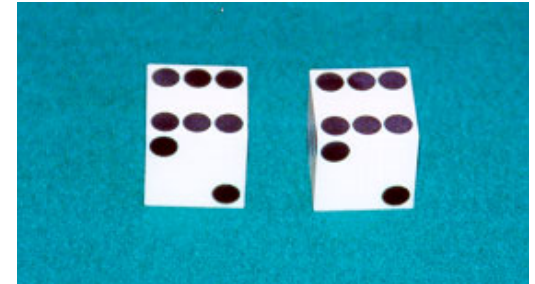
$$P(1)=P(2)=P(3)=P(4)=P(5) = 1/10$$

$$P(6) = 1/2$$

Casino occasionally switches between dice
(and you want to know when)

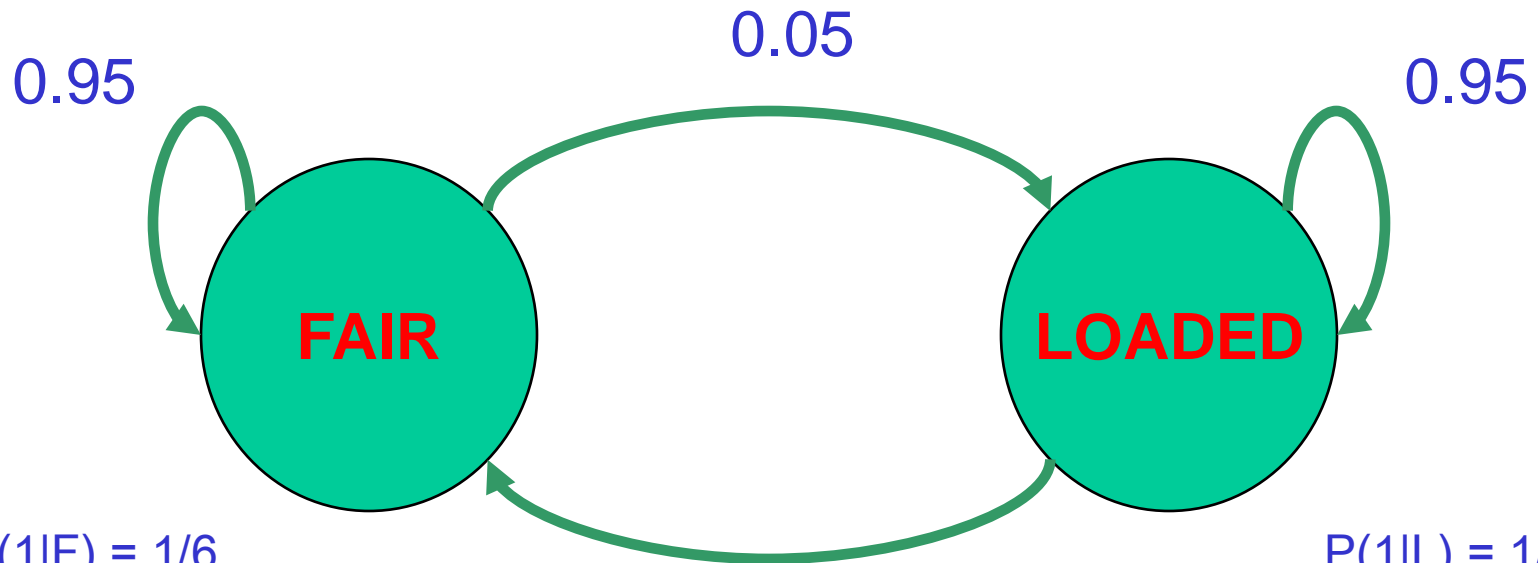
Game:

1. You bet \$1
2. You roll (always with a fair die)
3. You may bet more or surrender
4. Casino player rolls (with some die...)
5. Highest number wins



Quelle: Batzoglou, Stanford

Model



$$\begin{aligned} P(1|F) &= 1/6 \\ P(2|F) &= 1/6 \\ P(3|F) &= 1/6 \\ P(4|F) &= 1/6 \\ P(5|F) &= 1/6 \\ P(6|F) &= 1/6 \end{aligned}$$

$$\begin{aligned} P(1|L) &= 1/10 \\ P(2|L) &= 1/10 \\ P(3|L) &= 1/10 \\ P(4|L) &= 1/10 \\ P(5|L) &= 1/10 \\ P(6|L) &= 1/2 \end{aligned}$$

Question # 1 – Decoding

GIVEN A sequence of rolls by the casino player

62146146136136661664661636616366163616515615115146123562344

What portion of the sequence was generated with the fair die, and what portion with the loaded die?

