**UNIVERSITÄT DUISBURG-ESSEN**
FAKULTÄT FÜR INGENIEURWISSENSCHAFTEN
ABTEILUNG INFORMATIK UND ANGEWANDTE KOGNITIONSWISSENSCHAFT

**Bachelorarbeit**

# Fancy thesis template

Vorname Nachname
Matrikelnummer: 0123456
Angewandte Informatik (Bachelor)

UNIVERSITÄT
DUISBURG
ESSEN

Fachgebiet Verteilte Systeme, Abteilung Informatik
Fakultät für Ingenieurwissenschaften
Universität Duisburg-Essen

28. Februar 2023

**Erstgutachter:** Prof. Dr-Ing. Torben Weis
**Zweitgutachter:** Hier Zweitgutachter eintragen
**Zeitraum:** 1. September 2042 - 1. Januar 2043

# Abstract

An abstract is a brief summary of a research article, thesis, review, conference proceeding, or any in-depth analysis of a particular subject and is often used to help the reader quickly ascertain the paper's purpose. When used, an abstract always appears at the beginning of a manuscript or typescript, acting as the point-of-entry for any given academic paper or patent application. Abstracting and indexing services for various academic disciplines are aimed at compiling a body of literature for that particular subject. [1]

[1] Wikipedia: `https://en.wikipedia.org/wiki/Abstract_(summary)`

# Contents

# Chapter 1

# Grundlagen

## 1.1 Grundbegriffe der Stochastik

### 1.1.1 Zufallsexperimente und Elementarereignisse

Der Grundbaustein der Stochastik sind sogenannte **Zufallsexperimente**. Klassische Beispiele für Zufallsexperimente sind das werfen eines fairen 6-seitigen Wüfels oder das ziehen von Kugeln aus einer Urne. Das Ergebnis eines Zufallsexperimentes nennt man **Elementarereignis** $w$. Die Menge aller Möglichen Elementarereignisse ist $\Omega$. Wenn unser Zufallsexperiment das einmalige Werfen eines fairen Würfels ist, dann gilt $\Omega = \{1, 2, 3, 4, 5, 6\}$ und für das einmalige Werfen zweier fairer Würfel gilt $\Omega = \{1, 2, 3, 4, 5, 6\} \times \{1, 2, 3, 4, 5, 6\}$.

### 1.1.2 Zufallsereignisse

Ein **Ereignis** $E$ ist eine Teilmenge von $\Omega$. Zum Beispiel ist $E = \{6\}$ das Ereigniss, dass die Zahl 6 gewürfelt wird und $E = \{2, 4, 6\}$ das Ereignis, dass eine gerade Zahl fällt. Die Menge aller möglichen Ereignisse auf einer Menge von Elementarereignissen ist die **Ereignisalgebra** $\mathcal{E}$. Sie entspricht der Potenzmenge der Elementarereignisse $\mathcal{E} = \mathcal{P}(\Omega)$.

### 1.1.3 Wahrscheinlichkeiten

Die Wahrscheinlichkeit eines Ereignisses $E \in \mathcal{E}$ ist gegeben durch ein **Wahrscheinlichkeitsmaß** $P : \mathcal{E} \to [0, 1]$. Zufallsereignisse können durch **Mengenoperationen** kombiniert werden. So ist die Wahrscheilichkeit für das gemeinsame Eintreten von $A$ und $B$ gegeben durch die Vereinigung der Mengen $P(A \cap B)$ und die Wahrscheinlichkeit

für das Eintreten von $A$ oder $B$ gegeben durch die Vereinigung $P(A \cup B)$. Zwei Zufallsereignisse $A$ und $B$ sind voneinander **unabhängig** wenn gilt

$$P(A \cap B) = P(A) \cdot P(B) \tag{1.1}$$

Für Zufallsereignisse die nicht unabhängig sind können wir die **bedingte Wahrscheinlichket** berechnen. Die Wahrscheinlichkeit von $A$ abhängig von $B$ wird notiert mit $P(A \mid B)$ und ist definiert als:

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)} \tag{1.2}$$

Die Wahrscheinlichkeit von $A$ lässt sich nur berechnen falls die Wahrscheinlichkeit von $B$ **wohldefiniert** ist, also $P(B) > 0$.

