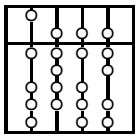
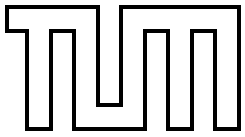


INSTITUT FÜR INFORMATIK
DER TECHNISCHEN UNIVERSITÄT MÜNCHEN
LEHRSTUHL FÜR EFFIZIENTE ALGORITHMEN



Skriptum

zur Vorlesung

Algorithmische Bioinformatik I/II

gehalten im Wintersemester 2001/2002

und im Sommersemester 2002 von

Volker Heun

Erstellt unter Mithilfe von:

Peter Lücke – Hamed Behrouzi – Michael Engelhardt

Sabine Spreer – Hanjo Täubig

Jens Ernst – Moritz Maaß

14. Mai 2003

Version 0.96

Vorwort

Dieses Skript entstand parallel zu den Vorlesungen *Algorithmische Bioinformatik I* und *Algorithmische Bioinformatik II*, die im Wintersemester 2001/2002 sowie im Sommersemester 2002 für Studenten der Bioinformatik und Informatik sowie anderer Fachrichtungen an der Technischen Universität München im Rahmen des von der Ludwig-Maximilians-Universität und der Technischen Universität gemeinsam veranstalteten Studiengangs Bioinformatik gehalten wurde. Einige Teile des Skripts basieren auf der bereits im Sommersemester 2000 an der Technischen Universität München gehaltenen Vorlesung *Algorithmen der Bioinformatik* für Studierende der Informatik.

Das Skript selbst umfasst im Wesentlichen die grundlegenden Themen, die man im Bereich Algorithmische Bioinformatik einmal gehört haben sollte. Die vorliegende Version bedarf allerdings noch einer Ergänzung weiterer wichtiger Themen, die leider nicht in den Vorlesungen behandelt werden konnten.

An dieser Stelle möchte ich insbesondere Hamed Behrouzi, Michael Engelhardt und Peter Lücke danken, die an der Erstellung des ersten Teils dieses Skriptes (Kapitel 2 mit 5) maßgeblich beteiligt waren. Bei Sabine Spreer möchte ich mich für die Unterstützung bei Teilen des siebten Kapitels bedanken. Bei meinen Übungsleitern Jens Ernst und Moritz Maaß für deren Unterstützung der Durchführung des Übungsbetriebs, aus der einige Lösungen von Übungsaufgaben in dieses Text eingeflossen sind. Bei Hanjo Täubig möchte ich mich für die Mithilfe zur Fehlerfindung bedanken, insbesondere bei den biologischen Grundlagen.

Falls sich dennoch weitere (Tipp)Fehler unserer Aufmerksamkeit entzogen haben sollten, so bin ich für jeden Hinweis darauf (an heun@in.tum.de) dankbar.

München, im September 2002 Volker Heun

Inhaltsverzeichnis

1 Molekularbiologische Grundlagen 1

1.1 Mendelsche Genetik 1

1.1.1 Mendelsche Experimente 1

1.1.2 Modellbildung 2

1.1.3 Mendelsche Gesetze 4

1.1.4 Wo und wie sind die Erbinformationen gespeichert? 4

1.2 Chemische Grundlagen 4

1.2.1 Kovalente Bindungen 5

1.2.2 Ionische Bindungen 7

1.2.3 Wasserstoffbrücken 8

1.2.4 Van der Waals-Kräfte 9

1.2.5 Hydrophobe Kräfte 10

1.2.6 Funktionelle Gruppen 10

1.2.7 Stereochemie und Enantiomerie 11

1.2.8 Tautomerien 13

1.3 DNS und RNS 14

1.3.1 Zucker 14

1.3.2 Basen 16

1.3.3 Polymerisation 18

1.3.4 Komplementarität der Basen 18

1.3.5 Doppelhelix 20

1.4 Proteine 22

1.4.1 Aminosäuren 22

1.4.2 Peptidbindungen 23
1.4.3 Proteinstrukturen 26
1.5 Der genetische Informationsfluss 29
1.5.1 Replikation 29
1.5.2 Transkription 30
1.5.3 Translation 31
1.5.4 Das zentrale Dogma 34
1.5.5 Promotoren 34
1.6 Biotechnologie 35
1.6.1 Hybridisierung 35
1.6.2 Klonierung 35
1.6.3 Polymerasekettenreaktion 36
1.6.4 Restriktionsenzyme 37
1.6.5 Sequenzierung kurzer DNS-Stücke 38
1.6.6 Sequenzierung eines Genoms 40

2 Suchen in Texten 43

2.1 Grundlagen 43
2.2 Der Algorithmus von Knuth, Morris und Pratt 43
2.2.1 Ein naiver Ansatz 44
2.2.2 Laufzeitanalyse des naiven Algorithmus: 45
2.2.3 Eine bessere Idee 45
2.2.4 Der Knuth-Morris-Pratt-Algorithmus 47
2.2.5 Laufzeitanalyse des KMP-Algorithmus: 48
2.2.6 Berechnung der Border-Tabelle 48
2.2.7 Laufzeitanalyse: 51
2.3 Der Algorithmus von Aho und Corasick 51

2.3.1 Naiver Lösungsansatz 52
2.3.2 Der Algorithmus von Aho und Corasick 52
2.3.3 Korrektheit von Aho-Corasick 55
2.4 Der Algorithmus von Boyer und Moore 59
2.4.1 Ein zweiter naiver Ansatz 59
2.4.2 Der Algorithmus von Boyer-Moore 60
2.4.3 Bestimmung der Shift-Tabelle 63
2.4.4 Laufzeitanalyse des Boyer-Moore Algorithmus: 64
2.4.5 Bad-Character-Rule 71
2.5 Der Algorithmus von Karp und Rabin 72
2.5.1 Ein numerischer Ansatz 72
2.5.2 Der Algorithmus von Karp und Rabin 75
2.5.3 Bestimmung der optimalen Primzahl 75
2.6 Suffix-Tries und Suffix-Bäume 79
2.6.1 Suffix-Tries 79
2.6.2 Ukkonens Online-Algorithmus für Suffix-Tries 81
2.6.3 Laufzeitanalyse für die Konstruktion von T^n 83
2.6.4 Wie groß kann ein Suffix-Trie werden? 83
2.6.5 Suffix-Bäume 85
2.6.6 Ukkonens Online-Algorithmus für Suffix-Bäume 86
2.6.7 Laufzeitanalyse 96
2.6.8 Problem: Verwaltung der Kinder eines Knotens 97

3 Paarweises Sequenzen Alignment 101

3.1 Distanz- und Ähnlichkeitsmaße . . . 101

3.1.1 Edit-Distanz . . . 102

3.1.2 Alignment-Distanz . . . 106

3.1.3 Beziehung zwischen Edit- und Alignment-Distanz . . . 107

3.1.4 Ähnlichkeitsmaße . . . 110

3.1.5 Beziehung zwischen Distanz- und Ähnlichkeitsmaßen . . . 111

3.2 Bestimmung optimaler globaler Alignments . . . 115

3.2.1 Der Algorithmus nach Needleman-Wunsch . . . 115

3.2.2 Sequenzen Alignment mit linearem Platz (Modifikation von Hirschberg) . . . 121

3.3 Besondere Berücksichtigung von Lücken . . . 130

3.3.1 Semi-Globale Alignments . . . 130

3.3.2 Lokale Alignments (Smith-Waterman) . . . 133

3.3.3 Lücken-Strafen . . . 136

3.3.4 Allgemeine Lücken-Strafen (Waterman-Smith-Byers) . . . 137

3.3.5 Affine Lücken-Strafen (Gotoh) . . . 139

3.3.6 Konkave Lücken-Strafen . . . 142

3.4 Hybride Verfahren . . . 142

3.4.1 One-Against-All-Problem . . . 143

3.4.2 All-Against-All-Problem . . . 145

3.5 Datenbanksuche . . . 147

3.5.1 FASTA (FAST All oder FAST Alignments) . . . 147

3.5.2 BLAST (Basic Local Alignment Search Tool) . . . 150

3.6 Konstruktion von Ähnlichkeitsmaßen . . . 150

3.6.1 Maximum-Likelihood-Prinzip . . . 150

3.6.2 PAM-Matrizen . . . 152

4 Mehrfaches Sequenzen Alignment 155

4.1 Distanz- und Ähnlichkeitsmaße 155

4.1.1 Mehrfache Alignments 155

4.1.2 Alignment-Distanz und -Ähnlichkeit 155

4.2 Dynamische Programmierung 157

4.2.1 Rekursionsgleichungen 157

4.2.2 Zeitanalyse 158

4.3 Alignment mit Hilfe eines Baumes 159

4.3.1 Mit Bäumen konsistente Alignments 159

4.3.2 Effiziente Konstruktion 160

4.4 Center-Star-Approximation 161

4.4.1 Die Wahl des Baumes 161

4.4.2 Approximationsgüte 162

4.4.3 Laufzeit für Center-Star-Methode 164

4.4.4 Randomisierte Varianten 164

4.5 Konsensus eines mehrfachen Alignments 167

4.5.1 Konsensus-Fehler und Steiner-Strings 168

4.5.2 Alignment-Fehler und Konsensus-String 171

4.5.3 Beziehung zwischen Steiner-String und Konsensus-String . . . 172

4.6 Phylogenetische Alignments 174

4.6.1 Definition phylogenetischer Alignments 175

4.6.2 Geliftete Alignments 176

4.6.3 Konstruktion eines gelifteten aus einem optimalem Alignment 177

4.6.4 Güte gelifteter Alignments 177

4.6.5 Berechnung eines optimalen gelifteten PMSA 180

5 Fragment Assembly **183**

5.1 Sequenzierung ganzer Genome 183

5.1.1 Shotgun-Sequencing 183

5.1.2 Sequence Assembly 184

5.2 Overlap-Detection und Fragment-Layout 185

5.2.1 Overlap-Detection mit Fehlern 185

5.2.2 Overlap-Detection ohne Fehler 185

5.2.3 Greedy-Ansatz für das Fragment-Layout 188

5.3 Shortest Superstring Problem 189

5.3.1 Ein Approximationsalgorithmus 190

5.3.2 Hamiltonsche Kreise und Zyklenüberdeckungen 194

5.3.3 Berechnung einer optimalen Zyklenüberdeckung 197

5.3.4 Berechnung gewichtsmaximaler Matchings 200

5.3.5 Greedy-Algorithmus liefert eine 4-Approximation 204

5.3.6 Zusammenfassung und Beispiel 210

5.4 (*) Whole Genome Shotgun-Sequencing 213

5.4.1 Sequencing by Hybridization 213

5.4.2 Anwendung auf Fragment Assembly 215

6 Physical Mapping **219**

6.1 Biologischer Hintergrund und Modellierung 219

6.1.1 Genomische Karten 219

6.1.2 Konstruktion genomischer Karten 220

6.1.3 Modellierung mit Permutationen und Matrizen 221

6.1.4 Fehlerquellen 222

6.2 PQ-Bäume 223

6.2.1 Definition von PQ-Bäumen 223

6.2.2 Konstruktion von PQ-Bäumen . . . 226
6.2.3 Korrektheit . . . 234
6.2.4 Implementierung . . . 236
6.2.5 Laufzeitanalyse . . . 241
6.2.6 Anzahlbestimmung angewendeter Schablonen . . . 244
6.3 Intervall-Graphen . . . 246
6.3.1 Definition von Intervall-Graphen . . . 247
6.3.2 Modellierung . . . 248
6.3.3 Komplexitäten . . . 250
6.4 Intervall Sandwich Problem . . . 251
6.4.1 Allgemeines Lösungsprinzip . . . 251
6.4.2 Lösungsansatz für Bounded Degree Interval Sandwich . . . 255
6.4.3 Laufzeitabschätzung . . . 262

7 Phylogenetische Bäume 265

7.1 Einleitung . . . 265
7.1.1 Distanzbasierte Verfahren . . . 266
7.1.2 Charakterbasierte Methoden . . . 267
7.2 Ultrametriken und ultrametrische Bäume . . . 268
7.2.1 Metriken und Ultrametriken . . . 268
7.2.2 Ultrametrische Bäume . . . 271
7.2.3 Charakterisierung ultrametrischer Bäume . . . 274
7.2.4 Konstruktion ultrametrischer Bäume . . . 278
7.3 Additive Distanzen und Bäume . . . 281
7.3.1 Additive Bäume . . . 281
7.3.2 Charakterisierung additiver Bäume . . . 283
7.3.3 Algorithmus zur Erkennung additiver Matrizen . . . 290

7.3.4 4-Punkte-Bedingung 291
7.3.5 Charakterisierung kompakter additiver Bäume 294
7.3.6 Konstruktion kompakter additiver Bäume 297
7.4 Perfekte binäre Phylogenie 298
7.4.1 Charakterisierung perfekter Phylogenie 299
7.4.2 Binäre Phylogenien und Ultrametriken 303
7.5 Sandwich Probleme 305
7.5.1 Fehlertolerante Modellierungen 306
7.5.2 Eine einfache Lösung 307
7.5.3 Charakterisierung einer effizienteren Lösung 314
7.5.4 Algorithmus für das ultrametrische Sandwich-Problem 322
7.5.5 Approximationsprobleme 335

8 Hidden Markov Modelle 337
8.1 Markov-Ketten 337
8.1.1 Definition von Markov-Ketten 337
8.1.2 Wahrscheinlichkeiten von Pfaden 339
8.1.3 Beispiel: CpG-Inseln 340
8.2 Hidden Markov Modelle 342
8.2.1 Definition 342
8.2.2 Modellierung von CpG-Inseln 343
8.2.3 Modellierung eines gezinkten Würfels 344
8.3 Viterbi-Algorithmus 345
8.3.1 Decodierungsproblem 345
8.3.2 Dynamische Programmierung 345
8.3.3 Implementierungstechnische Details 346
8.4 Posteriori-Decodierung 347

8.4.1 Ansatz zur Lösung 348
8.4.2 Vorwärts-Algorithmus 348
8.4.3 Rückwärts-Algorithmus 349
8.4.4 Implementierungstechnische Details 350
8.4.5 Anwendung 351
8.5 Schätzen von HMM-Parametern 353
8.5.1 Zustandsfolge bekannt 353
8.5.2 Zustandsfolge unbekannt — Baum-Welch-Algorithmus 354
8.5.3 Erwartungswert-Maximierungs-Methode 356
8.6 Mehrfaches Sequenzen Alignment mit HMM 360
8.6.1 Profile 360
8.6.2 Erweiterung um InDel-Operationen 361
8.6.3 Alignment gegen ein Profil-HMM 363

A Literaturhinweise **367**
A.1 Lehrbücher zur Vorlesung 367
A.2 Skripten anderer Universitäten 367
A.3 Lehrbücher zu angrenzenden Themen 368
A.4 Originalarbeiten 368

B Index **371**

Hidden Markov Modelle

8.1 Markov-Ketten

Um im Folgenden einige Problemstellungen mit Hilfe von so genannten Hidden Markov Modellen modellieren zu können, müssen wir uns zuerst mit den so genannten gewöhnlichen Markov-Ketten vertraut machen.

8.1.1 Definition von Markov-Ketten

Bevor wir zur Definition von Markov-Ketten kommen, wiederholen wir noch einmal die Definition einer stochastischen Matrix.