62146146136136661664661636616366163616515615115146123562344

This is the **DECODING** question in HMMs

Question # 2 – Evaluation

GIVEN A sequence of rolls by the casino player

62146146136136661664661636616366163616515615115146123562344

How likely is this sequence, given our model of how the casino works?

62146146136136661664661636616366163616515615115146123562344
62146146136136661664661636616366163616515615115146123562344
62146146136136661664661636616366163616515615115146123562344

...

This is the **EVALUATION** problem in HMMs

Question # 3 – Learning

GIVEN A sequence of rolls by the casino player

6146136136661664661636616366163616515615115146123562344

How “loaded” is the loaded die? How “fair” is the fair die? How often does the casino player change from fair to loaded, and back?

This is the **LEARNING** question in HMMs

[Note: We need to know how many dice there are!]

Inhalt der Vorlesung

- Hidden Markov Modelle
- Klassische Probleme
- Dekodierung: Viterbi-Algorithmus

Dekodierproblem

- Definition

*Gegeben ein HMM M und eine Sequenz S . Das **Dekodierproblem** sucht nach der Zustandsfolge p , die mit der höchsten Wsk (unter allen Zustandsfolgen von M) S erzeugt hat*

$$p^*(S | M) = \max_{p \in \text{paths}} p(S, p)$$

- Bemerkung

- Eine konkrete Zustandsfolge bezeichnen wir auch als **Pfad**

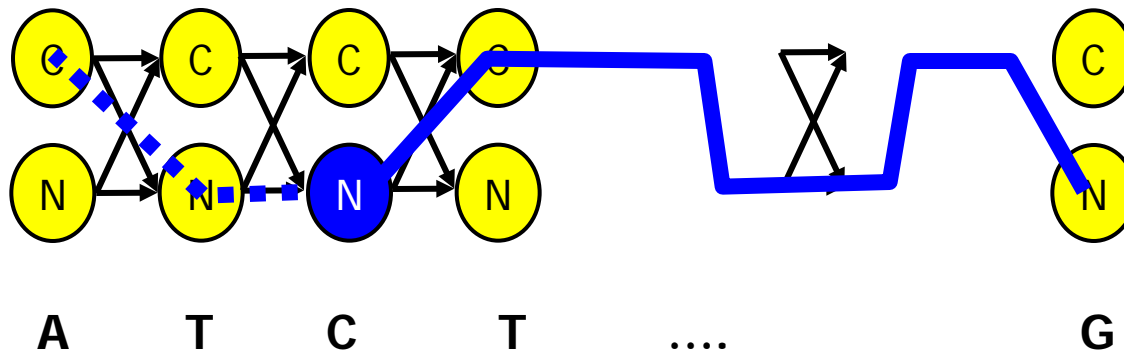
Naive Lösung

- Nehmen wir an, dass $a_{ij} > 0$ und $e_i(x) > 0$ für alle x, i, j
- Dann gibt es **wie viele Pfade?**
 - Es gibt k^n Pfade
- Alle aufzählen und vergleichen ist eine schlechte Idee
- Besser: **Viterbi-Algorithmus**
 - Viterbi, A. J. (1967). "Error bounds for convolution codes and an asymptotically optimal decoding algorithm." *IEEE Transactions on Information Theory* **IT-13**: 260-269.
 - Prinzip: **Dynamische Programmierung** über Pfadpräfixe

Viterbi Algorithmus: Grundidee

- Beobachtung

- Es gibt viele Pfade, um einen **Zustand s an Position i** zu erreichen
- **Einer davon** muss der wahrscheinlichste sein
 - Eigentlich: Einer der wahrscheinlichsten
- Alle Fortsetzungen von Pfaden ab „s an i“ brauchen nur diese Wsk
- Also: Sukzessive höchste Wahrscheinlichkeiten für alle Zustände s an allen Positionen i berechnen



Sequenz:

A

T

C

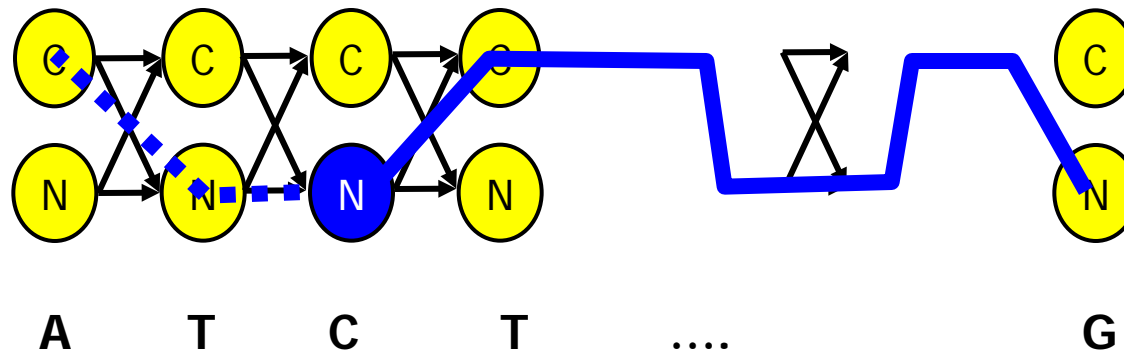
T

....