### 1.1.4 Wahrscheinlichkeitsräume

Eine Menge an Elementarereignissen $\Omega$, die auf $\Omega$ definierte Ereignisalgebra $\mathcal{E}$ und das Wahrscheinlichkeitsmaß $P : \mathcal{E} \to [0, 1]$ bilden zusammen einen **Wahrscheinlichkeitsraum** $(\Omega, \mathcal{E}, P)$. Im folgenden beschränken wir uns auf diskrete Wahrscheinlichkeitsräume, welche dadurch gekennzeichnet sind, dass $\Omega$ abzählbar endlich ist [1].

### 1.1.5 Zufallsvariablen

Eine **Zufallsvariable** $X$ ist eine Abbildung von $\Omega$ in eine beliebige Menge $C$ [1]. Sei zum Beispiel $\Omega$ die Menge aller möglichen Ergebnisse des Werfens zweier fairer Würfel. Dann ist die Augensumme der Würfel gegeben durch die Abbildung $X : \Omega \to [0, 1]$ mit $X(i, j) := i + j$ für alle $(i, j) = w \in \Omega$. Die Wahrscheinlichkeit, dass die Augensumme 3 ergibt lässt sich dann berechnen mit

$$P(X = 3) = P(\{(1, 2), (2, 1)\}) = P((1, 2)) + P((2, 1)) = \frac{1}{36} + \frac{1}{36} = \frac{1}{18}$$

Analog zu Zufallsereignissen sind Zufallsvariablen unabhängig wenn gilt

$$P(X_1 = x_1 \mid X_2 = x_2, \dots X_n = x_n) = P(X_1 = x_1) \cdot P(X_2 = x_2) \cdot \dots \cdot P(X_n = x_n) \tag{1.3}$$

### 1.1.6 Stochastische Prozesse

Ein Stochastischer Prozess beschreibt ein System welches sich zu einem gegebenen Zeitpunk $t$ in einem von endlich vielen Zuständen $S = \{s_1, s_2, \dots, s_n\}$ befinden kann, wobei das Verhalten des Systems durch Zufallsereignisse bestimmt wird. Formaler ausgedrückt ist ein stochastischer Prozess ist eine Menge an Zufallsvariablen $\{X_t\}_{t \in T}$, indiziert durch $T$ [11]. Die Zufallsvariablen sind eine Abbildung in den Zustandsraum $S$ [3].$X_t -> S = \{s_1, s_2, \dots, s_n\}$

## 1.2 Markovprozesse und Markov Modelle

### 1.2.1 Markovketten

Eine Markovkette ist ein stochastischer Prozess in welchem der nächste Zustand einzig vom gegenwärtigen Zustand abhängt. Diese Eigenschaft ist bekannt als Markov-Eigneschaft [11].

**Definition 1 (Markovkette)** *Ein Stochastischer Prozess $\{X_t\}_{t\in T}$, welcher Werte aus einer Menge an Zuständen $S = \{s_1, s_2, \ldots, s_n\}$ annehmen kann ist ein Markovprozess, falls gilt*

$$P(X_{n+1} = s_{n+1} \mid X_n = i_n, \ldots, X_2 = i_2, X_1 = i_1) = P(X_{n+1} = i_{n+1} \mid X_n = i_n) \quad (1.4)$$

*Für alle $n \in N_0$ und alle $i_k \in S$ unter der Vorraussetzung, dass alle bedingten Wahrscheinlichkeiten Wohldefiniert sind, also $P(X_n = i_n, \ldots, X_2 = i_2, X_1 = i_1) > 0$.*

Für solch eine Markovkette definieren wir nun die zwei folgenden Variablen.

**Definition 2 (Startwahrscheinlichkeitsvektor** *pi***)** *$\pi$ ist ein Vektor mit Länge $n = |S|$. $\pi_i$ gibt die Wahrscheinlichkeit an, in Zustand $s_i$ zu starten. Also die Wahrscheinlichkeit, dass sich der Prozess in Zeitpunkt $t = 0$ in Zustand $s_i$ befindet.*

$$\pi_i = P(X_0 = s_i) \quad (1.5)$$

**Definition 3 (Transitionsmatrix** *A***)** *Die Transitionsmatrix A beschreibt die bedingten Wahrscheinlichkeiten, mit denen ein Zustandsübergang geschieht. $a_i(j)$ ist die Wahrscheinlichkeit dass sich der Prozess in Zeitpunkt $t+1$ in Zustand $s_j$ befindet, unter der Bedingung das er sich in Zeitpunk $t$ in Zustand $s_i$ befindet.*

$$a_{ij} = P(X_{t+1} = s_j \mid X_t = s_i) \quad (1.6)$$

Eine Markovkette ist **zeithomogen** wenn die Transitionswahrscheinlichkeiten $a_{ij}$ unabhängig von der Zeit $t$ sind [4].

$$a_{ij} = P(X_{t+1} = s_j \mid X_t = s_i) = P(X_t = s_j \mid X_{t-1} = s_i) \quad (1.7)$$

In dieser Arbeit werden ausschließlich zeithomogene Markovketten betrachtet.