Definition 8.1 *Ein $n \times m$-Matrix M heißt* stochastisch*, wenn $M_{i,j} \in [0,1]$ für alle $i \in [1:n]$ und $j \in [1:m]$ sowie $\sum_{j=1}^{m} M_{i,j} = 1$ für alle $i \in [1:n]$. Ein Vektor $x = (x_1, \ldots, x_n)$ (Zeilen- oder Spaltenvektor) heißt* stochastisch*, wenn $x_i \in [0,1]$ und $\sum_{i=1}^{n} x_i = 1$ gilt.*

Eine Markov-Kette ist nichts anderes als eine Menge von Zuständen zwischen denen man in jedem (diskreten) Zeitschritt mit einer gewissen Wahrscheinlichkeit, die vom momentanen Zustand abhängt, in einen anderen Zustand wechselt.

Definition 8.2 *Eine* Markov-Kette *ist ein Tripel $M = (Q, P, \pi)$, wobei*

- *$Q = \{q_1, \ldots, q_n\}$ eine endliche Menge von Zuständen ist;*
- *P eine stochastische $n \times n$-Matrix der Zustandsübergangswahrscheinlichkeiten ist;*
- *$\pi = (\pi_1, \ldots, \pi_n)$ ist ein stochastischer Vektor der Anfangswahrscheinlichkeiten.*

Die Anfangswahrscheinlichkeiten gibt hierbei an, mit welcher Wahrscheinlichkeit π_i wir uns zu Beginn im Zustand $q_i \in Q$ befinden. Die Zustandsübergangswahrscheinlichkeiten $p_{i,j}$ geben an, mit welcher Wahrscheinlichkeit wir vom Zustand q_i in den Zustand q_j wechseln. Damit muss die Summe der Wahrscheinlichkeiten vom Zustand q_i aus 1 sein und daher fordern wir, dass P eine stochastische Matrix sein muss.

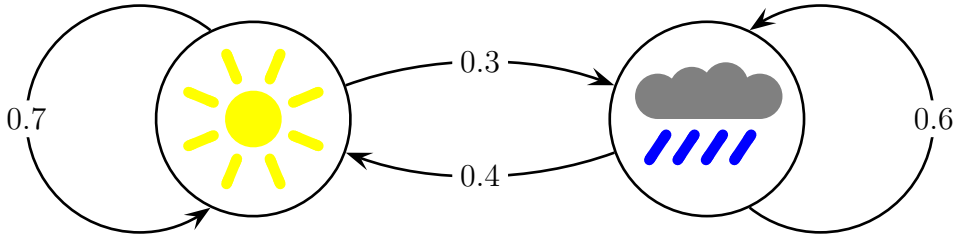


Abbildung 8.1: Beispiel: Einfache Markov-Kette für das Wetter

Betrachten wir dazu ein einfaches Beispiel, nämlich eine simple Modellierung des Wetters, wie sie in Abbildung 8.1 illustriert ist. Wir besitzen zwei Zustände, nämlich schönes Wetter und schlechtes Wetter. Die Übergangswahrscheinlichkeiten geben dabei an, mit welcher Wahrscheinlichkeit man beispielsweise vom schönen Wetter aus wechseln kann. Mit Wahrscheinlichkeit 0.7 bleibt es schön und mit Wahrscheinlichkeit 0.3 wird es schlecht, sofern momentan schönes Wetter herrscht. Umgekehrt wird es mit Wahrscheinlichkeit 0.4 schön und mit Wahrscheinlichkeit 0.6 bleibt es schlecht, sofern das Wetter gerade schlecht ist.

Mit $(x_1, \ldots, x_\ell) \in Q^*$ bezeichnen wir die Folge von Zuständen, die eine Markovkette zu den (diskreten) Zeitpunkten t_i mit $i \in [1:\ell]$ und $t_1 < \cdots < t_\ell$ durchläuft. Dabei interpretieren wir gemäß der Definition einer Markov-Kette

$$\forall i \in [2:\ell]: \ \forall r, s \in Q: \ \mathrm{Ws}(t_i = s \mid t_{i-1} = r) = p_{r,s}$$

und

$$\forall r \in Q: \ \mathrm{Ws}(t_1 = r) = \pi_r.$$

Wir bemerken hier insbesondere, dass die Wahrscheinlichkeiten der Zustandsübergänge nur von den Zuständen und nicht von den Zeitpunkten t_i mit $i \in [1:\ell]$ selbst abhängen.

Man sagt Markov-Ketten sind *gedächtnislos*, da die Wahrscheinlichkeit des nächsten Zustandes nur vom aktuellen und nicht von den vergangenen Zuständen abhängt. Formal bedeutet dies, dass die folgende Beziehung gilt:

$$\mathrm{Ws}(t_i = x_i \mid t_1 = x_1 \wedge \cdots \wedge t_{i-1} = x_{i-1}) = \mathrm{Ws}(t_i = x_i \mid t_{i-1} = x_{i-1}).$$

Die letzte Gleichung wird oft auch als *Markov-Eigenschaft* bezeichnet.

Die Markov-Eigenschaft bzw. Gedächtnislosigkeit besagt nicht anderes als dass die Wahrscheinlichkeit des Folgezustandes nur vom letzten Zustand abhängt.

Der Vollständigkeit halber wollen wir hier noch erwähnen, dass es auch *Markov-Ketten k-ter Ordnung* gibt. Hierbei hängt die Übergangswahrscheinlichkeit von den

letzten k eingenommenen Zuständen ab. Wie man sich leicht überlegt, lassen sich solche Markov-Ketten k-ter Ordnung durch Markov-Ketten erster Ordnung simulieren (dies sind genau die, die wir in Definition 8.2 definiert haben). Wir wählen als Zustandsmenge nur Q^k, wobei dann ein Zustand der neuen Markov-Kette erster Ordnung ein k-Tupel von Zuständen der Markov-Kette k-ter Ordnung ist, die dann die k zuletzt besuchten Zustände speichern. Die Details der Simulation seien dem Leser zur Übung überlassen.

Im Folgenden werden wir oft Sequenzen betrachten. Dann entsprechen die Zeitpunkte t_i den Positionen i innerhalb der Sequenz, d.h. wir stellen uns den Aufbau einer Sequenz als den Prozess einer Markov-Kette vor. Mit dieser Modellierungen bekommen wir eine Abhängigkeiten der Wahrscheinlichkeiten der Zeichen innerhalb einer Sequenz von ihrer Nachbarschaft.

8.1.2 Wahrscheinlichkeiten von Pfaden

Wir wollen jetzt die Wahrscheinlichkeit von bestimmten Pfaden durch eine Markov-Kette bestimmen. Hierbei benutzen wir oft die folgenden Abkürzung für eine gegebene Folge $X = (x_1, \ldots, x_\ell)$ von Zuständen.

$$\begin{aligned} \mathrm{Ws}(X) &:= \mathrm{Ws}((x_1, \ldots, x_\ell)) \\ &:= \mathrm{Ws}((t_1, \ldots, t_\ell) = (x_1, \ldots, x_\ell)) \\ &:= \mathrm{Ws}(t_1 = x_1 \wedge \cdots \wedge t_\ell = x_\ell). \end{aligned}$$

Sei $X = (x_1, \ldots, x_\ell)$ eine Folge von Zuständen. Die Wahrscheinlichkeit, dass diese Zustandsfolge durchlaufen wird, berechnet sich dann zu

$$\mathrm{Ws}(X) = \pi_{x_1} \cdot \prod_{i=2}^{\ell} p_{x_{i-1},x_i}.$$

Die Anfangswahrscheinlichkeiten π kann auch durch einen neuen Startzustand q_0, der zum Zeitpunkt t_0 eingenommen wird, mit einer Erweiterung der Matrix P der Zustandsübergangswahrscheinlichkeiten eliminiert werden. Wir setzen dann

$$p_{0,j} := \mathrm{Ws}(t_1 = q_j \mid t_0 = q_0) = \pi_{x_j}$$

für $j \in [1:n]$. Weiter definieren wir noch für $i \in [0:n]$:

$$p_{i,0} := 0.$$

Damit vereinfacht sich die Formel zur Berechnung der Wahrscheinlichkeiten für eine Folge $X = (x_1, \ldots, x_\ell)$ von Zuständen zu:

$$\mathrm{Ws}(X) = \prod_{i=1}^{\ell} p_{x_{i-1},x_i}.$$

8.1.3 Beispiel: CpG-Inseln

Im Folgenden wollen wir für ein Beispiel mit biologischem Hintergrund eine Markov-Kette angeben. Im Genom kommt die Basenabfolge CG sehr selten vor (wir schreiben hierfür CpG, damit es nicht mit dem Basenpaar CG verwechselt werden kann, wobei p für den Phosphatrest zwischen den entsprechenden Nukleosiden steht). Dies hat den Hintergrund, dass die beiden Basen in dieser Abfolge chemischen Reaktionen unterworfen sind, die für eine Mutation in der DNS sorgen würde. Man beachte hier, dass mit CG im komplementären Strang der DNS ebenfalls die Basenabfolge CG vorkommt.

Es gibt jedoch Bereiche, wo diese Abfolge überaus häufig auftritt. Es hat sich herausgestellt, dass solche Bereiche, in denen die Folge CpG überdurchschnittlich häufig vorkommt, oft Promotoren enthält. In diesen Bereichen wird die chemische Reaktion von CpG-Paaren in der Regel verhindert. Daher kann man die Kenntnis von Bereichen mit vielen CpG-Teilsequenzen als Kandidaten für Promotoren betrachten, die dann natürlich für das Auffinden von Genen im Genom besonders wichtig sind. Wir formalisieren die Problemstellung wie folgt.

Identifikation von CpG-Inseln

Eingabe: Eine kurze DNS-Sequenz $X = (x_1, \ldots, x_\ell) \in \{A, C, G, T\}^\ell$.
Ausgabe: Befindet sich X innerhalb einer CpG-Insel.

Wir versuchen jetzt mit Hilfe von Markov-Ketten solche CpG-Inseln zu identifizieren. Dazu sind in Abbildung 8.2 die Wahrscheinlichkeiten angegeben, mit denen im Genom auf eine Base X eine Base Y folgt. Dabei ist P^+ die Matrix der Wahrscheinlichkeiten innerhalb einer CpG-Insel und P^- die außerhalb einer solchen.

P^+	A	C	G	T
A	0.18	0.27	0.43	0.12
C	0.17	0.37	0.27	0.19
G	0.16	0.34	0.37	0.13
T	0.08	0.36	0.38	0.18

P^-	A	C	G	T
A	0.30	0.21	0.28	0.21
C	0.32	0.30	0.08	0.30
G	0.25	0.32	0.30	0.21
T	0.18	0.24	0.29	0.29

Abbildung 8.2: Skizze: Zustandsübergangswahrscheinlichkeiten innerhalb und außerhalb von CpG-Inseln

Wir modellieren jeweils eine Markov-Kette für die Bereiche innerhalb bzw. außerhalb der CpG-Inseln. Mit $M^+ = (Q, P^+, \pi)$ bzw. $M^- = (Q, P^-, \pi)$ bezeichnen wir zwei

Markov-Ketten: eine für Sequenzen innerhalb (M^+) und eine außerhalb der CpG-Inseln (M^-). Dabei besteht die Zustandsmenge $Q = \{A, C, G, T\}$ gerade jeweils aus den vier Basen und π ist die Anfangswahrscheinlichkeiten, die man aus den relativen Häufigkeiten der Basen im gesamten Genom ermittelt hat.

Wir berechnen dann die Wahrscheinlichkeit des Pfades X innerhalb der beiden Modelle:

$$\begin{aligned} \mathrm{Ws}_{P^+}(X) &= \pi(x_1) \cdot \prod_{i=2}^{\ell} p^+_{x_{i-1},x_i}, \\ \mathrm{Ws}_{P^-}(X) &= \pi(x_1) \cdot \prod_{i=2}^{\ell} p^-_{x_{i-1},x_i}. \end{aligned}$$

Wie bereits erwähnt, nehmen wir hier für die Anfangswahrscheinlichkeiten an, dass diese für beide Modell gleich ist.

Für die Entscheidung betrachten wir dann den Quotienten der entsprechenden Wahrscheinlichkeiten:

$$\frac{\mathrm{Ws}_{P^+}(X)}{\mathrm{Ws}_{P^-}(X)} = \frac{\pi(x_1) \cdot \prod_{i=2}^{\ell} p^+_{x_{i-1},x_i}}{\pi(x_1) \cdot \prod_{i=2}^{\ell} p^-_{x_{i-1},x_i}} = \prod_{i=2}^{\ell} \frac{p^+_{x_{i-1},x_i}}{p^-_{x_{i-1},x_i}}.$$

Je größer und weiter weg von 1 der Quotient ist, desto wahrscheinlicher ist es, dass wir uns innerhalb einer CpG-Insel befinden.

Da es sich zum einen leichter mit Summen als mit Produkten rechnen lässt und zum anderen die numerischen Stabilität der Ergebnisse erhöht, definieren wir den Score als den Logarithmus des obigen Quotienten:

$$\mathrm{Score}(X) := \log\left(\frac{\mathrm{Ws}_{P^+}(X)}{\mathrm{Ws}_{P^-}(X)}\right) = \sum_{i=2}^{\ell} \left(\log(p^+_{x_{i-1},x_i}) - \log(p^-_{x_{i-1},x_i})\right).$$

Positive Scores deuten dann auf eine CpG-Inseln hin, während negative Scores Bereiche außerhalb einer CpG-Insel charakterisieren.

In der Regel wollen wir nicht für eine kurze Sequenz feststellen, ob sie sich innerhalb oder außerhalb einer CpG-Insel befindet, sondern wir wollen herausfinden, wo sich im Genom solche CpG-Inseln befinden. Daher betrachten wir die folgende Problemstellung, die diesem Ziel Rechnung trägt.

Lokalisieren von CpG-Inseln

Eingabe: Eine lange DNS-Sequenz $X = (x_1, \ldots, x_\ell) \in \{A, C, G, T\}^\ell$.
Ausgabe: Teilsequenzen in X, die CpG-Inseln bilden.

Ein naiver Ansatz würde für alle kurzen Sequenzen der Länge k mit $k \in [100 : 1000]$ den vorherigen Algorithmus anwenden. Problematisch ist hier weniger der Rechenaufwand, der sich durch geeignete Tricks wie beim Karp-Rabin-Algorithmus in Grenzen halten lässt, sondern die Festlegung von k.

Wählt man k zu groß, so wird man CpG-Inseln sicherlich nur schwerlich hinreichend sicher identifizieren können. Wählt man k zu klein, so können die Sequenzen so kurz sein, dass man keine signifikanten Unterscheidungen mehr erhält. Daher werden wir im nächsten Abschnitt versuchen, beide Modelle in einem Modell zu vereinigen.

8.2 Hidden Markov Modelle

Wir kommen jetzt zu einer Verallgemeinerung von Markov-Ketten, den so genannten Hidden Markov Modellen. Hierbei werden die im Modell vorhanden Zustände und die nach außen sichtbaren Ereignisse, die in den Markov-Ketten streng miteinander verbunden waren, voneinander trennen.

8.2.1 Definition

Wir geben zuerst die formale Definition eines Hidden Markov Modells an.