G

Viterbi: Dynamische Programmierung

- Wir berechnen **optimale Pfade** für länger werdende Präfixe **von S** für alle möglichen Endzustände
- Sei $v_s(i)$ die Wahrscheinlichkeit des optimalen Pfads für $S[..i]$, der in Zustand s endet
- Wir brauchen eine **Rekursionsformel** für $v_s(i+1)$
 - ... und Randbedingungen



Sequenz:

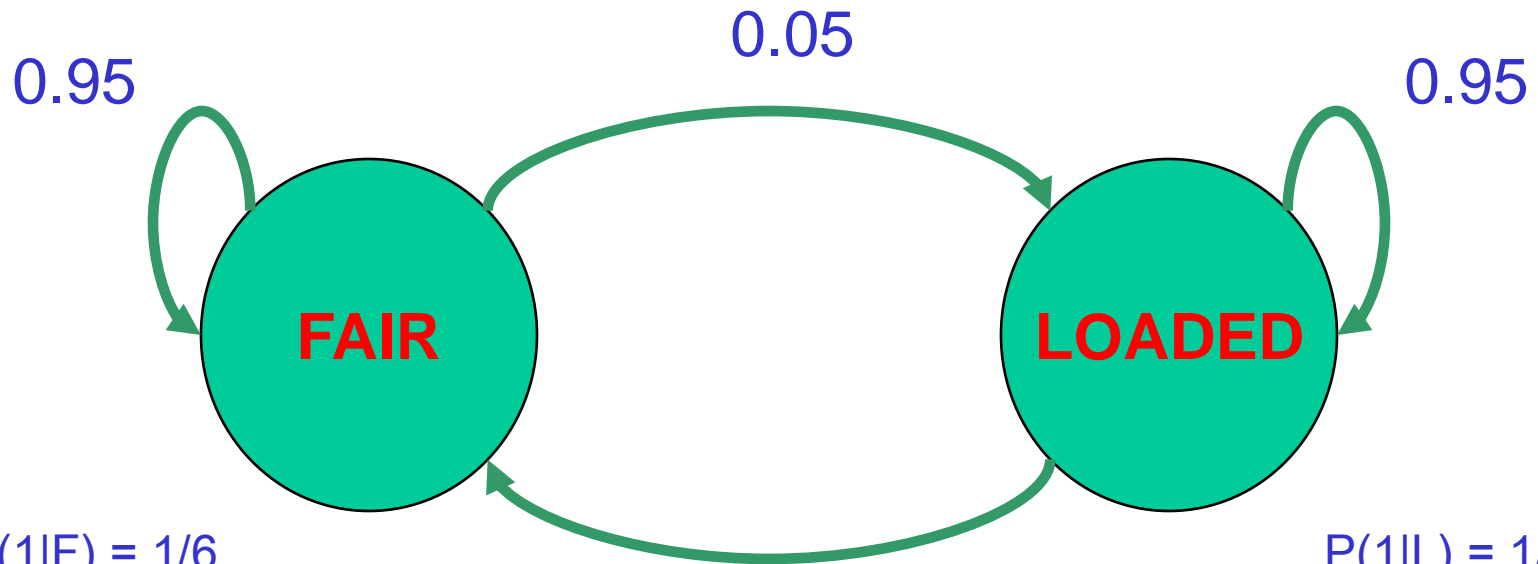
A T C T G

Rekursion

- Sei $v_s(i)$ die Wahrscheinlichkeit des optimalen Pfad für $S[..i]$, der in Zustand s endet
- Wenn wir zu Zustand t übergehen
 - Wir können von jedem Vorgängerzustand (s) kommen
 - Von Zustand s gehen wir mit Wsk $a_{st}=v(t|s)$ nach t
 - Dann emittieren wir in jedem Fall das Zeichen $S[i+1]$ mit Emissionswahrscheinlichkeit $e_t(S[i+1])$
- Zusammen