Das Ereignis in irgendeinem Zustand zu starten, so wie das Ereignis von einem gegebenen Zustand in irgendeinen anderen Zustand zu wechseln sind sichere Ereignisse. Somit gelten die Bedingungen, dass die Werte des Startwahrscheinlichkeitsvektors $\pi$ und die

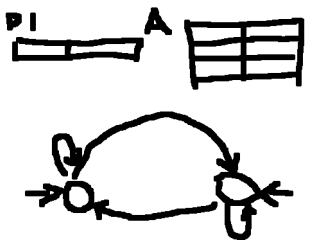
Werte der ausgehenden Transitionen für jeden Zustand $A_i$ in Summe jeweils 1 ergeben müssen. Diese Bedingung nennt man **Reihenstochastizität**

$$\sum_{i=0}^{N} \pi_i = 1 \qquad\qquad \sum_{j=0}^{N} a_{ij} = 1 \tag{1.8}$$

Als Beispiel für eine zeitdiskrete homogene Markovkette werden wir die durchschnittliche jährliche Temperatur modellieren. Seien die Zustände des Modells gegeben durch $S = \{H, K\}$, wobei $H$ für heiß und $K$ für kalt steht. Der Startvektor $\pi$ und die Transitionsmatrix $A$ seien gegeben durch

$$\pi = (0.7, 0.3) \qquad\qquad A = \begin{bmatrix} 0.8 & 0.2 \\ 0.4 & 0.6 \end{bmatrix} \tag{1.9}$$

Wir können diese Markovkette anschaulich als gerichteten Transitionsgraphen representieren.




Mit unserem Temperaturmodell können wir nun zum Beispiel berechnen, was die Wahrscheinlichkeit ist die Temperaturabfolge $O = \{H, K, K, H, H\}$ zu beobachten.

$$\begin{aligned} P(O) &= P(O_1 = H) \cdot P(O_2 = K \mid O_1 = H) \cdot P(O_3 = K \mid O_2 = K) \\ &\quad \cdot P(O_4 = H \mid O_3 = K) \cdot P(O_5 = H \mid O_4 = H) \\ &= \pi_1 \cdot a_{12} \cdot a_{22} \cdot a_{21} \cdot a_{11} \\ &= 0.7 \cdot 0.2 \cdot 0.6 \cdot 0.4 \cdot 0.8 = 0.02688 \end{aligned}$$

Eine Markovkette gibt uns also die möglichkeit probabilistische Systeme, in denen die Zustände zu den Observationen korrespondieren zu modellieren.

### 1.2.2 Hidden Markov Modelle

Bei einem Hidden Markov Modell ist die Observationssequenz nicht identisch zur Zustandssequenz, sondern die Observationssymbole sind das Ergebnis einer stochastischen Funktion der Zustände. Die Zustände selbst können wir nicht beobachten denn sie sind

"hidden". Ein HMM besteht also aus zwei stochastischen Prozessen. Ein versteckter Prozess, der die Transitionen in Abhängigkeit des vorherigen Zustandes bestimmt und ein zweiter Prozess, der die Observationssymbole in Abhängigkeit des gegenwärtigen Zustandes bestimmt. Dieser zweite Prozess wird beschrieben durch eine Emissionsmatrix $B$.

**Definition 4 (Emissionsmatrix $B = \{b_j(k)\}$)** *Die Emissionsmatrix $B$ gibt die Wahrscheinlichkeit eines Observationssymbols $v_k$ unter der Bedingung, dass sich der Prozess im gegenwärtigen Zeitpunkt $t$ in Zustand $s_j$ befindet an.*

$$b_j(k) = P(O_t = v_k \mid q_t = S_j) \qquad 1 \leq j \leq N \qquad 1 \leq k \leq M \tag{1.10}$$

Wie ein solches HMM nun eine Observationssequenz $O$ von Länge $T$ erzeugen kann möchte ich durch folgendes Gedankenexperiment verdeutlichen.