Definition 8.3 *Ein* Hidden Markov Modell M *(kurz* HMM*) ist ein 5-Tupel* $M = (Q, \Sigma, P, S, \pi)$*, wobei:*

- $Q = \{q_1, \ldots, q_n\}$ *eine endliche Menge von Zuständen ist;*
- $\Sigma = \{a_1, \ldots, a_m\}$ *eine endliche Menge von Symbolen ist;*
- P *eine stochastische* $n \times n$*-Matrix der Zustandsübergangswahrscheinlichkeiten ist;*
- S *eine stochastische* $n \times m$*-Matrix der Emissionswahrscheinlichkeiten ist;*
- $\pi = (\pi_1, \ldots, \pi_n)$ *ein stochastischer Vektor der Anfangswahrscheinlichkeiten ist.*

Ein Pfad $X = (x_1, \ldots, x_\ell) \in Q^*$ in M ist wiederum eine Folge von Zuständen. Solche Pfade verhalten sich wie gewöhnliche Markov-Ketten. Nach außen sind jedoch nur Symbole aus Σ sichtbar, wobei zu jedem Zeitpunkt t_i ein Symbol gemäß der

Emissionswahrscheinlichkeiten sichtbar wird. Für einen Pfad $X = (x_1, \ldots, x_\ell) \in Q^*$ in M sind die sichtbaren Symbolen eine Folge $Y = (y_1, \ldots, y_\ell) \in \Sigma^*$, wobei

$$\mathrm{Ws}(y_i = a \mid x_i = q) = s_{q,a}$$

für $a \in \Sigma$ und $q \in Q$.

Gilt $\Sigma = Q$ und $s_{q,q} = 1$ und $s_{q,q'} = 0$ für $q \neq q' \in Q$, dann ist ein Hidden Markov Modell M nichts anderes als unsere bekannte Markov-Kette.

Die Wahrscheinlichkeit einer Zustandsfolge $X = (x_1, \ldots, x_\ell)$ mit der emittierten Symbolfolge $Y = (y_1, \ldots, y_\ell)$ ist für ein Hidden Markov Modell M wie folgt gegeben:

$$\mathrm{Ws}_M(X \wedge Y) = \pi_{x_1} \cdot s_{x_1,y_1} \cdot \prod_{i=2}^{\ell} (p_{x_{i-1},x_i} \cdot s_{x_i,y_i}).$$

Im Allgemeinen ist uns bei den Hidden Markov Modellen jedoch nur Y, nicht aber X bekannt. Dies ist auch der Grund für den Namen, da die Zustandfolge für uns versteckt ist. Allerdings wird die Zustandsfolge X in der Regel die Information beinhalten, an der wir interessiert sind. Wir werden uns daher später mit Verfahren beschäftigen, wie wir aus der emittierten Folge Y die Zustandsfolge X mit großer Wahrscheinlichkeit rekonstruieren können.

8.2.2 Modellierung von CpG-Inseln

Wir werden jetzt die CpG-Inseln mit Hilfe von Hidden Markov Ketten modellieren. Das Hidden Markov Modell $M = (Q, \Sigma, P, S, \pi)$ wird dann wie folgt definiert. Als Zustandsmenge wählen wir $Q = \{A^+, C^+, G^+, T^+, A^-, C^-, G^-, T^-\}$. Für jede der möglichen Basen wählen wir zwei Ausprägungen: eine, sofern sich die Base innerhalb der CpG-Insel befindet (die mit + indizierten), und eine, sofern sie sich außerhalb befindet (die mit − indizierten). Das Symbolalphabet ist dann $\Sigma = \{A, C, G, T\}$. Nach außen können wir ja zunächst nicht feststellen, ob wir innerhalb oder außerhalb einer CpG-Insel sind.

Die Zustandsübergangsmatrix sieht wie folgt aus

$$P = \left(\begin{array}{ccc|ccc} & & & \frac{p}{4} & \cdots & \frac{p}{4} \\ & P^+ \cdot (1-p) & & \vdots & & \vdots \\ & & & \frac{p}{4} & \cdots & \frac{p}{4} \\ \hline \frac{q}{4} & \cdots & \frac{q}{4} & & & \\ \vdots & & \vdots & & P^- \cdots (1-q) & \\ \frac{q}{4} & \cdots & \frac{q}{4} & & & \end{array}\right).$$

Wir haben hier zusätzlich noch Zustandsübergangswahrscheinlichkeiten definiert mit denen wir aus einer CpG-Insel bzw. in eine CpG-Insel übergehen können. Dabei

gelangen wir mit Wahrscheinlichkeit p aus einer CpG-Inseln und mit Wahrscheinlichkeit q in eine CpG-Insel. Dazu mussten wir natürlich die Zustandsübergangswahrscheinlichkeiten innerhalb bzw. außerhalb der CpG-Inseln mit $(1-p)$ bzw. $(1-q)$ normieren.

Für die Übergänge in bzw. aus einer CpG-Insel haben wir uns das Leben leicht gemacht und haben die Wahrscheinlichkeit von p bzw. q auf die einzelnen Zustände aus $\{A^+, C^+, G^+, T^+\}$ bzw. $\{A^-, C^-, G^-, T^-\}$ gleich verteilt.

Für die Emissionswahrscheinlichkeiten gilt $s_{a^+,a} = s_{a^-,a} = 1$ und $s_{a^+,b} = s_{a^-,b} = 0$ für $a \neq b \in \Sigma$. Hier finden die Emission deterministisch, also anhand des Zustandes statt. Im folgenden Beispiel werden wir sehen, dass es bei Hidden Markov Modellen auch möglich sein kann, von allen Zuständen aus alle Zeichen (eben mit unterschiedlichen Wahrscheinlichkeiten) zu emittieren.

8.2.3 Modellierung eines gezinkten Würfels

Wir geben nun noch ein Hidden Markov Modell an, bei dem die Emissionswahrscheinlichkeiten weniger mit dem Zustand korreliert sind als im vorherigen Beispiel der Modellierung der CpG-Inseln. Wir nehmen an, ein Croupier besitzt zwei Würfel, einen normalen und einen gezinkten. Daher besteht die Zustandsmenge $Q = \{F, U\}$ aus nur zwei Zuständen, je nachdem, ob der normale (F=Fair) oder der gezinkte Würfel (U=Unfair) benutzt wird.

Das Alphabet $\Sigma = [1:6]$ modelliert dann die gewürfelte Anzahl Augen. Die Emissionswahrscheinlichkeiten sind dann $s_{F,i} = \frac{1}{6}$ für $i \in [1:6]$ sowie $s_{U,i} = \frac{1}{10}$ für $i \in [1:5]$ und $s_{U,6} = \frac{1}{2}$. Beim fairen Würfel sind alle Seiten gleich wahrscheinlich, während beim unfairen Würfel der Ausgang von sechs Augen erheblich wahrscheinlicher ist.

Die Matrix der Zustandsübergangswahrscheinlichkeiten legen wir mittels

$$P = \begin{pmatrix} 0.9 & 0.1 \\ 0.2 & 0.8 \end{pmatrix}$$

fest. Dies sind die Wahrscheinlichkeiten, mit denen der Croupier den gerade verwendeten Würfel wechselt oder auch nicht. Beispielsweise behält der Croupier mit Wahrscheinlichkeit 0.9 den normalen Würfel bei.

8.3 Viterbi-Algorithmus

In diesem Abschnitt wollen wir zeigen, wie sich zu einer gegebenen emittierten Symbolfolge eine Zustandsfolge konstruieren lässt, die in dem betrachten Hidden Markov Modell die größte Wahrscheinlichkeit für die gegebene Symbolfolge besitzt.

8.3.1 Decodierungsproblem

Zuerst einmal wollen wir die Aufgabenstellung noch ein wenig formalisieren und präzisieren.

DECODIERUNGSPROBLEM

Eingabe: Ein HMM $M = (Q, \Sigma, P, S, \pi)$ und eine Folge $Y \in \Sigma^*$.
Ausgabe: Ein wahrscheinlichster Pfad $X \in Q^*$ für Y in M, d.h. ein $X \in Q^*$, so dass $\mathrm{Ws}_M(X \mid Y)$ maximal ist.

Das bedeutet wir suchen eine Folge $X \in Q^*$ mit

$$\mathrm{Ws}(X \mid Y) = \max\{\mathrm{Ws}(Z \mid Y) \ : \ Z \in Q^*\}\,.$$

Dafür schreiben wir auch

$$X \in \mathrm{argmax}\{\mathrm{Ws}(Z \mid Y) \ : \ Z \in Q^*\}\,.$$

Wir schreiben hier $\in$ statt $=$, da es ja durchaus mehrere verschiedene Folgen mit der gleichen höchsten Wahrscheinlichkeit geben kann. Oft verwendet man hier auch etwas schlampig das Gleichheitszeichen, ohne das nachgewiesen wurde, dass das Maximum eindeutig ist. Man verwendet sogar das Gleichheitszeichen, wenn das Maximum eben nicht eindeutig ist.

8.3.2 Dynamische Programmierung

Wir lösen jetzt das Decodierungsproblem mit Hilfe der Dynamischen Programmierung. Dazu definieren wir:

$$\tilde{W}_q(i) := \max\{\mathrm{Ws}((x_1, \ldots, x_{i-1}, q) \mid (y_1, \ldots, y_i)) \ : \ (x_1, \ldots, x_{i-1}) \in Q^*\}\,.$$

Offensichtlich sind wir eigentlich am Wert von $\max\{\tilde{W}_q(|Y|) : q \in Q\}$ interessiert. Wie wir sehen werden, sind die anderen Werte Hilfsgrößen, die uns bei der Bestimmung einer wahrscheinlichsten Zustandsfolge helfen werden.

Mit Hilfe der Definition der bedingten Wahrscheinlichkeit $\mathrm{Ws}(X \mid Y) = \frac{\mathrm{Ws}(X \wedge Y)}{\mathrm{Ws}(Y)}$ erhalten wir:

$$\tilde{W}_q(i) := \max\left\{ \frac{\mathrm{Ws}((x_1,\ldots,x_{i-1},q) \wedge (y_1,\ldots,y_i))}{\mathrm{Ws}((y_1,\ldots,y_i))} \;:\; (x_1,\ldots,x_{i-1}) \in Q^* \right\}.$$

Da der Nenner ja nicht von der Zustandsfolge X abhängt, über die ja maximiert werden soll, genügt es den Zähler allein zu maximieren. Wir versuchen daher im Folgenden eine Zustandsfolge X zu finden, so dass $\mathrm{Ws}((x_1,\ldots,x_{i-1},q) \wedge (y_1,\ldots,y_i))$ maximal wird, und definieren hierfür:

$$W_q(i) := \max\left\{ \mathrm{Ws}((x_1,\ldots,x_{i-1},q) \wedge (y_1,\ldots,y_i)) \;:\; (x_1,\ldots,x_{i-1}) \in Q^* \right\}.$$

Es gilt offensichtlich:

$$\begin{aligned} W_q(1) &= \mathrm{Ws}(y_1 \wedge q) = \pi_q \cdot s_{q,y_1}, \\ W_q(i+1) &= \max\left\{ W_{q'}(i) \cdot p_{q',q} \cdot s_{q,y_{i+1}} \;:\; q' \in Q \right\}. \end{aligned}$$

Offensichtlich gilt $\mathrm{Ws}(Y \wedge X) = \max\{W_q(\ell) \;:\; q \in Q\}$. Somit müssen wir eine Matrix der Größe $O(|Q| \cdot \ell)$ berechnen. Für jeden Eintrag muss dabei ein Maximum über $|Q|$ Werte gebildet werden. Somit ist die Laufzeit $O(|Q|^2 \cdot \ell)$. Der obige, auf dynamischer Programmierung basierende Algorithmus ist in der Literatur unter dem Namen *Viterbi-Algorithmus* bekannt.

Damit haben wir jedoch nur die Wahrscheinlichkeit einer wahrscheinlichsten Folge von Zuständen berechnet. Die Zustandsfolge selbst können wir jedoch wie beim paarweisen Sequenzen Alignment berechnen, wenn wir uns bei jeder Maximumsbildung merken, welcher Wert gewonnen hat. Ebenso wie beim paarweisen Sequenzen Alignment können wir den Platzbedarf mit Hilfe derselben Techniken des Hirschberg-Algorithmus auf $(|Q|)$ senken.

Theorem 8.4 *Sei $M = (Q, \Sigma, P, S, \pi)$ ein Hidden Markov Modell und $Y \in \Sigma^*$. Eine Zustandsfolge $X \in Q^*$, die $\mathrm{Ws}(X \mid Y)$ maximiert, kann in Zeit $O(|Q|^2 \cdot |Y|)$ und Platz $O(|Q|)$ konstruiert werden.*

8.3.3 Implementierungstechnische Details

Die Berechnung von Produkten und Division ist zum einen nicht sonderlich einfach und zum anderen auch nicht immer numerisch stabil, insbesondere wenn die Werte (wie hier) ziemlich klein werden. Daher rechnen wir mit den Logarithmen der Wahrscheinlichkeiten, damit wir es nur mit Addition und Subtraktion zu tun haben.

Außerdem erhöht dies die numerische Stabilität, da im Rechner sich kleine Zahlen nur schwer speichern lassen und durch das Logarithmieren betragsmäßig größer werden.

Wir definieren also $\hat{W}_q(i)$ anstelle $W_q(i)$ wie folgt:

$$\hat{W}_q(i) := \max\left\{\log(\mathrm{Ws}((x_1,\ldots,x_{i-1},q) \wedge (y_1,\ldots,y_i))) \;:\; (x_1,\ldots,x_{i-1}) \in Q^*\right\}.$$

Dafür ergeben sich dann die folgenden Rekursionsgleichungen:

$$\begin{aligned} \hat{W}_q(1) &= \log(\pi_q) + \log(s_{q,y_1}), \\ \hat{W}_q(i+1) &= \log(s_{q,y_{i+1}}) + \max\left\{\hat{W}_{q'}(i) + \log(p_{q',q}) \;:\; q' \in Q\right\}. \end{aligned}$$

Wie man sieht muss man nur einmal zu Beginn die teuren Kosten für das Logarithmieren investieren.

8.4 Posteriori-Decodierung

Wir wollen jetzt noch eine andere Möglichkeit angeben, wie man die Zustandsfolge im Hidden Markov Modell für eine gegebene emittierte Sequenz Y rekonstruieren kann. Hierzu wollen wir nun den Zustand zum Zeitpunkt t_i bestimmen, der die größte Wahrscheinlichkeit unter der Annahme besitzt, dass insgesamt die Sequenz Y emittiert wurde.

Wir geben hier also explizit die Forderung auf, eine Zustandsfolge im Hidden Markov Modell zu betrachten. Wir werden später sehen, dass sich aus der Kenntnis der wahrscheinlichsten Zuständen zu den verschiedenen Zeitpunkten t_i wieder eine Zustandsfolge rekonstruieren lässt. Allerdings kann diese Folge illegal in dem Sinne sein, dass für zwei aufeinander folgende Zustände x_{i-1} und x_i gilt, dass $p_{x_{i-1},x_i} = 0$ ist. Dies folgt methodisch daher, dass wir nur die Zustände gewählt haben, die zu einem festen Zeitpunkt am wahrscheinlichsten sind, aber nicht die Folge samt Zustandsübergangswahrscheinlichkeiten berücksichtigt haben.

Wir kommen darauf später noch einmal zurück und geben jetzt die formalisierte Problemstellung an.

Posteriori-Decodierung

Eingabe: Ein HMM $M = (Q, \Sigma, P, S, \pi)$ und $Y \in \Sigma^\ell$.
Ausgabe: Bestimme $\mathrm{Ws}(x_i = q \mid Y)$ für alle $i \in [1:\ell]$ und für alle $q \in Q$.