$$v_t(i+1) = e_t(S[i+1]) * \max_{s \in M} (v_s(i) * a_{st})$$

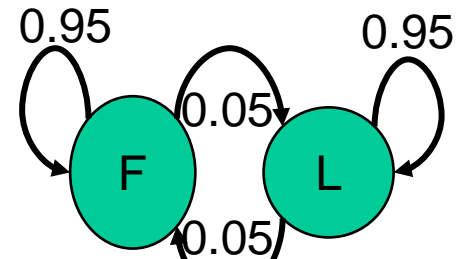
Model



$$\begin{aligned} P(1|F) &= 1/6 \\ P(2|F) &= 1/6 \\ P(3|F) &= 1/6 \\ P(4|F) &= 1/6 \\ P(5|F) &= 1/6 \\ P(6|F) &= 1/6 \end{aligned}$$

$$\begin{aligned} P(1|L) &= 1/10 \\ P(2|L) &= 1/10 \\ P(3|L) &= 1/10 \\ P(4|L) &= 1/10 \\ P(5|L) &= 1/10 \\ P(6|L) &= 1/2 \end{aligned}$$

Tabellarische Darstellung



		6	3	4	2	6	1	6	6
S_0	1								
Fair	0	0,83	1,32	2,09	3,31	5,24	8,29	13,13	20,79
Load	0	2,50	2,38	2,26	2,14	10,18	9,67	45,94	218,23

Bereich: In jedem Schritt *10

- **Start:** s_0 hat Wsk 1, alle anderen Zustände haben Wsk 0
- Wir berechnen die Tabelle spaltenweise
- Jedes Feld kann von **jedem Feld der Vorgängerspalte** erreicht werden
 - Also aus jedem Endzustand für das um 1 Zeichen kürzere Präfix
- In jeder Spalte i ist die Wsk aller Zustände, die nicht $S[i]$ emittieren, 0 (weil $e(S[i])=0$)

Varianten

Loaded zur 1 statt 6

		6	3	4	2	6	1	6	6
S ₀	1								
Fair	0	0,83	1,32	2,09	3,31	5,24	8,29	13,13	20,79
Load	0	0,50	0,48	0,45	0,43	0,41	1,93	1,84	1,75
Ratio		1,67	2,78	4,63	7,72	12,86	4,29	7,14	11,91

Übergangswahrscheinlichkeiten alle 0,5

		6	3	4	2	6	1	6	6
S ₀	1								
Fair	0	0,83	2,08	1,74	1,45	1,21	3,01	2,51	6,28
Load	0	2,50	1,25	1,04	0,87	3,62	1,81	7,54	18,84
Ratio		0,33	1,67	1,67	1,67	0,33	1,67	0,33	0,33

Zustandssequenz

		6	3	4	2	6	1	6	6
S ₀	1								
Fair	0	0,83	1,32	2,09	3,31	5,24	8,29	13,13	20,79
Load	0	2,50	2,38	2,26	2,14	10,18	9,67	45,94	218,23

- Gesuchte Wsk ist die **höchste Zahl am rechten Rand**
- Optimaler **Pfad über Traceback**
 - Ergibt hier „LLLLLLLL“ – an Position 4 endet die Sequenz nicht, und aus dem dort optimalen „F“ müsste man später wieder raus kommen – zu teuer

Komplexität

- Sei $|S|=n$, HMM habe $|M|=k$ Zustände
- Allgemein
 - Tabelle hat $n*k$ Zellen
 - Für jede Zelle greifen wir auf k Vorgängerzellen zu
 - Zusammen: $O(n*k^2)$
- Unser konkretes CpG-Insel Modell
 - In unserem aktuellen Modell sind in jedem Zustand **alle Emissionswsk bis auf eine 0**
 - Damit müssen wir pro Spalte nur zwei Zellen betrachten, und die haben nur 2 Vorläufer mit Wsk $\neq 0$
 - Damit ist **der Algorithmus: $\sim O(4*n)=O(n)$**

Numerische Schwierigkeiten

- Multiplikation vieler Zahlen $\ll 1$ erreicht schnell die **Rechengenauigkeitsgrenze**
- Besser: Logarithmieren
 - Statt:

$$v_t(i+1) = e_t(S[i+1]) * \max_{s \in M} (v_s(i) * a_{st})$$

- Berechnet man:

$$v_t(i+1) = \log(e_t(S[i+1])) + \max_{s \in M} (v_s(i) + \log(a_{st}))$$

Selbsttest

- Was ist ein HMM?
- Was muss für die Wsk aller ausgehenden Emissionskanten gelten?
- Im „Dishonest Casino“ – wie könnte man die Übergangswsk zwischen den beiden Zuständen nur aus Beobachtungen schätzen?
- Wie funktioniert der Viterbi-Algorithmus?
- Warum ist der Viterbi ein Beispiel für Dynamische Programmierung?