Stellen wir uns einen Raum vor, in dem $N$ Urnen stehen und in welchem sich eine Person befindet. Wir können in diesen Raum nicht hineinsehen. Die Person wird nun insgesamt $T$ Kugeln aus den Urnen ziehen und zurücklegen, wobei sie sich abhängig von einem Zufallsprozess von einer Urne zur nächsten bewegt. Um ethische Bedenken über dieses Gedankenexperiment zu minimieren sei angemerkt, dass der Raum klimatisiert ist und der Person Kakao und Kekse zur Verfügung stehen.

- Zunächst wird die initiale Urne anhand des Startwahrscheinlichkeitsvektor $\pi$ ausgewählt.
- Dann zieht die Person eine Kugel aus der Urne, notiert deren Farbe als $O_1$ und legt die Kugel zurück.
- Danach wird eine neue Urne $q_2 = S_j$ mit Wahrscheinlichkeit $a_{ij} = P(q_2 = S_j \mid q_1 = S_1)$ gewählt. Aus dieser neuen Urne wird die nächste Kugel $O_2$ der Farbe $v_k$ mit Wahrscheinlichkeit $b_j(k)$ gezogen.

Diese Prozedur wird so oft wiederholt bis der Zeitpunkt $t = T$ erreicht ist. Wir erhalten somit eine Observationssequenz von Farben $O = \{O_1, O_2, \ldots, O_T\}$. Abbildung 1.1 zeigt eine Visualisierung dieses Gedankenexperiments für $N = 3$

#### 1.2.2.1 Die drei Probleme

Ähnlich wie im Namen der kultigen Hörspielserie über drei junior-Investigatoren aus der kalifornischen Kleinstadt Rocky Beach, präsentieren sich auch im Zusammenhang mit Hidden Markov Modellen drei Fragezeichen. Diese sind allerdings nicht auf einer Visitenkarte anzusiedeln, sondern befinden sich am respektiven Ende folgender Fragen.

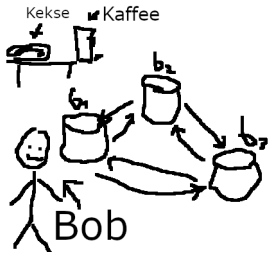



Figure 1.1: Urn Ball Model

- Wie berechnet man die Wahrscheinlichkeit einer Observationssequenz? - Wie berechnet man die Wahrscheinlichste Zustandssequenz? - Wie maximiert man die Wahrscheinlichkeit einer Observationssequenz?

#### 1.2.2.2 Berechnen der Wahrscheinlichkeit einer Observationssequenz

Zum Beispiel Was ist die Wahrscheinlichkeit 5 aufeinanderfolgende Jahre mit dicken Baumringen zu beobachten? Formell formuliert $P(L, L, L, L, L \mid \lambda)$. Wenn wir die Abfolge der Zustände $Q = q_1, q_2, q_3, q_4, q_5$ bereits kennen lässt sich $P(O \mid \lambda)$ sehr einfach berechnen. Die Wahrscheinlichkeit ein Symbol $k$ zu beobachten unter der Bedingung in Zustand $S_i$ zu sein ist nichts anderes als $B_{i,k}$ Die Wahrscheilichkeit einer Observationssequenz für ein gegebenes Modell und eine gegebene Abfolge von Zuständen ist also $P(O \mid Q, \lambda) = \prod_{t=1}^{T} b_{q_t}(O_t)$ Wenn wir die Abfolge der Zustände jedoch nicht kennen gestaltet sich die Berechnung etwas komplizierter. Die Wahrscheinlichkeit von $P(O \mid \lambda)$ ist equivalent zu der Summe der bedingten Wahrscheinlichkeiten aller möglichen Zustandsfolgen. $P(O \mid \lambda) = \sum_{allQ} P(O \mid Q, \lambda)$ Es ist möglich $P(O \mid \lambda)$ so zu berechnen, jedoch wächst die Anzahl der Möglichen Zustandsabfolgen $Q$ exponentiell in Abhängigkeit von der länge der Observationssequenz $T$. Präzise gesagt gilt $|Q| = 2T \cdot N^T$. Selbst wenn N und T kleine Werte einnehmen ist die benötigte Rechenleistung untragbar. Für N=5 und T=100 müssten bereits $2 \cdot 100 \cdot 5^{100} \approx 10^{72}$ Berechnungen durchgeführt werden. Um diese Zahl in Relation zu stellen, $10^{72}$ ist mindestens 3 mal mehr als 1000 und 1000 ist schon ziemlich groß. Zum Glück gibt es eine effiziente Methode um $P(O \mid \lambda)$ zu berechnen, die Forwärtsvariable.