8.4.1 Ansatz zur Lösung

Um jetzt für die Posteriori-Decodierung einen Algorithmus zu entwickeln, formen wir die gesuchte Wahrscheinlichkeit erst noch einmal ein wenig mit Hilfe der Definition für bedingte Wahrscheinlichkeiten $\mathrm{Ws}(X|Y) = \frac{\mathrm{Ws}(X \wedge Y)}{\mathrm{Ws}(Y)}$ um:

$$\begin{aligned} &\mathrm{Ws}(x_i = q \mid Y) \\ &\quad = \frac{\mathrm{Ws}(x_i = q \wedge Y)}{\mathrm{Ws}(Y)} \end{aligned}$$

Nach dem so genannten Multiplikationssatz gilt $\mathrm{Ws}(X \wedge Y) = \mathrm{Ws}(X) \cdot \mathrm{Ws}(Y \mid X)$ und wir erhalten somit:

$$= \frac{\mathrm{Ws}((y_1, \ldots, y_i) \wedge x_i = q) \cdot \mathrm{Ws}((y_{i+1}, \ldots, y_\ell) \mid (y_1, \ldots, y_i) \wedge x_i = q)}{\mathrm{Ws}(Y)}$$

Aufgrund der Definition von Hidden Markov Modellen folgt, dass die emittierten Symbole $(y_{i+1}, \ldots, y_\ell)$ nur von den Zuständen $(x_{i+1}, \ldots, x_\ell)$ abhängen. Aufgrund der Markov-Eigenschaft hängen die Zustände $(x_{i+1}, \ldots, x_\ell)$ aber nur vom Zustand x_i ab. Somit folgt:

$$\begin{aligned} &= \frac{\mathrm{Ws}((y_1, \ldots, y_i) \wedge x_i = q) \cdot \mathrm{Ws}((y_{i+1}, \ldots, y_\ell) \mid x_i = q)}{\mathrm{Ws}(Y)} \\ &= \frac{f_q(i) \cdot b_q(i)}{\mathrm{Ws}(Y)} \end{aligned}$$

Hierbei haben wir für die Umformung in der letzten Zeile die folgende Notation verwendet:

$$\begin{aligned} f_q(i) &:= \mathrm{Ws}(x_i = q \wedge (y_1, \ldots, y_i)), \\ b_q(i) &:= \mathrm{Ws}((y_{i+1}, \ldots, y_\ell) \mid x_i = q). \end{aligned}$$

Die hier eingeführten Wahrscheinlichkeiten $f_q(i)$ bzw. $b_q(i)$ werden wir im Folgenden als *Vorwärtswahrscheinlichkeiten* bzw. als *Rückwärtswahrscheinlichkeiten* bezeichnen. Wie wir noch sehen werden ist der Grund hierfür ist, dass sich bei $f_q(i)$ in der Zustandsfolge vorwärts aus $f_q(1), \ldots, f_q(i-1)$ berechnen lässt, während $b_q(i)$ in der Zustandsfolge rückwärts aus $b_q(\ell), \ldots, b_q(i+1)$ berechnen lässt.

8.4.2 Vorwärts-Algorithmus

Wir werden uns jetzt überlegen, wie sich die Vorwärtswahrscheinlichkeiten mit Hilfe der dynamische Programmierung berechnen lässt. Dazu stellen wir fest, dass die

folgenden Rekursionsgleichungen gelten:

$$\begin{aligned} f_q(1) &= \pi_q \cdot s_{q,y_1}, \\ f_q(i+1) &= s_{q,y_{i+1}} \cdot \sum_{q' \in Q} (f_{q'}(i) \cdot p_{q',q}). \end{aligned}$$

Diese Rekursionsgleichungen sind denjenigen aus dem Viterbi-Algorithmus sehr ähnlich. Wir haben hier nur die Summe statt einer Maximumsbildung, da wir über alle Pfade in den gesuchten Zustand gelangen dürfen. Dies ist in letzter Instanz die Folge dessen, dass wir nach einem wahrscheinlichen Zustand zum Zeitpunkt t_i suchen und nicht nach einer wahrscheinlichen Zustandsfolge für die Zeitpunkte $(t_1, \ldots, t_i)$.

Wir halten noch fest, dass offensichtlich $\mathrm{Ws}(Y) = \sum_{q \in Q} f_q(\ell)$ gilt.

Lemma 8.5 *Sei $M = (Q, \Sigma, P, S, \pi)$ ein Hidden Markov Modell und $Y \in \Sigma^\ell$. Die Vorwärtswahrscheinlichkeit $f_q(i) = \mathrm{Ws}(x_i = q \wedge (y_1, \ldots, y_i))$ kann für alle $i \in [1:\ell]$ und alle $q \in Q$ in Zeit $O(|Q|^2 \cdot |Y|)$ und Platz $O(|Q|)$ berechnet werden.*

Den Algorithmus zur Berechnung der Vorwärtswahrscheinlichkeiten wird in der Literatur auch oft als *Vorwärts-Algorithmus* bezeichnet.

8.4.3 Rückwärts-Algorithmus

Wir werden uns jetzt überlegen, wie sich die Rückwärtswahrscheinlichkeiten mit Hilfe der dynamische Programmierung berechnen lässt. Dazu stellen wir fest, dass die folgenden Rekursionsgleichungen gelten:

$$\begin{aligned} b_q(n) &= 1, \\ b_q(i) &= \sum_{q' \in Q} (p_{q,q'} \cdot s_{q',y_{i+1}} \cdot b_{q'}(i+1)). \end{aligned}$$

Auch hier gilt wieder $\mathrm{Ws}(Y) = \sum_{q \in Q} \pi_1 \cdot s_{q,y_1} \cdot b_q(1)$.

Lemma 8.6 *Sei $M = (Q, \Sigma, P, S, \pi)$ ein Hidden Markov Modell und $Y \in \Sigma^\ell$. Die Rückwärtswahrscheinlichkeit $b_q(i) = \mathrm{Ws}((y_{i+1}, \ldots, y_\ell) \mid x_i = q)$ kann für alle $i \in [1:\ell]$ und alle $q \in Q$ in Zeit $O(|Q|^2 \cdot |Y|)$ und Platz $O(|Q|)$ berechnet werden.*

Den Algorithmus zur Berechnung der Rückwärtswahrscheinlichkeiten wird in der Literatur auch oft als *Rückwärts-Algorithmus* bezeichnet.

Aus den beiden letzten Lemmata folgt aus unserer vorherigen Diskussion unmittelbar das folgende Theorem.

Theorem 8.7 *Sei $M = (Q, \Sigma, P, S, \pi)$ ein Hidden Markov Modell und $Y \in \Sigma^\ell$. Die Posteriori-Wahrscheinlichkeit $\mathrm{Ws}(x_i = q \mid Y)$ kann für alle $i \in [1 : \ell]$ und alle $q \in Q$ in Zeit $O(|Q|^2 \cdot |Y|)$ und Platz $O(|Q|)$ berechnet werden.*

8.4.4 Implementierungstechnische Details

Auch hier ist die Berechnung von Produkten und Division zum einen nicht sonderlich einfach und zum anderen auch nicht immer numerisch stabil, insbesondere wenn die Werte (wie hier) ziemlich klein werden. Daher rechnen wir mit den Logarithmen der Wahrscheinlichkeiten, da wir es dann nur mit Addition und Subtraktion zu tun haben. Wie bereits erwähnt, erhöht sich dadurch ebenfalls die numerische Stabilität.

Wir definieren also jetzt die *logarithmischen Vorwärts- und Rückwärtswahrscheinlichkeiten*:

$$\begin{aligned} \hat{f}_q(i) &:= \log(f_q(i)) = \log(\mathrm{Ws}(x_i = q \wedge (y_1, \ldots, y_i))), \\ \hat{b}_q(i) &:= \log(b_q(i)) = \log(\mathrm{Ws}((y_{i+1}, \ldots, y_\ell) \mid x_i = 1)). \end{aligned}$$

Dazu stellen wir fest, dass dann die folgenden Rekursionsgleichungen

$$\begin{aligned} \hat{f}_q(1) &= \log(\pi_q) + \log(s_{q,y_1}), \\ \hat{f}_q(i+1) &= \log(s_{q,y_{i+1}}) + \log\left(\sum_{q' \in Q} (\exp(\hat{f}_{q'}(i)) \cdot p_{q',q}\right) \end{aligned}$$

für die logarithmischen Vorwärtswahrscheinlichkeiten gelten und analog für die logarithmischen Rückwärtswahrscheinlichkeiten

$$\begin{aligned} \hat{b}_q(n) &= \log(s_{q,y_\ell}), \\ \hat{b}_q(i) &= \log\left(\sum_{q' \in Q} (p_{q,q'} \cdot s_{q',y_{i+1}} \cdot \exp(b_{q'}(i+1)))\right) \end{aligned}$$

gelten. Da hier nun statt der Maximumsbildung eine Summe involviert ist, müssen wir bei der dynamischen Programmierung sowohl Logarithmieren als auch Exponenzieren.

8.4.5 Anwendung

Mit Hilfe der Posteriori-Decodierung können wir jetzt einen alternativen Pfad $\tilde{X}$ wie folgt definieren:

$$\tilde{x}_i :\in \operatorname{argmax}\{\mathrm{Ws}(x_i = q \mid Y) \ : \ q \in Q\}\,.$$

Wir merken hier noch an, dass die Zustandsfolge nicht unbedingt eine zulässige Zustandsfolge sein muss. In der Folge $\tilde{X}$ können durchaus die Teilsequenz $(\tilde{x}_{i-1}, \tilde{x}_i)$ auftreten, so dass $p_{\tilde{x}_{i-1},\tilde{x}_i} = 0$. Dies folgt daraus, dass wir bei der Konstruktion der Folge $\tilde{X}$ für den Zeitpunkt t_i immer nur den Zustand auswählen, der die größte Wahrscheinlichkeit für die beobachtete emittierte Sequenz Y besitzt. Wir achten dabei im Gegensatz zum Viterbi-Algorithmus nicht darauf, eine ganze konsistente Sequenz zu betrachten.

Dies wollen wir uns noch einmal an dem folgenden Beispiel des Hidden Markov Modells $M = (Q, \Sigma, P, S, \pi)$ klar machen. Hierbei ist $Q = \{q_1, \ldots, q_5\}$, $\Sigma = \{a, b\}$ und $\pi = (1, 0, 0, 0, 0)$. Die Matrix P der Zustandsübergangswahrscheinlichkeiten ist in der Abbildung 8.3 illustriert.

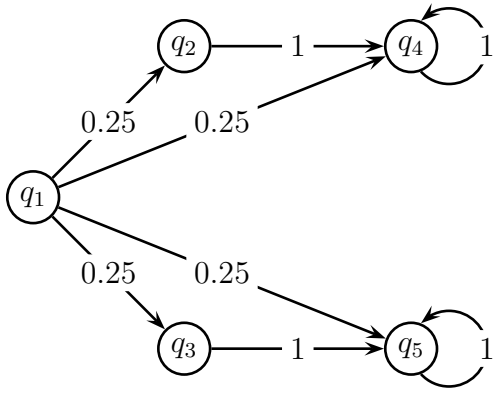


Abbildung 8.3: Beispiel: Ein HMM mit illegaler a posteriori Zustandsfolge

Die Matrix S der Emissionswahrscheinlichkeiten ist wie folgt gegeben:

$$S = \begin{pmatrix} s_1 := s_{q_1,a} & s_{q_1,b} = 1 - s_1 \\ s_2 := s_{q_2,a} & s_{q_2,b} = 1 - s_2 \\ s_3 := s_{q_3,a} & s_{q_3,b} = 1 - s_3 \\ s_4 := s_{q_4,a} & s_{q_4,b} = 1 - s_4 \\ s_5 := s_{q_5,a} & s_{q_5,b} = 1 - s_5 \end{pmatrix}.$$

Wir wählen jetzt $s_1 = 1$, $s_2 = 0.8$, $s_3 = 0.6$, $s_4 = 0.4$ und $s_5 = 0.5$

Für die Zeichenfolge *aaa* kann man jetzt die Posteriori-Wahrscheinlichkeit (ausgehend von den zugehörigen Vorwärts- und Rückwärtswahrscheinlichkeiten) zu den

Zeitpunkten t_1, t_2 und t_3 für alle Zustände bestimmen. In der Abbildung 8.4 wurde die Vorwärts- und Rückwärtswahrscheinlichkeiten $f_q(i)$ und $b_q(i)$ sowie die daraus resultierenden Posteriori-Wahrscheinlichkeiten $f_q(i) \cdot b_q(i)$ berechnet.

$f_q(i)$	1	2	3
q_1	1.0	0.0	0.0
q_2	0.0	$\frac{s_2}{4}$	0.0
q_3	0.0	$\frac{s_3}{4}$	0.0
q_4	0.0	$\frac{s_4}{4}$	$\frac{s_2 \cdot s_4 + s_4^2}{4}$
q_5	0.0	$\frac{s_5}{4}$	$\frac{s_3 \cdot s_5 + s_5^2}{4}$

$b_q(i)$	1	2	3
q_1	$\cdots$	$\frac{1}{4}$	1.0
q_2	$\cdots$	s_4	1.0
q_3	$\cdots$	s_5	1.0
q_4	$\cdots$	s_4	1.0
q_5	$\cdots$	s_5	1.0

$f_q(i)$	1	2	3
q_1	1.0000	0.0000	0.0000
q_2	0.0000	0.2000	0.0000
q_3	0.0000	0.1500	0.0000
q_4	0.0000	0.1000	0.1200
q_5	0.0000	0.1250	0.1375

$b_q(i)$	1	2	3
q_1	$\cdots$	0.2500	1.0000
q_2	$\cdots$	0.4000	1.0000
q_3	$\cdots$	0.5000	1.0000
q_4	$\cdots$	0.4000	1.0000
q_5	$\cdots$	0.5000	1.0000

$f_q(i) \cdot b_q(i)$	1	2	3
q_1	$\cdots$	0.0000	0.0000
q_2	$\cdots$	0.0800	0.0000
q_3	$\cdots$	0.0750	0.0000
q_4	$\cdots$	0.0400	0.1200
q_5	$\cdots$	0.0625	0.1375

Abbildung 8.4: Beispiel: Die expliziten Vorwärts- und Rückwärts- sowie Posteriori-Wahrscheinlichkeiten für M

Wie man daraus ablieẞt, ist für die Ausgabefolge *aaa* der wahrscheinlichste Zustand zum Zeitpunkt t_2 bzw. t_3 gerade q_2 bzw. q_5. Es gilt jedoch $p_{q_2,q_5} = 0$. Somit enthält die mit dieser Methode vorhergesagte Zustandsfolge für die emittierte Folge *aaa* einen illegalen Zustandsübergang.

Weiterhin lassen sich mit Hilfe der Posteriori-Decodierung auch wichtige Eigenschaften von der konstruierten Folge $\tilde{X}$ ableiten, die sich mit Hilfe einer Funktion $g : Q \to \mathbb{R}$ beschreiben lassen. Dazu definieren wir basierend auf der emittierten Folge Y und der gegebenen Funktion g eine Funktion G, die jedem Zeitpunkt einen

Wert wie folgt zuordnet:

$$G(i \mid Y) := \sum_{q \in Q} \mathrm{Ws}(x_i = q) \cdot g(q).$$

Ein Beispiel hierfür ist unser Problem der CpG-Inseln. Hierfür definieren wir die Funktion g wie folgt:

$$g:\ Q \to \mathbb{R}:\ Q \ni q \mapsto \begin{cases} 1 & \text{für } q \in \{A^+, C^+, G^+, T^+\} \\ 0 & \text{für } q \in \{A^-, C^-, G^-, T^-\} \end{cases}.$$

$G(i \mid Y)$ kann man dann als die a posteriori Wahrscheinlichkeit interpretieren, dass sich das Zeichen an Position i (d.h. zum Zeitpunkt t_i) innerhalb einer CpG-Insel befindet.

8.5 Schätzen von HMM-Parametern

In diesem Abschnitt wollen wir uns mit dem Schätzen der Parameter eines Hidden Markov Modells beschäftigen. Hierbei nehmen wir an, dass uns die Zustände des Hidden Markov Modells selbst bekannt sind. Wir wollen also versuchen, die stochastischen Matrizen P und S zu schätzen.

Um diese stochastischen Matrizen schätzen zu können, werden wir einige Folgen der emittierten Symbole, also $Y^{(1)}, \ldots, Y^{(m)} \in \Sigma^*$ der Länge $\ell^{(1)}, \ldots, \ell^{(m)}$ betrachten und versuchen, auf die gesuchten Wahrscheinlichkeiten zu schließen. Wir werden jetzt zwei Fälle unterscheiden, je nachdem, ob uns auch die zugehörige Zustandsfolgen $X^{(1)}, \ldots, X^{(m)}$ bekannt sind oder nicht. Die Folgen $Y^{(i)}$ werden oft auch *Trainingssequenzen* genannt. Wir schreiben im Folgenden $\vec{Y} = (Y^{(1)}, \ldots, Y^m)$ und mit Y bezeichen wir Element aus $\vec{Y}$, wenn uns der obere Index nicht wichtig ist, d.h. $Y = Y^{(i)}$ für ein $i \in [1 : m]$.

8.5.1 Zustandsfolge bekannt

Ein Beispiel, für das auch die Zustandsfolgen unter gewissen Umständen bekannt sind, ist das Problem der CpG-Inseln. Hier können wir aus anderen, biologischen Untersuchungen auch noch einige CpG-Inseln und somit die dort durchlaufenen Zustandfolgen kennen. In diesem Beispiel ist uns sogar die stochastische Matrix S der Emissionswahrscheinlichkeiten bereits bekannt.

Wir zählen jetzt die Anzahl $\hat{p}_{q,q'}$ der Transition $q \to q'$ und die Anzahl $\hat{s}_{q,a}$ der Emission von a aus dem Zustand q. Als Schätzer wählen wir dann ganz einfach die entsprechenden relativen Häufigkeiten:

$$p_{q,q'} := \frac{\hat{p}_{q,q'}}{\sum_{r \in Q} \hat{p}_{q,r}}, \tag{8.1}$$

$$s_{q,a} := \frac{\hat{s}_{q,a}}{\sum_{\sigma \in \Sigma} \hat{s}_{q,\sigma}}. \tag{8.2}$$

Ist die Anzahl der Trainingssequenzen sehr klein, so verändern wir die Schätzer ein wenig, um künstliche Wahrscheinlichkeiten von 0 zu vermeiden. Dazu erhöhen wir die jeweilige beobachtete Anzahl um 1 und passen die Gesamtanzahl entsprechend an. Die Anzahl der Transitionen aus dem Zustand q erhöhen wir also um $|Q|$, da wir für jede mögliche Transition (q, q') eine künstliche hinzugenommen haben. Die Anzahl der emittierten Zeichen aus dem Zustand q um $|\Sigma|$, da wir für jedes Zeichen $a \in \Sigma$ eine weitere künstliche Emission hinzugefügt haben.

$$p'_{q,q'} := \frac{1 + \hat{p}_{q,q'}}{|Q| + \sum_{r \in Q} \hat{p}_{q,r}}, \tag{8.3}$$

$$s'_{q,a} := \frac{1 + \hat{s}_{q,a}}{|\Sigma| + \sum_{\sigma \in \Sigma} \hat{s}_{q,\sigma}}. \tag{8.4}$$

Der Grund für die letzte Modifikation ist, dass es ein qualitativer Unterschied ist, ob eine Übergangswahrscheinlichkeit gleich Null oder nur sehr klein ist. Bei einer geringen Anzahl von Trainingssequenzen haben wir keinen Grund, eine Übergangswahrscheinlichkeit von Null anzunehmen, da das Nichtauftreten nur aufgrund der kleinen Stichprobe möglich ist. Daher vergeben wir bei einer relativen Häufigkeit von Null bei kleinen Stichproben lieber eine kleine Wahrscheinlichkeit.

8.5.2 Zustandsfolge unbekannt — Baum-Welch-Algorithmus

In der Regel werden die Zustandsfolgen, die zu den bekannten Trainingssequenzen gehören, jedoch unbekannt sein. In diesem Fall ist das Problem wieder $\mathcal{NP}$-hart (unter geeigneten Definitionen der gewünschten Eigenschaften der Schätzer). Daher werden wir in diesem Fall eine Heuristik zum Schätzen der Parameter vorstellen.

Auch hier werden wir wieder versuchen die relative Häufigkeit als Schätzer zu verwenden. Dazu werden gemäß gegebener Matrizen P bzw. S für die Zustandsübergangswahrscheinlichkeiten bzw. für die Emissionswahrscheinlichkeiten die Zustandsfolge ermittelt und wir können dann wieder die relativen Häufigkeiten abschätzen.

Woher erhalten wir jedoch die Matrizen P und S? Wir werden zuerst mit beliebigen stochastischen Matrizen P und S starten und dann mit Hilfe der neuen Schätzungen versuchen die Schätzungen immer weiter iterativ zu verbessern.

Seien P und S also beliebige stochastische Matrizen. Wir versuchen zuerst die Anzahl der Transitionen (q, q') unter Annahme dieser Matrizen als Zustandsübergangs- bzw. Emissionswahrscheinlichkeiten zu bestimmen. Es gilt für eine beliebige emittierte Folge Y (wobei wir wie bei der Berechnung der Vorwärts- und Rückwärtswahrscheinlichkeiten den Multiplikationssatz ($\mathrm{Ws}(X, Y) = \mathrm{Ws}(X) \cdot \mathrm{Ws}(Y \mid X)$) und die Markov-Eigenschaft bei fast jeder Zeile ausnutzen):

$$
\begin{aligned}
&\mathrm{Ws}(x_i = q, x_{i+1} = q' \mid Y, P, S) \\
&= \frac{\mathrm{Ws}(Y, x_i = q, x_{i+1} = q' \mid P, S)}{\mathrm{Ws}(Y)} \\
&= \frac{\mathrm{Ws}((y_1, \ldots, y_i), x_i = q \mid P, S) \cdot \mathrm{Ws}((y_{i+1}, \ldots, y_\ell), x_{i+1} = q' \mid x_i = q, P, S)}{\mathrm{Ws}(Y)} \\
&= \frac{1}{\mathrm{Ws}(Y)} \Big(\mathrm{Ws}((y_1, \ldots, y_i), x_i = q \mid P, S) \cdot \mathrm{Ws}(x_{i+1} = q' \mid x_i = q, P, S) \cdot \\
&\qquad \cdot \mathrm{Ws}((y_{i+1}, \ldots, y_\ell) \mid x_{i+1} = q', P, S) \Big) \\
&= \frac{1}{\mathrm{Ws}(Y)} \Big(\mathrm{Ws}((y_1, \ldots, y_i), x_i = q \mid P, S) \cdot \mathrm{Ws}(x_{i+1} = q' \mid x_i = q, P, S) \cdot \\
&\qquad \cdot \mathrm{Ws}(y_{i+1} \mid x_{i+1} = q', P, S) \cdot \mathrm{Ws}((y_{i+2}, \ldots, y_\ell) \mid x_{i+1} = q', P, S) \Big) \\
&= \frac{f_q(i) \cdot p_{q,q'} \cdot s_{q', y_{i+1}} \cdot b_{q'}(i+1)}{\mathrm{Ws}(Y)}.
\end{aligned}
$$

Die in der letzten Umformung auftretenden Terme $f_q(i)$ und $b_q(i)$ sind die bereits bekannten (und somit auch effizient berechenbaren) Vorwärts- und Rückwärtswahrscheinlichkeiten unter Berücksichtigung der Matrix P der Zustandsübergangswahrscheinlichkeiten und der Matrix S der Emissionswahrscheinlichkeiten.

Damit haben wir jetzt die Wahrscheinlichkeit der Transitionen $(q \to q')$ für die emittierte Symbolfolge Y unter Berücksichtigung der Matrizen P bzw. S für die Zustandsübergangs- bzw. Emissionswahrscheinlichkeiten bestimmt, die sich mit Hilfe der verwendeten Vorwärts- und Rückwärtswahrscheinlichkeiten effizient berechnen können. Somit können wir jetzt einen neuen Schätzer (basierend auf P und S) für die erwartete Anzahl der Zustandsübergange $(q \to q')$ in den Trainingssequenzen und somit mithilfe der Gleichung 8.1 bzw. 8.3 auch für die zugehörigen Zustandübergangswahrscheinlichkeit angeben:

$$
\begin{aligned}
\hat{p}_{q,q'} &:= \mathbb{E}((q \to q') \mid \vec{Y}, P, S) \\
&= \sum_{j=1}^{m} \sum_{i=1}^{\ell^{(j)}} \mathrm{Ws}(x_i = q, x_{i+1} = q' \mid Y^{(j)}, P, S)
\end{aligned}
$$

$$= \sum_{j=1}^{m} \frac{1}{\mathrm{Ws}(Y^{(j)})} \sum_{i=1}^{\ell^{(j)}} f_q^{(j)}(i) \cdot p_{q,q'} \cdot s_{q,y_{i+q}} \cdot b_{q'}^{(j)}(i+1).$$

Analog können wir für die Emissionwahrscheinlichkeiten vorgehen. Berechnen wir zuerst wieder die Wahrscheinlichkeit für eine Emissionen von $a \in \Sigma$ aus dem Zustand $q \in Q$ für eine gegebene emittierte Folge Y unter Berücksichtigung der Matrizen P bzw. S der Zustandsübergangs- und Emissionswahrscheinlichkeiten:

$$\begin{aligned}
&\mathrm{Ws}(x_i = q \mid Y, P, S) \\
&= \frac{\mathrm{Ws}(x_i = q, Y \mid P, S)}{\mathrm{Ws}(Y)} \\
&= \frac{\mathrm{Ws}((y_1, \ldots, y_i), x_i = q \mid P, S) \cdot \mathrm{Ws}((y_{i+1}, \ldots, y_\ell) \mid x_i = q, P, S)}{\mathrm{Ws}(Y)} \\
&= \frac{1}{\mathrm{Ws}(Y)} \cdot f_q(i) \cdot b_q(i+1).
\end{aligned}$$

Somit können wir die erwartete Anzahl von Emissionen des Symbols $a \in \Sigma$ aus dem Zustand $q \in Q$ für die Trainingssequenzen $\vec{Y}$ unter Berücksichtigung der alten Schätzer P bzw. S für die Zustandsübergangs- bzw. Emissionswahrscheinlichkeiten und somit mithilfe der Gleichung 8.2 bzw. 8.4 auch die zugehörigen Emissionswahrscheinlichkeiten angeben:

$$\begin{aligned}
\hat{s}_{q,a} &:= \mathbb{E}((q \to a) \mid \vec{Y}, P, S) \\
&= \sum_{j-1}^{m} \sum_{\substack{i=1 \\ y_i^{(j)}=a}}^{\ell^{(j)}} \mathrm{Ws}(x_i = q \mid Y^{(j)}, P, S) \\
&= \sum_{j=1}^{m} \frac{1}{\mathrm{Ws}(Y^{(j)})} \sum_{\substack{i=1 \\ y_i^{(j)}=a}}^{\ell^{(j)}} f_q^{(j)}(i) \cdot b_q^{(j)}(i).
\end{aligned}$$

Wir können also aus einer Schätzung der stochastischen Matrizen P und S und der der gegebenen emittierten Folgen Y (der Trainingssequenzen) neue Schätzer für P und S generieren, indem wir versuchen, die entsprechenden geschätzten Wahrscheinlichkeiten mit den Trainingssequenzen zu adjustieren. Diese Methode wird in der Literatur als *Baum-Welch-Algorithmus* bezeichnet. Die algorithmische Idee ist in Abbildung 8.5 noch einmal skizziert.

8.5.3 Erwartungswert-Maximierungs-Methode

Der Baum-Welch-Algorithmus ist ein Spezialfall einer Methode, die in der in der Literatur als *Erwartungswert-Maximierungs-Methode* oder kurz *EM-Methode* bekannt

1) Wähle Wahrscheinlichkeiten S und P;

2) Bestimme die erwartete relative Häufigkeit $p'_{q,q'}$ von Zustandsübergängen (q, q') und die erwartete relative Emissionshäufigkeit $s'_{q,a}$ unter der Annahme von P und S.

3) War die Verbesserung von (P, S) zu (P', S') eine deutliche Verbesserung so setzte $P = P'$ und $S = S'$ und gehe zu Schritt 2). Ansonsten gib P' und S' als Lösung aus.

Abbildung 8.5: Algorithmus: Skizze des Baum-Welch-Algorithmus

ist, da wir versuchen aus einem Schätzer der Wahrscheinlichkeiten und den Trainingssequenzen versuchen, den Erwartungswert der gegebenen Trainingssequenzen unter den geschätzten Wahrscheinlichkeit zu bestimmen und anschließend die Schätzer zu verbessern, so dass sie den Erwartungswert in Richtung der beobachteten Werte zu maximieren.

Wir wollen jetzt allgemein zeigen, dass die EM-Methode auch in gewisser Weise konvergiert. Es könnte ja auch sein, dass wir zwischen zwei oder mehreren Verteilungen hin- und herhüpfen und nie einen endgültigen Schätzer erhalten.

Wir wollen die Parameter P bzw. S der Zustandsübergangs- bzw. Emissionswahrscheinlichkeiten schätzen, die wir im Folgenden kurz $\Psi := (P, S)$ nennen wollen. Dazu verwenden wir wieder die Maximum-Likelihood-Methode Methode, d.h. wir versuchen Ψ so zu wählen, dass $\mathrm{Ws}(\vec{Y} \mid \Psi)$ maximal wird. Wir wissen jedoch zusätzlich, dass die Folgen $\vec{Y}$ mit Hilfe von Zustandsfolgen $\vec{X}$ im Hidden Markov Modell generiert werden. Daher wissen wir mit Hilfe des Satzes der totalen Wahrscheinlichkeit und der Definition der bedingten Wahrscheinlichkeit, dass

$$\mathrm{Ws}(\vec{Y} \mid \Psi) \;=\; \sum_{\vec{X} \in Q^{|\vec{Y}|}} \mathrm{Ws}(\vec{Y} \mid \vec{X}, \Psi) \cdot \mathrm{Ws}(\vec{X} \mid \Psi) \;=\; \sum_{\vec{X} \in Q^{|\vec{Y}|}} \mathrm{Ws}(\vec{Y}, \vec{X} \mid \Psi),$$

wobei $\vec{X} \in Q^{|\vec{Y}|}$ wieder Zustandsfolgen im gegebenen Hidden Markov Modell sind.