#### Die Forwärtsvariable

Sei alpha die Forwärtsvariable folgendermaßen definiert $\alpha_t(i) = P(O_1 O_2 \ldots O_t \wedge q_t = S_i \mid \lambda)$ $\alpha_t(i)$ Beschreibt also die Wahrscheinlichkeit die partielle Observationssequenz $O_1 O_2 \ldots O_t$ zu beobachten und in Zeitpunkt $t$ in in Zustand $i$ zu sein. $\alpha_t(i)$ kann folgendermaßen induktiv berechnet werden Für $t = 0$ lässt sich $alpha_t(i)$ aus $\pi$ und $B$

berechnen, da noch keine Transition stattgefunden hat. $\alpha_0(i) = \pi_i \cdot b_i(O_0)$ Für $t > 0$ gilt $\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) \cdot a_{i,j}\right] \cdot b_j(O_{t+1})$ Der Casus Knacksus, welcher eine effiziente Berechung von $\alpha$ ermöglicht ist die Markov-Eigenschaft. Diese besagt, dass der Zustand in Zeitpunkt $t + 1$ nur vom Zustand in Zeitpunkt $t$ abhängt. Für jeden Zeitpunkt und jeden Zustand gibt es also genau $N$ Zustände in denen sich der Prozess zuvor befunden haben kann. Da der Rechenaufwand für jeden Zeitpunkt gleich ist hängt die Komplexität von *alpha* mit $N^2 \cdot T$ nur noch linear von $T$ ab und nicht exponentiell wie in dem vorherigen Ansatz.

## 1.3 Metaheuristische Algorithmen

Etablierte Mopeden local Search, Tabu Search, Simulated Annealing, Evolution Strategies, Genetic Algorithms, Ant Colony Optimization and Particle Swarm Optimization

Metaheuristische Algorithmen werden angewendet auf "Harte" Optimierungsprobleme. In diese Kategorie fallen unter anderem Probleme für die kein Algorithmus bekannt ist, der ein globales Optimum in endlicher Zeit findet. [8] Das bestimmen optimaler Parameter eines Hidden Markov Models ist demnach ein hartes Problem.

Heuristischer Algorithmus ist salopp gesagt schlaues Raten.

Vorteile Gegenüber klassischen iterativen Algorithmen - - Metaheuristische Optimierungsverfahren können vielverpsrechend sein, wenn wir keine hochoptimierte Lösung suchen, sondern eine Lösung welche "gut genug" ist, aber einfach zu berechnen ist. [9]

- Eine Metaheuristik ist nicht ein bestimmter Algorithmus sondern vielmehr eine Ansammlung von Ideen, Konzepten und Operatoren, welche verwendet werden können um einen heuristischen Algorithmus zu erstellen. [7]

Eine Metaheuristik bietet ein Problemunabhängiges Framework, aus welchem man sich einen Heuristischen Algorithmus für ein spezifisches Problem bastelt. So muss man nicht ständig das Rad neu erfinden.

## 1.4 Funktionsweise

- heuristische Algorithmen lassen eine verschlechterung einer Lösung zu in der Hoffnung dadurch lokalen Minima zu entkommen. [8]

Metaheuristische Algorithmen in letzter zeit am stabbo

Ein Metaheuristischer Algorithmus ist inspiriert durch die dynamiken in einem Biologischen, Physikalischen oder auch Sozialen System.

Bekannte Beipsiele sind Particle Swarm Optimization, Genetic Algorithm und

Key unterschied ob es sich um einen Single-solution oder Population based handelt

Key komponente ist Exploration und Exploitation Manchmal auch intensivication und diversification gennant [missing quote]

Exploitation beschreibt das verbessern einer gegebenen Lösung Exploration beschreibt das finden neuer Lösungen

Es ist wichtig eine gute Balance zu finden, denn wenn die Nur Exploitation ohne Exploration ist equivalent zu Lokaler Suche