Für eine algorithmisch besser zugängliche Darstellung von $\mathrm{Ws}(\vec{Y} \mid X)$ bestimmen wir eine weitere Beziehung. Mit Hilfe der Definition der bedingten Wahrscheinlichkeit gilt:

$$\mathrm{Ws}(\vec{X} \mid \vec{Y}, \Psi) = \frac{\mathrm{Ws}(\vec{X}, \vec{Y} \mid \Psi)}{\mathrm{Ws}(\vec{Y} \mid \Psi)}.$$

Durch Umformung und Logarithmieren erhalten wir

$$\log(\mathrm{Ws}(\vec{Y} \mid \Psi)) = \log(\mathrm{Ws}(\vec{Y}, \vec{X} \mid \Psi)) - \log(\mathrm{Ws}(\vec{X} \mid \Psi)). \tag{8.5}$$

Sei im Folgenden Ψ_i der Parametersatz, den wir im i-ten Iterationsschritt ausgewählt haben. Wir multiplizieren jetzt beiden Seiten der Gleichung 8.5 mit $\mathrm{Ws}(\vec{X} \mid \vec{Y}, \Psi_i)$

und summieren über alle Zustandfolgen $\vec{X} \in Q^{|\vec{Y}|}$. Wir erhalten dann:

$$\begin{aligned}\log(\mathrm{Ws}(\vec{Y} \mid \Psi)) &= \underbrace{\sum_{\vec{X} \in Q^{|\vec{Y}|}} \mathrm{Ws}(\vec{X} \mid \vec{Y}, \Psi_i)}_{=1} \cdot \log(\mathrm{Ws}(\vec{Y} \mid \Psi)) \\ &= \sum_{\vec{X} \in Q^{|\vec{Y}|}} \mathrm{Ws}(\vec{X} \mid \vec{Y}, \Psi_i) \cdot \log(\mathrm{Ws}(\vec{Y}, \vec{X} \mid \Psi)) \\ &\quad - \sum_{\vec{X} \in Q^{|\vec{Y}|}} \mathrm{Ws}(\vec{X} \mid \vec{Y}, \Psi_i) \cdot \log(\mathrm{Ws}(\vec{X} \mid \vec{Y}, \Psi)).\end{aligned}$$

Anstatt nun $\mathrm{Ws}(\vec{Y} \mid \Psi)$ zu maximieren können wir aber auch $\log(\mathrm{Ws}(\vec{Y} \mid \Psi))$ maximieren, da der Logarithmus ein streng monoton wachsende Funktion ist. Somit müssen beide Maxima an derselben Stelle angenommen werden, d.h. für dieselbe Wahl von Ψ.

Wir werden jetzt iterativ vorgehen und versuchen, stattdessen ein neues Ψ so zu wählen, dass $\log(\mathrm{Ws}(\vec{Y} \mid \Psi)) > \log(\mathrm{Ws}(\vec{Y} \mid \Psi_i))$ wird und setzen dieses als Ψ_{i+1}. Damit suchen wir also nach einem Ψ mit

$$\sum_{\vec{X} \in Q^{|\vec{Y}|}} \mathrm{Ws}(\vec{X} \mid \vec{Y}, \Psi_i) \log(\mathrm{Ws}(\vec{Y}, \vec{X} \mid \Psi)) - \sum_{\vec{X} \in Q^{|\vec{Y}|}} \mathrm{Ws}(\vec{X} \mid \vec{Y}, \Psi_i) \log(\mathrm{Ws}(\vec{X} \mid \vec{Y}, \Psi))$$

$$> \sum_{\vec{X} \in Q^{|\vec{Y}|}} \mathrm{Ws}(\vec{X} \mid \vec{Y}, \Psi_i) \log(\mathrm{Ws}(\vec{Y}, \vec{X} \mid \Psi_i)) - \sum_{\vec{X} \in Q^{|\vec{Y}|}} \mathrm{Ws}(\vec{X} \mid \vec{Y}, \Psi_i) \log(\mathrm{Ws}(\vec{X} \mid \vec{Y}, \Psi_i)).$$

Dies ist äquivalent dazu, dass die folgende Ungleichung gilt:

$$\sum_{\vec{X} \in Q^{|\vec{Y}|}} \mathrm{Ws}(\vec{X} \mid \vec{Y}, \Psi_i) \log\left(\frac{\mathrm{Ws}(\vec{Y}, \vec{X} \mid \Psi)}{\mathrm{Ws}(\vec{Y}, \vec{X} \mid \Psi_i)}\right)$$

$$+ \sum_{\vec{X} \in Q^{|\vec{Y}|}} \mathrm{Ws}(\vec{X} \mid \vec{Y}, \Psi_i) \log\left(\frac{\mathrm{Ws}(\vec{X} \mid \vec{Y}, \Psi_i)}{\mathrm{Ws}(\vec{X} \mid \vec{Y}, \Psi)}\right) > 0.$$

In dieser Ungleichung ist der zweite Term auf der linken Seite die so genannte *Kullback-Leibler-Distanz* und dieses ist stets nichtnegativ (was man beispielsweise mit einigen Tricks mit Hilfe der Jensenschen Ungleichung beweisen kann). Es genügt

EM (int n, float $\Psi[]$, float ε)
{
    $\Psi_0 = \Psi$;
    int $i = 0$;
    repeat
    {
        i++;
        /* E-Schritt */
        compute $\Delta(\Psi, \Psi_{i-1})$;
        /* M-Schritt */
        $\Psi_i :\in \operatorname{argmax}\{\Delta(\Psi, \Psi_{i-1}) \,:\, \Psi\}$;
    }
    until $(\|\Psi_i - \Psi_{i-1}\| < \varepsilon)$
}

Abbildung 8.6: Skizze: Erwartungswert-Maximierungs-Methode

daher ein Ψ zu finden, so dass

$$\sum_{\vec{X} \in Q^{|\vec{Y}|}} \mathrm{Ws}(\vec{X} \mid \vec{Y}, \Psi_i) \log \left(\frac{\mathrm{Ws}(\vec{Y}, \vec{X} \mid \Psi)}{\mathrm{Ws}(\vec{Y}, \vec{X} \mid \Psi_i)} \right) > 0$$

Definieren wir

$$\Delta(\Psi, \Psi_i) := \sum_{\vec{X} \in Q^{|\vec{Y}|}} \mathrm{Ws}(\vec{X} \mid \vec{Y}, \Psi_i) \log \Big(\mathrm{Ws}(\vec{Y}, \vec{X} \mid \Psi) \Big),$$

so wollen wir als ein Ψ mit

$$\Delta(\Psi, \Psi_i) - \Delta(\Psi_i, \Psi_i) > 0.$$

finden. Da für $\Psi = \Psi_i$ der Term $\Delta(\Psi, \Psi_i) - \delta(\Psi_i, \Psi_i)$ zumindest gleich Null wird, können wir zumindest einen nichtnegativen Wert finden. Es genügt also, Ψ so zu wählen, dass der $\Delta(\Psi, \Psi_i)$ maximal wird ($\Delta(\Psi_i, \Psi_i)$ ist ja konstant), d.h. wir wählen Ψ mittels

$$\Psi_{i+1} :\in \operatorname{argmax}\{\Delta(\Psi, \Psi_i) \,:\, \Psi\}.$$

Somit wächst die Folge in jedem Schritt und da der Zuwachs beim Optimum aufhören muss, konvergiert diese Methode zumindest gegen ein lokales Maximum.

Wir bemerken jetzt noch, dass der Ausdruck $\Delta(\Psi, \Psi_i)$ der Erwartungswert der Variablen $\log(\mathrm{Ws}(\vec{Y}, \vec{X} \mid \Psi))$ unter der Verteilung $\mathrm{Ws}(\vec{X} \mid \vec{Y}, \Psi)$ ist. Somit wird auch der Name Erwartungswert-Maximierungs-Methode klar, da wir zuerst einen

Erwartungswert bestimmen und dann mit Hilfe einer Maximierung de neuen Verteilungen wählen. Das Verfahren der EM-Methode ist in Abbildung 8.6 noch einmal schematisch dargestellt. Man kann zeigen , dass der Baum-Welch-Algorithmus ein Spezialfall der EM-Methode ist und somit auch konvergiert.

8.6 Mehrfaches Sequenzen Alignment mit HMM

Wir wollen jetzt noch einmal auf das mehrfache Sequenzen Alignment aus dem vierten Kapitel zurückgreifen und versuchen dieses mit Hilfe von Hidden Markov Modellen zu lösen.

8.6.1 Profile

Zunächst einmal benötigen wir noch eine Definition, was wir unter einem Profil verstehen wollen.

Definition 8.8 *Ein* Profil $\mathcal{P}$ *der Länge* ℓ *für ein Alphabet* Σ *ist eine Matrix* $(e_i(a))_{\substack{i\in[1:\ell]\\ a\in\Sigma}}$ *von Wahrscheinlichkeiten, wobei* $e_i(a)$ *die Wahrscheinlichkeit angibt, dass an der Position* i *ein* a *auftritt.*

Ein Profil ist also eine Wahrscheinlichkeitsverteilung von Buchstaben über ℓ Positionen. Wenn wir schon ein mehrfaches Sequenz Alignment hätten, dann könnte man an den einzelnen Positionen die Verteilung der Buchstaben bestimmen, das man dann als Profil interpretieren kann.

Sei $Y \in \Sigma^\ell$, dann ist für ein Profil $\mathcal{P}$ die Wahrscheinlichkeit, dass es die Folge Y generiert hat:

$$\mathrm{Ws}(Y \mid \mathcal{P}) = \prod_{i=1}^{\ell} e_i(y_i).$$

Wir werden daher den Score eines Wortes $Y \in \Sigma^\ell$ wieder über den Logarithmus des Bruchs der Wahrscheinlichkeit dieser Zeichenreihe unter dem Profil $\mathcal{P}$ und der Wahrscheinlichkeit, wenn die Wahrscheinlichkeiten der einzelnen Zeichen unabhängig von der Position ist:

$$\mathrm{Score}(Y \mid \mathcal{P}) = \sum_{i=1}^{\ell} \log\left(\frac{e_i(y_i)}{p(y_i)}\right),$$

wobei p eine Wahrscheinlichkeitsverteilung ist, die angibt, mit welcher Wahrscheinlichkeit $p(a)$ ein Zeichen $a \in \Sigma$ (unabhängig von der Position) auftritt.

Wir konstruieren jetzt ein Hidden Markov Modell $M = (Q, \Sigma, P, S)$ für ein gegebenes Profil $\mathcal{P}$. Dazu führen wir für jede Position einen Zustand ein, sowie zwei Hilfszustände am Anfang und am Ende, die linear miteinander verbunden sind (s.a. Abbildung 8.7)

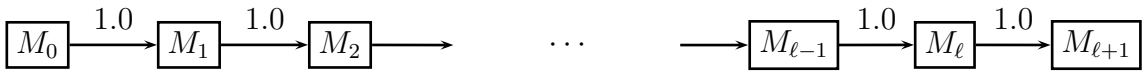


Abbildung 8.7: Skizze: HMM für ein gegebenes Profil

Für die Zustandsübergangswahrscheinlichkeiten gilt offensichtlich $p_{i,j} = 1$, wenn $j = i+1$ und 0 sonst. Die Emissionswahrscheinlichkeiten sind natürlich $s_{M_i,a} = e_i(a)$ für $i \in [1 : \ell]$. Die Zustände M_0 und $M_{\ell+1}$ widersprechen eigentlich der Definition eines Hidden Markov Modells, da sie keine Zeichen ausgeben. Daher müssen wir unser Modell noch etwas erweitern.

Definition 8.9 *Ein Zustand eines Hidden Markov Modells, der keine Zeichen emittiert heißt* stiller Zustand *(engl.* silent state*).*

Somit sind die Zustände M_0 und $M_{\ell+1}$ unseres Hidden Markov Modells für ein gegebenes Profil der Länge ℓ stille Zustände.

8.6.2 Erweiterung um InDel-Operationen

Ein Durchlauf durch das Profil-HMM entspricht einem Alignment gegen das Profil (was man als Verteilung des Konsensus-Strings ansehen kann). Was ist aber mit den InDel-Operationen, die bei Alignments essentiell sind? Wir werden jetzt das Hidden Markov Modell für solche Operationen erweitern.

Für die Einfügungen zwischen zwei Zeichen des Konsensus-Strings führen wir jetzt neue Zustände ein, die diese modellieren wollen. Dies ist in Abbildung 8.8 illustriert.

Nun sind die Zustandsübergangswahrscheinlichkeiten wieder offen. Wir werden später darauf zurückkommen, wie diese zu bestimmen sind. Die Emissionswahrscheinlichkeiten für die Insertions-Zustände sind durch $s_{I_i,a} = p(a)$ gegeben.

Betrachten wir jetzt noch eine Einfügung von k Zeichen zwischen zwei Matching-Zuständen, d.h. der Zustandsfolge $(M_i, I_i, \ldots, I_i, M_{i+1})$. Was bedeutet dies für den

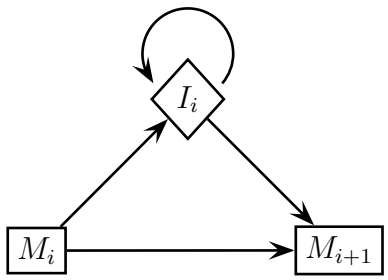


Abbildung 8.8: Skizze: Einbau von Insertionen

Score für die k Einfügungen? Der additive Teil für den Score beträgt dann:

$$\log(p_{M_i,I_i} \cdot p_{I_i,I_i}^{k-1} \cdot p_{I_i,M_{i+1}}) = \underbrace{\log(p_{M_i,I_i} \cdot p_{I_i,M_{i+1}})}_{\text{Gap}-\text{Eröffnungs}-\text{Strafe}} + \underbrace{(k-1)\log(p_{I_i,I_i})}_{\text{Gap}-\text{Verlängerungs}-\text{Strafe}} .$$

Man sieht, dass hier die Gaps gemäß einer affinen Lücken-Strafe bewertet werden. Allerdings hängt hier die Konstanten von der Position ab, d.h. sie sind nicht für alle Lücken gleich.

Des Weiteren müssen wir noch die Deletionen modellieren. Eine einfache Möglichkeit ist, weitere Möglichkeiten der Übergänge von M_i nach M_j für $i < j$ einzufügen, wie dies in Abbildung 8.9 schematisch dargestellt ist.

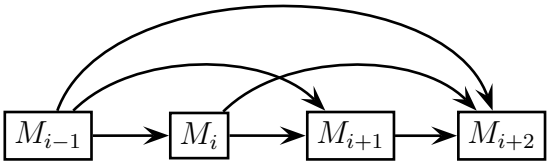


Abbildung 8.9: Skizze: Einbau von Deletionen

Wir werden jetzt noch eine andere Modellierung vorstellen, die später eine effizientere Berechnung der Vorwärts- und Rückwärtswahrscheinlichkeiten erlaubt. Hierfür ist die Anzahl der Pfeile, die in einem Zustand endet entscheidend. Im obigen Modell können dies bis zu ℓ Pfeile sein, wenn wir ein Profil der Länge ℓ zugrunde legen.

Wir werden jetzt ℓ spezielle Zustände einführen, die die Deletionen an den ℓ Positionen modellieren werden. Diese Zustände kann man als Bypässe für das entsprechende Zeichen interpretieren. Die genaue Anordnung dieser Zustände ist in Abbildung 8.10 illustriert.

Hierbei ist jetzt wichtig, dass diese Deletions-Zustände stille Zustände sind, da ja beim Löschen nichts ausgegeben werden kann. Wir wollen hier noch anmerken, dass

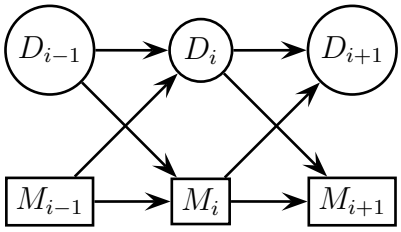


Abbildung 8.10: Skizze: Einbau von Deletions-Zuständen

wir mit Hilfe von Deletions-Zuständen nicht alle Verteilungen modellieren können, die mit Hilfe von Deletionsübergängen ohne zusätzliche Zustände möglich sind.