Nur Exploration ohne Exploitation ist equivalent zu random Search

### 1.4.1 Kritik an Metaheuristischen Algorithmen

In den letzten Jahren wurde das Feld der Optimierung regelrecht überschwemmt mit "neuen" Metapher basierten Algorithmen. Ob Mikrofledermaus, Jazz-Musiker, Schwarze Löcher oder auch intelligente Wassertropfen. Jedes erdenkliche Konzept kann in einen Optimierungsalgorithmus verwandelt werden. Oft stellt sich jedoch heraus, dass das einzige was diese "neuen" Algorithmen zum Feld beitragen eine Umbenennung eines bereits etablierten Algorithmus ist. [2] So ist zum Beispiel Harmony search, ein Suchalgorithmus der auf dem Prinzip von Jazz-Musikern funktioniert nichts weiteres als eine Umbenennung eines speziellen Falles des Genetischen Algorithmus. [10] Trotz mangelnder Innovation verzeichnet eine Suche nach "harmony search" auf Google Scholar laut Weyland im Jahre 2010 586 Einträge. Im Jahre 2023 ist diese Zahl auf stolze 57.500 Einträge gestiegen, wovon 7.840 Einträge nach 2022 erschienen. Die Flut solcher vermeintlich "neuen" Algorithmen ist sehr nachteilig für das Feld der Optimierung, denn die elaborierten Metaphern für bereits existierende Konzepte führen zu Verwirrung und tatsächlich innovative Ansätze werden übersehen. [7]

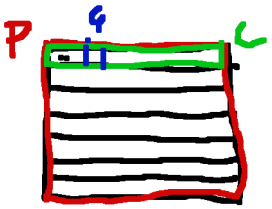
## 1.5 Genetische Algorithmen

Genetische Algorithmen nehmen Inspiration von der Evolutionstheorie von Charles Darwin Dass angepasstere Individuen erfolgreicher darin sind ihre Genetischen Informationen zu verbreiten

Die zugrundeliegenden Idee ist, dass angepasstere Individuen eine höhere Chance haben Nachkommen zu erzeugen. [5]

Wie die meißten stochastischen Algorithmen haben genetische Algorithmen keine Garantie ein globales Optimum zu finden [5]

Das besonderen an genetischen Algorithmen ist, dass diese nicht mit Werten aus dem Suchraum des Problems selbst arbeiten, sondern mit Chromosomrepresentationen. [6] Ein Chromosom ist also eine Vektorrepresentation einer Lösung aus dem Suchraum. Die einzelnen Werte eines Chromosoms nennt man Gene und eine Menge an Genen bilden eine Population. [Siehe Grafik 01] Die Chromosomrepresentation kann Binär, Hexadezimal, mit Fließkommazahlen oder auch anderen Symbolen geschehen. Welche Werte die Gene einnehmen können hängt von der Problemdomäne ab.




Die Hauptbestandteile eines Genetischen Algorithmus, welche von jeder Variante implementiert werden sind Representationsoperator, Fitnessfunktion und genetische Operatoren.

Der Representationsoperator übersetzt eine Lösung aus dem Suchraum in eine Chromosomrepresentation.

Die Fitnessfunktion einem gegebenen Chromosom einen Fitnesswert zu, anhand welcher die Chromosome verglichen werden können. Die Fitness eines Chromosoms entspricht dem Wert der objektiven Funktion an der Stelle welche das Chromosom representiert. [6]

Die genetischen Operatoren sind aufgeteilt in Selektionsoperator, Mutationsoperator und Crossoveroperator. Im folgenden werde ich die einzelnen Operatoren erläutern und beliebte Implementationen aufführen.

### 1.5.1 Selektionsoperator

Der Selektionsoperator wählt eine Menge von Eltern aus der Population, aus welchen Kinder für die nächste Generation erzeugt werden. Typischerweise hat jedes Kind-Chromosom zwei Eltern-Chromosome. Es gibt jedoch auch Crossover-Operatoren welche mit mehr als zwei Eltern arbeiten. Ganz nach der Evolutionstheorie sollen hier Chromosome mit einer höheren Fitness auch eine höhere Chance haben als Elternteil gewählt zu werden. Wie stark Chromosome mit einer höheren Fitness bevorzugt werden wird als Selektionsdruck bezeichnet. Selektionsschemata können in zwei Klassen unterteilt werden, proportionale Selektion und ordinale Selektion. [6] Eine proportionale Selektion gewichtet Chromosome anhand ihrer Fitness. Ordinale Selektion gewichtet Chromosome

anhand ihres Ranges. Der Selektionsdruck Bei einer proportionalen Selektion ist der Selektionsdruck hoch und es besteht das Risiko einer verfrühten Konvergenz. Denn wenn es ein Chromosom gibt, welches weitaus fitter als der Rest der Population ist wird dieses einen proportionalen Selektionsprozess dominieren und somit die genetische Diversität der Population senken. Andererseits führt ein geringer Selektionsdruck zu langsamer konvergenz. [6] Die Auswahl des Selektionsoperators sollte also wohlüberdacht sein.

#### Roulette Rad Selektion

Roulette Rad Selektion ist einer der traditionellen proportionalen Selektionsoperatoren. Stellen wir uns ein Roulette Rad vor, welches in $N$ Segmente unterteilt ist, wobei $N$ die Anzahl der Chromosome in einer Population ist. Die Länge eines Segmentes $s_i$ ist proportional zu der normalisierten Fitness des korrespondierenden Chromosoms.

$$s_i = 2\pi \cdot \frac{fitness(i)}{\sum_{j=1}^{N} fitness(j)} \tag{1.11}$$

Nun wird das Roulette Rad gedreht und das Chromosom auf wessen Feld man landet wird in die Menge der Eltern aufgenommen. Diese Prozedur wird so oft wiederholt bis man die gewünschte Anzahl an Eltern gesammelt hat.

#### Zufällige Selektion

Der zufällige Selektionsoperator ist der simpelste. Jedes Chromosom hat die gleiche Wahrscheinlichkeit in die Elternmenge aufgenommen zu werden.

#### Rang Selektion

Die Rang Selektion ist eine Abwandlung der Roulette Rad Selektion. Die länge eines Segmentes ist jedoch nicht proportional zu der fitness sondern proportional zu dem Rang des Chromosoms.

$$s_i = 2\pi \cdot \frac{N - rank(i)}{\sum_{j=1}^{N} rank(j)} \tag{1.12}$$

#### Elitismus

Die Selektionsoperatoren garantieren nicht, dass die besten Chromosome ausgewählt werden. Darüber hinaus kann es vorkommen, dass die besten Chromosome durch Crossover und Mutation verschlechtert werden, sodass die Fitness der folgenden Generation geringer als die vorherige ist. Um eine Abnahme der Fitness zu verhindern kann

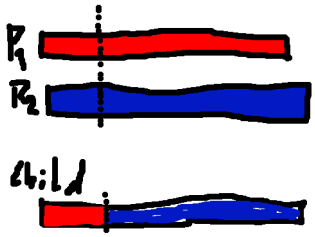
wird Elitismus angewendet. Die n besten Chromosome einer Population, die sogenannten Eliten werden auf jeden Fall in die nächste Generation aufgenommen ohne durch Crossover und Mutation verändert zu werden.

### 1.5.2 Crossoveroperator

Crossover bezeichnet das Rekombinieren von typischerweise zwei Eltern-Chromosomen zu einem Kind-Chromosom. Der vollständigkeit halber sei angeführt, dass auch Crossoveroperatoren existieren, welche mit mehr als zwei Eltern akzeptieren oder mehr als ein Kind produzieren. In dieser Arbeit beschränken wir uns jedoch auf Crossoveroperatoren von zwei Eltern zu einem Kind. Beim Crossover werden keine neuen Informationen in den Genpool eingeführt sondern es werden nur die vorhandenen Gene der Eltern rekombiniert. Es folg eine Aufzählung gängiger Crossoveroperatoren.

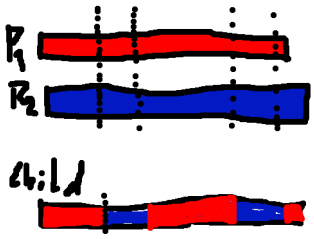
#### Single Point Crossover

Beim Single Point Crossover wird ein Cutpoint entlang der länge der Eltern gewählt. Beide Eltern werden dann an diesem Cutpoint geschnitten und das Kind-Chromosom setzt sich zusammen aus der ersten Hälfte des ersten Elternteils und der zweiten Hälfte des zweiten Elternteils.



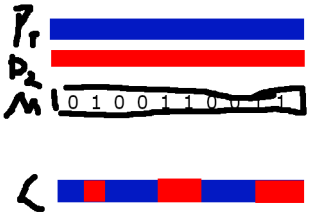
#### N-Point Crossover

Der N-Point Crossover ist eine Generallisierung des Single Point Crossovers. Es werden $n$ Crossover Points entlang der Länge der Eltern gewählt. Anhand dieser Crossover Points werden die Eltern in $n + 1$ Segmente unterteilt. Das Kind erhält alle Segmente mit geradem Index vom ersten Elternteil und alle Segmente mit ungeradem Index vom zweiten Elternteil. Im Allgemeinen führen mehr cutpoints jedoch zu einer geringeren Effizienz des genetischen Algorithmus. [6]