Auch hier wollen wir uns jetzt den resultierenden Score für eine Folge von Deletionen $(M_i, D_{i+1}, \ldots, D_{i+k}, M_{i+k+1})$ genauer betrachten:

$$\log\left(p_{M_i,D_{i+1}} \cdot \prod_{j=1}^{k-1} p_{D_{i+j},D_{i+j+1}} \cdot p_{D_{i+k},M_{i+k+1}}\right)$$
$$= \underbrace{\log(p_{M_i,D_{i+1}} \cdot p_{D_{i+k},M_{i+k+1}})}_{\text{Gap}-\text{Eröffnungs}-\text{Strafe}} + \underbrace{\sum_{j=1}^{k-1} \log(p_{D_{i+j},D_{i+j+1}})}_{\text{Gap}-\text{Verlängerungs}-\text{Strafe}} .$$

Auch hier haben wir im Wesentlichen affine Lücken-Strafen. Allerdings werden die Deletionen unterschiedich bewertet, was natürlich von den entsprechenden Wahrscheinlichkeiten abhängt.

In der Abbildung 8.11 ist jetzt ein vollständiges Hidden Markov Modell für eine Sequenz der Länge 4 angegeben. Man beachte, dass die fehlenden Pfeile für die Wahrscheinlichkeit 0 für die entsprechenden Zustandsübergangswahrscheinlichkeiten stehen. Wie wir in diesem vollständigen Beispiel eines Hidden Markov Modells für ein Profil der Länge 4 sehen, hat jeder Zustand nur maximal 3 eingehende Pfeile. Somit können wir die Vorwärts- und Rückwärtswahrscheinlichkeiten viel einfacher berechnen, da wir nur jeweils maximal vier mögliche vorherige Zustände und nicht bis zu 3ℓ verschiedenen berücksichtigen müssen.

8.6.3 Alignment gegen ein Profil-HMM

Um ein mehrfaches Sequenzen Alignment zu konstruieren, berechnen wir einen wahrscheinlichsten Pfad durch das zugehörige Hidden Markov Modell, das wir eben konstruiert haben. Der Pfad gibt uns dann für jede Sequenz die entsprechende Alignierung an.

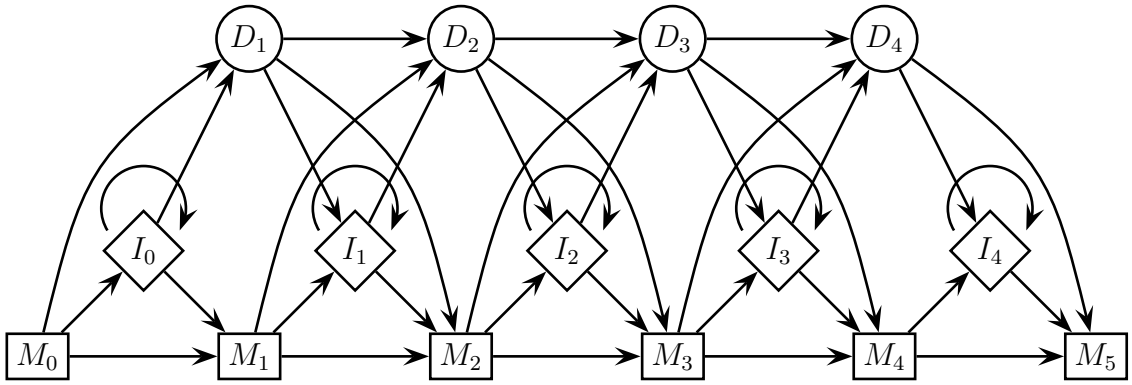


Abbildung 8.11: Skizze: Ein vollständiges HMM für $\ell = 3$

Wir merken an dieser Stelle noch an, dass wir hierfür die Zustandsübergangswahrscheinlichkeiten und die Emissionswahrscheinlichkeiten benötigen. Wir werden uns später überlegen, wie wir diese aus den gegebenen Sequenzen abschätzen können.

Für die Berechnung des wahrscheinlichsten Pfades durch unser Profil-HMM definieren wir die folgenden Scores analog zum Viterbi-Algorithmus:

$$\begin{aligned} M_j(i) &:= \text{Max. Score für } (y_1, \ldots, y_i), \text{ der in } M_j \text{ endet} \\ D_j(i) &:= \text{Max. Score für } (y_1, \ldots, y_i), \text{ der in } D_j \text{ endet} \\ I_j(i) &:= \text{Max. Score für } (y_1, \ldots, y_i), \text{ der in } I_j \text{ endet} \end{aligned}$$

Wie man sich leicht überlegt, kann man die beim Viterbi-Algorithmus verwendeten Rekursionsgleichungen für unser Profil-HMM wie folgt vereinfachen. Für die Matching-Zustände erhalten wir:

$$M_j(i+1) = \log\left(\frac{s_{M_j,y_{i+1}}}{p_{y_{i+1}}}\right) + \max\left\{\begin{array}{c} M_{j-1}(i) + \log(p_{M_{j-1},M_j}), \\ I_{j-1}(i) + \log(p_{I_{j-1},M_j}), \\ D_{j-1}(i) + \log(p_{D_{j-1},M_j}) \end{array}\right\}.$$

Für die Insertions-Zustände erhalten wir:

$$I_j(i+1) = \log\left(\frac{s_{I_j,y_{i+1}}}{p_{y_{i+1}}}\right) + \max\left\{\begin{array}{c} M_j(i) + \log(p_{M_j,I_j}), \\ I_j(i) + \log(p_{I_j,I_j}), \\ D_j(i) + \log(p_{D_j,I_j}) \end{array}\right\}.$$

Für die Deletions-Zustände erhalten wir:

$$D_j(i+1) = \max\left\{\begin{array}{c} M_{j-1}(i+1) + \log(p_{M_{j-1},D_j}), \\ I_{j-1}(i+1) + \log(p_{I_{j-1},D_j}), \\ D_{j-1}(i+1) + \log(p_{D_{j-1},D_j}) \end{array}\right\}.$$

Für die Anfangsbedingung gilt offensichtlich $M_0(0) = 1$. Alle undefinierten Werte sind als $-\infty$ zu interpretieren.

Theorem 8.10 *Sei $M = (Q, \Sigma, P, S)$ ein Hidden Markov Modell für ein mehrfaches Sequenzen Alignment der Länge ℓ. Für eine gegebene Sequenz Y lässt sich wahrscheinlichste zugehörige Zustandsfolge in Zeit $O(|Q| \cdot (\ell + |Y|)$ und mit Platz $O(|Q|)$ berechnen.*

Mit Hilfe des obigen Theorems können wir jetzt ein mehrfaches Sequenzen Alignment konstruieren. Wir berechnen zuerst für jede Folge die wahrscheinlichste zugehörige Zustandsfolge durch das Hidden Markov Modells. Anhand dieser Zustandsfolge können wir aus den Zuständen das Alignment der zugehörigen Zeichenreihe ablesen.

Als einziges Problem bleibt hierbei natürlich noch die Bestimmung der stochastischen Matrizen P und S. Diese können wir wiederum mit Hilfe der so genannten Erwartungswert-Maximierungs-Methode schätzen. Aufgrund der speziellen Gestalt des Hidden Markov Modells für mehrfache Sequenzen Alignments ergibt sich für die Vorwärts- und Rückwärtswahrscheinlichkeiten folgende einfachere Rekursionsgleichungen:

$$\begin{aligned}
f_{M_j}(i) &:= \text{Ws(Ausgabe ist } (y_1, \ldots, y_i) \text{ und Endzustand ist } M_j) \\
f_{I_j}(i) &:= \text{Ws(Ausgabe ist } (y_1, \ldots, y_i) \text{ und Endzustand ist } I_j) \\
f_{D_j}(i) &:= \text{Ws(Ausgabe ist } (y_1, \ldots, y_i) \text{ und Endzustand ist } D_j)
\end{aligned}$$

$$\begin{aligned}
b_{M_j}(i) &:= \text{Ws(Ausgabe ist } (y_{i+1}, \ldots, y_{|Y|}) \text{ und Anfangszustand ist } M_j) \\
b_{I_j}(i) &:= \text{Ws(Ausgabe ist } (y_{i+1}, \ldots, y_{|Y|}) \text{ und Anfangszustand ist } I_j) \\
b_{D_j}(i) &:= \text{Ws(Ausgabe ist } (y_{i+1}, \ldots, y_{|Y|}) \text{ und Anfangszustand ist } D_j)
\end{aligned}$$

Mit Hilfe dieser Vorwärts- und Rückwärtswahrscheinlichkeiten können wir wieder mit der Baum-Welch Methode neue, verbesserte Zustandsübergans- und Emissionswahrscheinlichkeiten aus dem bestehenden Modell berechnen. Wir überlassen die Details der Implementierung dem Leser und halten nur noch das Ergebnis fest.

Lemma 8.11 *Die Vorwärts- und Rückwärtswahrscheinlichkeiten für ein Hidden Markov Modell $M = (Q, \Sigma, P, S)$ für ein mehrfaches Sequenzen Alignment der Länge ℓ können in Zeit $O(|Q| \cdot \ell)$ berechnet werden.*

Literaturhinweise

A.1 Lehrbücher zur Vorlesung

Peter Clote, Rolf Backofen: *Introduction to Computational Biology*; John Wiley and Sons, 2000.

Richard Durbin, Sean Eddy, Anders Krogh, Graeme Mitchison: *Biological Sequence Analysis*; Cambridge University Press, 1998.

Dan Gusfield: *Algorithms on Strings, Trees, and Sequences — Computer Science and Computational Biology*; Cambridge University Press, 1997.

David W. Mount: *Bioinformatics — Sequence and Genome Analysis*, Cold Spring Harbor Laboratory Press, 2001.

Pavel A. Pevzner: *Computational Molecular Biology - An Algorithmic Approach*; MIT Press, 2000.

João Carlos Setubal, João Meidanis: *Introduction to Computational Molecular Biology*; PWS Publishing Company, 1997.

Michael S. Waterman: *Introduction to Computational Biology: Maps, Sequences, and Genomes*; Chapman and Hall, 1995.

A.2 Skripten anderer Universitäten

Bonnie Berger: *Introduction to Computational Molecular Biology*, Massachusetts Institute of Technology, http://theory.lcs.mit.edu/~bab/01-18.417-home.html;

Bonnie Berger, *Topics in Computational Molecular Biology*, Massachusetts Institute of Technology, Spring 2001, http://theory.lcs.mit.edu/~bab/01-18.418-home.html;

Paul Fischer: *Einführung in die Bioinformatik* Universität Dortmund, Lehrstuhl II, WS2001/2002, http://ls2-www.cs.uni-dortmund.de/lehre/winter200102/bioinf/

Richard Karp, Larry Ruzzo: *Algorithms in Molecular Biology*; CSE 590BI, University of Washington, Winter 1998. http://www.cs.washington.edu/education/courses/590bi/98wi/

Larry Ruzzo: *Computational Biology*, CSE 527, University of Washington, Fall 2001; http://www.cs.washington.edu/education/courses/527/01au/

Georg Schnittger: *Algorithmen der Bioinformatik*, Johann Wolfgang Goethe-Universität Frankfurt am Main, Theoretische Informatik, WS 2000/2001, http://www.thi.informatik.uni-frankfurt.de/BIO/skript2.ps.

Ron Shamir: *Algorithms in Molecular Biology* Tel Aviv University, http://www.math.tau.ac.il/~rshamir/algmb.html; http://www.math.tau.ac.il/~rshamir/algmb/01/algmb01.html.

Ron Shamir: *Analysis of Gene Expression Data, DNA Chips and Gene Networks*, Tel Aviv University, 2002; http://www.math.tau.ac.il/~rshamir/ge/02/ge02.html;

Martin Tompa: *Computational Biology*, CSE 527, University of Washington, Winter 2000. http://www.cs.washington.edu/education/courses/527/00wi/

A.3 Lehrbücher zu angrenzenden Themen

Teresa K. Attwood, David J. Parry-Smith; *Introduction to Bioinformatics*; Prentice Hall, 1999.

Maxime Crochemore, Wojciech Rytter: *Text Algorithms*; Oxford University Press: New York, Oxford, 1994.

Martin C. Golumbic: *Algorithmic Graph Theory and perfect Graphs*; Academic Press, 1980.

Benjamin Lewin: *Genes*; Oxford University Press, 2000.

Milton B. Ormerod: *Struktur und Eigenschaften chemischer Verbindungen*; Verlag Chemie, 1976.

Hooman H. Rashidi, Lukas K. Bühler: *Grundriss der Bioinformatik — Anwendungen in den Biowissenschaften und der Medizin*,

Klaus Simon: *Effiziente Algorithmen für perfekte Graphen*; Teubner, 1992.

Maxine Singer, Paul Berg: *Gene und Genome*; Spektrum Akademischer Verlag, 2000.

Lubert Stryer: *Biochemie*, Spektrum Akademischer Verlag, 4. Auflage, 1996.

A.4 Originalarbeiten

Kellogg S. Booth, George S. Lueker: Testing for the Consecutive Ones property, Interval Graphs, and Graph Planarity Using PS-Tree Algorithms; *Journal of Computer and System Science*, Vol.13, 335–379, 1976.

Ting Chen, Ming-Yang Kao: On the Informational Asymmetry Between Upper and Lower Bounds for Ultrametric Evolutionary Trees, *Proceedings of the 7th Annual European Symposium on Algorithms, ESA'99*, Lecture Notes in Computer Science 1643, 248–256, Springer-Verlag, 1999.

Richard Cole: Tight Bounds on the Complexity of the Boyer-Moore String Matching Algorithm; *SIAM Journal on Computing*, Vol. 23, No. 5, 1075–1091, 1994.
s.a. *Technical Report*, Department of Computer Science, Courant Institute for Mathematical Sciences, New York University, TR1990-512, June, 1990, http://csdocs.cs.nyu.edu/Dienst/UI/2.0/Describe/ncstrl.nyu_cs%2fTR1990-512

Martin Farach, Sampath Kannan, Tandy Warnow: A Robust Model for Finding Optimal Evolutionary Trees, *Algorithmica*, Vol. 13, 155–179, 1995.

Wen-Lian Hsu: PC-Trees vs. PQ-Trees; *Proceedings of the 7th Annual International Conference on Computing and Combinatorics, COCOON 2001*, Lecture Notes in Computer Science 2108, 207–217, Springer-Verlag, 2001.

Wen-Lian Hsu: A Simple Test for the Consecutive Ones Property; *Journal of Algorithms*, Vol.43, No.1, 1–16, 2002.

Haim Kaplan, Ron Shamir: Bounded Degree Interval Sandwich Problems; *Algorithmica*, Vol. 24, 96–104, 1999.

Edward M. McCreight: A Space-Economical Suffix Tree Construction Algorithm; *Journal of the ACM*, Vol. 23, 262–272,1976.

Moritz Maaß: *Suffix Trees and Their Applications*, Ausarbeitung von der Ferienakademie '99, Kurs 2, Bäume: Algorithmik und Kombinatorik, 1999. http://www14.in.tum.de/konferenzen/Ferienakademie99/

Esko Ukkonen: On-Line Construction of Suffix Tress, *Algorithmica*, Vol. 14, 149–260, 1995.

Index

Symbole

α-Helix, 27
α-ständiges Kohlenstoffatom, 22
β-strand, 27
π-Bindung, 6
π-Orbital, 6
σ-Bindung, 6
σ-Orbital, 5
d-Layout, 257
d-zulässiger Kern, 257
k-Clique, 256
k-Färbung, 250
p-Norm, 306
p-Orbital, 5
q-Orbital, 5
s-Orbital, 5
sp-Hybridorbital, 6
sp^2-Hybridorbital, 6
sp^3-Hybridorbital, 5
1-PAM, 153
3-Punkte-Bedingung, 270
4-Punkte-Bedingung, 291

A

additive Matrix, 282
additiver Baum, 281
    externer, 282
    kompakter, 282
Additives Approximationsproblem, 306
Additives Sandwich Problem, 306
Adenin, 16
äquivalent, 225
Äquivalenz von PQ-Bäumen, 225
aktiv, 238
aktive Region, 252
akzeptierten Mutationen, 152
Akzeptoratom, 7
Aldose, 14
Alignment
    geliftetes, 176
    konsistentes, 159
    lokales, 133
Alignment-Fehler, 172
Alignments
    semi-global, 130
All-Against-All-Problem, 145
Allel, 2
Alphabet, 43
Aminosäure, 22
Aminosäuresequenz, 26
Anfangswahrscheinlichkeit, 337
Approximationsproblem
    additives, 306
    ultrametrisches, 307, 335
asymmetrisches Kohlenstoffatom, 12
aufspannend, 294
aufspannender Graph, 294
Ausgangsgrad, 196
    maximaler, 196
    minimaler, 196

B

BAC, 36
bacterial artificial chromosome, 36
Bad-Character-Rule, 71
Basen, 16
Basen-Triplett, 31
Baum
    additiver, 281
    additiver kompakter, 282
    evolutionärer, 265
    externer additiver, 282
    kartesischer, 327
    niedriger ultrametrischer, 309
    phylogenetischer, 265, 299
    strenger ultrametrischer, 271
    ultrametrischer, 271

Baum-Welch-Algorithmus, 356
benachbart, 216
Benzol, 7
Berechnungsgraph, 262
binäre Charaktermatrix, 299
binärer Charakter, 267
Bindung
    π-Bindung, 6
    σ-Bindung, 6
    ionische, 7
    kovalente, 5
Blatt
    leeres, 226
    volles, 226
blockierter Knoten, 238
Boten-RNS, 30
Bounded Degree and Width Interval Sandwich, 256
Bounded Degree Interval Sandwich, 257
Bunemans 4-Punkte-Bedingung, 291

C

C1P, 222
cDNA, 31
cDNS, 31
Center-String, 161
Charakter, 267
    binärer, 267
    numerischer, 267
    zeichenreihiges, 267
charakterbasiertes Verfahren, 267
Charaktermatrix
    binäre, 299
Chimeric Clone, 222
chiral, 12
Chromosom, 4
cis-Isomer, 11
Clique, 256
Cliquenzahl, 256
Codon, 31
complementary DNA, 31
Consecutive Ones Property, 222
CpG-Insel, 341
CpG-Inseln, 340
Crossing-Over-Mutation, 4
cut-weight, 319
cycle cover, 196
Cytosin, 17

D

Decodierungsproblem, 345
Deletion, 102
delokalisierte π-Elektronen, 7
deoxyribonucleic acid, 14
Desoxyribonukleinsäure, 14
Desoxyribose, 16
Diagonal Runs, 148
Dipeptid, 24
Distanz eines PMSA, 176
distanzbasiertes Verfahren, 266
Distanzmatrix, 270
    phylogenetische, 303
DL-Nomenklatur, 13
DNA, 14
    complementary, 31
    genetic, 31
DNA-Microarrays, 41
DNS, 14
    genetische, 31
    komplementäre, 31
Domains, 28
dominant, 3
dominantes Gen, 3
Donatoratom, 7
Doppelhantel, 5
dynamische Programmierung, 121, 332

E

echter Intervall-Graph, 248
echter PQ-Baum, 224
Edit-Distanz, 104
Edit-Graphen, 118
Edit-Operation, 102

eigentlicher Rand, 46
Eingangsgrad, 196
    maximaler, 196
    minimaler, 196
Einheits-Intervall-Graph, 248
Elektrophorese, 38
Elterngeneration, 1
EM-Methode, 356
Emissionswahrscheinlichkeit, 342
Enantiomer, 12
Enantiomerie, 11
enantiomorph, 12
Enzym, 37
erfolgloser Vergleich, 48
erfolgreicher Vergleich, 48
erste Filialgeneration, 1
erste Tochtergeneration, 1
Erwartungswert-Maximierungs-
        Methode,
        356
Erweiterung von Kernen, 253
Euler-Tour, 330
eulerscher Graph, 214
eulerscher Pfad, 214
evolutionärer Baum, 265
Exon, 31
expliziter Knoten, 86
Extended-Bad-Character-Rule, 72
externer additiver Baum, 282

F

Färbung, 250
    zulässige, 250
False Negatives, 222
False Positives, 222
Filialgeneration, 1
    erste, 1
    zweite, 1
Fingerabdruck, 75
fingerprint, 75
Fischer-Projektion, 12
Fragmente, 220
freier Knoten, 238
Frontier, 225
funktionelle Gruppe, 11
Furan, 15
Furanose, 15

G

Geburtstagsparadoxon, 99
gedächtnislos, 338
geliftetes Alignment, 176
Gen, 2, 4
    dominant, 3
    rezessiv, 3
Gene-Chips, 41
genetic DNA, 31
genetic map, 219
genetische DNS, 31
genetische Karte, 219
Genom, 4
genomische Karte, 219
genomische Kartierung, 219
Genotyp, 3
gespiegelte Zeichenreihe, 124
Gewicht eines Spannbaumes, 294
Good-Suffix-Rule, 61
Grad, 195, 196, 261
Graph
    aufspanneder, 294
    eulerscher, 214
    hamiltonscher, 194
Guanin, 16

H

Halb-Acetal, 15
hamiltonscher Graph, 194
hamiltonscher Kreis, 194
hamiltonscher Pfad, 194
heterozygot, 2
Hexose, 14
Hidden Markov Modell, 342
HMM, 342
homozygot, 2
Horner-Schema, 74

Hot Spots, 148
hydrophil, 10
hydrophob, 10
hydrophobe Kraft, 10

I

ICG, 250
impliziter Knoten, 86
Indel-Operation, 102
induzierte Metrik, 274
induzierte Ultrametrik, 274
initialer Vergleich, 66
Insertion, 102
intermediär, 2
interval graph, 247
    proper, 248
    unit, 248
Interval Sandwich, 249
Intervalizing Colored Graphs, 250
Intervall-Darstellung, 247
Intervall-Graph, 247
    echter, 248
    Einheits-echter, 248
Intron, 31
ionische Bindung, 7
IS, 249
isolierter Knoten, 195

K

kanonische Referenz, 87
Karte
    genetische, 219
    genomische, 219
kartesischer Baum, 327
Kern, 252
    d-zulässiger, 257
    zulässiger, 252, 257
Kern-Paar, 261
Keto-Enol-Tautomerie, 13
Ketose, 15
Knoten
    aktiver, 238
    blockierter, 238
    freier, 238
    leerer, 226
    partieller, 226
    voller, 226
Kohlenhydrate, 14
Kohlenstoffatom
    α-ständiges, 22
    asymmetrisches, 12
    zentrales, 22
Kollisionen, 99
kompakte Darstellung, 272
kompakter additiver Baum, 282
komplementäre DNS, 31
komplementäres Palindrom, 38
Komplementarität, 18
Konformation, 28
konkav, 142
Konsensus-Fehler, 168
Konsensus-MSA, 172
Konsensus-String, 171
Konsensus-Zeichen, 171
konsistentes Alignment, 159
Kosten, 314
Kosten der Edit-Operationen s, 104
Kostenfunktion, 153
kovalente Bindung, 5
Kreis
    hamiltonscher, 194
Kullback-Leibler-Distanz, 358
kurzer Shift, 68

L

Länge, 43
langer Shift, 68
Layout, 252, 257
    d, 257
least common ancestor, 271
leer, 226
leerer Knoten, 226
leerer Teilbaum, 226
leeres Blatt, 226
Leerzeichen, 102

link-edge, 319
linksdrehend, 13
logarithmische
Rückwärtswahrscheinlichkeit, 350
logarithmische
Vorwärtswahrscheinlichkeit, 350
lokales Alignment, 133

M

map
genetic, 219
physical, 219
Markov-Eigenschaft, 338
Markov-Kette, 337
Markov-Ketten
k-ter Ordnung, 338
Markov-Ketten k-ter Ordnung, 338
Match, 102
Matching, 198
perfektes, 198
Matrix
additive, 282
stochastische, 337
mature messenger RNA, 31
Maxam-Gilbert-Methode, 39
maximaler Ausgangsgrad, 196
maximaler Eingangsgrad, 196
Maximalgrad, 195, 196
Maximum-Likelihood-Methode, 357
Maximum-Likelihood-Prinzip, 150
mehrfaches Sequenzen Alignment (MSA), 155
Mendelsche Gesetze, 4
messenger RNA, 30
Metrik, 104, 269
induzierte, 274
minimaler Ausgangsgrad, 196
minimaler Eingangsgrad, 196
minimaler Spannbaum, 294
Minimalgrad, 195, 196
minimum spanning tree, 294
mischerbig, 2
Mismatch, 44
Monge-Bedingung, 201
Monge-Ungleichung, 201
Motifs, 28
mRNA, 30
Mutation
akzeptierte, 152
Mutationsmodell, 151

N

Nachbarschaft, 195
Nested Sequencing, 41
nichtbindendes Orbital, 9
niedriger ultrametrischer Baum, 309
niedrigste gemeinsame Vorfahr, 271
Norm, 306
Norm eines PQ-Baumes, 245
Nukleosid, 18
Nukleotid, 18
numerischer Charakter, 267

O

offene Referenz, 87
Okazaki-Fragmente, 30
Oligo-Graph, 215
Oligos, 213
One-Against-All-Problem, 143
optimaler Steiner-String, 168
Orbital, 5
π-, 6
σ-, 5
p, 5
q-, 5
s, 5
sp, 6
sp^2, 6
sp^3-hybridisiert, 5
nichtbindendes, 9
Overlap, 190
Overlap-Graph, 197

P

P-Knoten, 223
PAC, 36
Palindrom
    komplementäres, 38
Parentalgeneration, 1
partiell, 226
partieller Knoten, 226
partieller Teilbaum, 226
Patricia-Trie, 85
PCR, 36
Pentose, 14
Peptidbindung, 23
Percent Accepted Mutations, 153
perfekte Phylogenie, 299
perfektes Matching, 198
Periode, 204
Pfad
    eulerscher, 214
    hamiltonscher, 194
Phänotyp, 3
phylogenetische Distanzmatrix, 303
phylogenetischer Baum, 265, 299
phylogenetisches mehrfaches
        Sequenzen Alignment, 175
Phylogenie
    perfekte, 299
physical map, 219
physical mapping, 219
PIC, 249
PIS, 249
plasmid artificial chromosome, 36
Point Accepted Mutations, 153
polymerase chain reaction, 36
Polymerasekettenreaktion, 36
Polypeptid, 24
Posteriori-Decodierung, 347
PQ-Bäume
    universeller, 234
PQ-Baum, 223
    Äquivalenz, 225
    echter, 224
    Norm, 245
Präfix, 43, 190
Präfix-Graph, 193
Primärstruktur, 26
Primer, 36
Primer Walking, 40
Profil, 360
Promotoren, 34
Proper Interval Completion, 249
proper interval graph, 248
Proper Interval Selection (PIS), 249
Protein, 22, 24, 26
Proteinbiosynthese, 31
Proteinstruktur, 26
Pyran, 15
Pyranose, 15

Q

Q-Knoten, 223
Quartärstruktur, 29

R

Ramachandran-Plot, 26
Rand, 46, 252
    eigentlicher, 46
Range Minimum Query, 330
rechtsdrehend, 13
reduzierter Teilbaum, 226
Referenz, 87
    kanonische, 87
    offene, 87
reife Boten-RNS, 31
reinerbig, 2
relevanter reduzierter Teilbaum, 237
Replikationsgabel, 29
Restriktion, 225
rezessiv, 3
rezessives Gen, 3
ribonucleic acid, 14
Ribonukleinsäure, 14
Ribose, 16
ribosomal RNA, 31
ribosomaler RNS, 31

RNA, 14
mature messenger, 31
messenger, 30
ribosomal, 31
transfer, 33
RNS, 14
Boten-, 30
reife Boten, 31
ribosomal, 31
Transfer-, 33
rRNA, 31
rRNS, 31
RS-Nomenklatur, 13
Rückwärts-Algorithmus, 349
Rückwärtswahrscheinlichkeit, 348
logarithmische, 350

S

säureamidartige Bindung, 23
Sandwich Problem
additives, 306
ultrametrisches, 306
Sanger-Methode, 39
SBH, 41
Sektor, 238
semi-globaler Alignments, 130
separabel, 318
Sequence Pair, 150
Sequence Tagged Sites, 220
Sequenzieren durch Hybridisierung, 41
Sequenzierung, 38
Shift, 46
kurzer, 68
langer, 68
sicherer, 46, 62
zulässiger, 62
Shortest Superstring Problem, 189
sicherer Shift, 62
Sicherer Shift, 46
silent state, 361
solide, 216
Spannbaum, 294
Gewicht, 294
minimaler, 294
Spleißen, 31
Splicing, 31
SSP, 189
state
silent, 361
Steiner-String
optimaler, 168
Stereochemie, 11
stiller Zustand, 361
stochastische Matrix, 337
stochastischer Vektor, 337
strenger ultrametrischer Baum, 271
Strong-Good-Suffix-Rule, 61
STS, 220
Substitution, 102
Suffix, 43
Suffix-Bäume, 85
Suffix-Link, 82
suffix-trees, 85
Suffix-Trie, 80
Sum-of-Pairs-Funktion, 156
Supersekundärstruktur, 28

T

Tautomerien, 13
teilbaum
partieller, 226
Teilbaum
leerer, 226
reduzierter, 226
relevanter reduzierter, 237
voller, 226
Teilwort, 43
Tertiärstruktur, 28
Thymin, 17
Tochtergeneration, 1
erste, 1
zweite, 1
Trainingssequenz, 353

trans-Isomer, 11
transfer RNA, 33
Transfer-RNS, 33
Translation, 31
Traveling Salesperson Problem, 195
Trie, 79, 80
tRNA, 33
tRNS, 33
TSP, 195

U

Ultrametrik, 269
    induzierte, 274
ultrametrische Dreiecksungleichung, 269
ultrametrischer Baum, 271
    niedriger, 309
Ultrametrisches Approximationsproblem, 307, 335
Ultrametrisches Sandwich Problem, 306
Union-Find-Datenstruktur, 323
unit interval graph, 248
universeller PQ-Baum, 234
Uracil, 17

V

Van der Waals-Anziehung, 9
Van der Waals-Kräfte, 9
Vektor
    stochastischer, 337
Verfahren
    charakterbasiertes, 267
    distanzbasiertes, 266
Vergleich
    erfolgloser, 48
    erfolgreichre, 48
    initialer, 66
    wiederholter, 66
Viterbi-Algorithmus, 346
voll, 226
voller Knoten, 226
voller Teilbaum, 226
volles Blatt, 226
Vorwärts-Algorithmus, 349
Vorwärtswahrscheinlichkeit, 348
    logarithmische, 350

W

Waise, 216
Wasserstoffbrücken, 8
Weak-Good-Suffix-Rule, 61
wiederholter Vergleich, 66
Wort, 43

Y

YAC, 36
yeast artificial chromosomes, 36

Z

Zeichenreihe
    gespiegelte, 124
    reversierte, 124
zeichenreihige Charakter, 267
zentrales Dogma, 34
zentrales Kohlenstoffatom, 12, 22
Zufallsmodell R, 151
zugehöriger gewichteter Graph, 295
zulässig, 257
zulässige Färbung, 250
zulässiger Kern, 252
zulässiger Shift, 62
Zustand
    stiller, 361
Zustandsübergangswahrscheinlichkeit, 337
zweite Filialgeneration, 1
zweite Tochtergeneration, 1
Zyklenüberdeckung, 196