#### Uniform Crossover

Bei einem uniformen Crossover wird eine Crossovermaske $m$ mit gleicher Länge zu den Eltern erstellt. Das Kind erhält Gene der Eltern nach dieser Crossover Maske, wobei $m_i$ angibt, von welchem Elternteil das $i$-te Gen bezogen wird. Für jedes Elternpaar wird eine neue Crossovermaske erstellt. Typischerweise gilt $P(m_i = 1) = P(m_1 = 0) = 0.5$. Die Wahrscheinlichkeit, dass ein Gen von einem Elternteil bezogen wird kann jedoch auch gewichtet werden anhand der Fitness oder Ränge, so dass $P(m_i = 0) = w$ und $P(m_i = 1) = 1 - w$




### 1.5.3 Mutationsoperator

Der Crossoverschritt verringert unweigerlich die genetische Diversität der Population. Um dem entgegenzuwirken muss es einen Mechanismus geben, welcher neues genetisches Material hinzufügt. Dieser Mechanismus ist die Mutation, welche ein Chromosom zufällig verändert.

TODO: Include Examples for Real valued GA crossover

#### Mutationschance

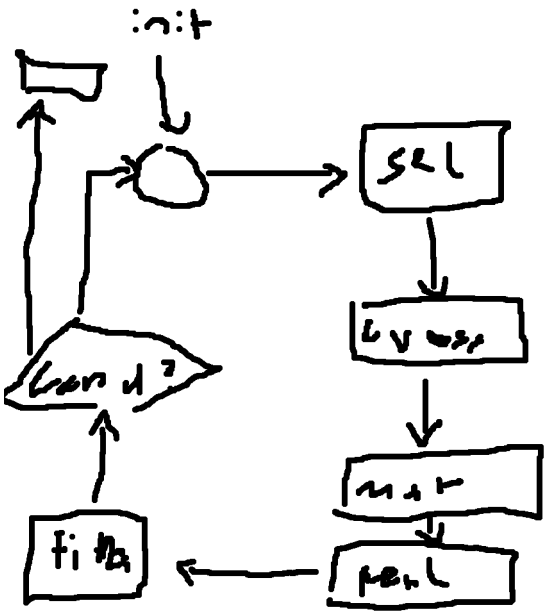
Die Mutationschance bestimmt wie viele Gene eines Chromosoms durch die Mutation verändert werden und wie viele Unverändert bleiben. Bei einer Mutationschance von 100% werden alle Gene eines Chromosoms verändert und der genetische Algorithmus ist equivalent zu einer zufälligen Suche. [6]

### 1.5.4 Ersetzen

TODO: Ersetzen moped schmoped

### 1.5.5 Ablauf eines genetischen Algorithmus

Ein Iterationsschritt des klassischen genetischen Algorithmu besteht aus Vier Schritten. Die Abbruchbedingung eines Genetischen Algorithmus kann zum Beispiel vom Mittelwert, der Summe oder dem Maximalwert der Fitness abhängen oder aber auch einfach nach einer vorher festgelegten Anzahl an Iterationen terminieren. Der Ablauf einer Iteration ist in Grafik so und so beschrieben.




Population Size Wählt man diese Zu gering kann es zu einer verfrühten Konvergenz und somit auch einer schlechten Lösung kommen Gleichzeitig ist die Population size ein Maßgebender Faktor in der Rechenzeit und eine zu große Population könnte Rechenzeit verschwenden [GAs]

Im folgenden werde ich bekannte implementationen der genetischen Operatoren eingehen und beschreiben worauf man bei der implementation achten muss

#### Mutationsoperatoren

Random Uniform Mutation: Random Unifor mutation

### Parameter eines genetischen Algorithmus

- Welche Crossover Funktion (evtl. Parameter der Crossover Funktion) - Welche Mutations Funktion (evtl. Parameter der Mutations Funktion) - Populationsgröße - Rekombination - Selektionsfunktion -

- Populationsgröße Die globale Suchkapazität eines genetischen Algorithmus hängt stark von der gewählten Populationsgröße ab. Eine größere Population ist hier von Vorteil. Jedoch benötigt eine große Population auch mehr Rechenleistung, Speicher und Zeit. [6]

- Auswahl des Mutationsoperators. Mutationsoperatoren unterscheiden sich stark in der benötigten Rechenleistung. Der Single Point Crossover-Operator benötigt zum Beispiel nur ein einziges Zufallsereignis, wohingegen ein Uniformer Crossover-Operator so viele Zufallsereignisse wie es Gene gibt benötigt. Bei einer Chromosomrepresentation mit vielen Genen macht sich dieser Unterschied bemerkbar.

# Chapter 2

# Trainieren von HMMs mit einem GA

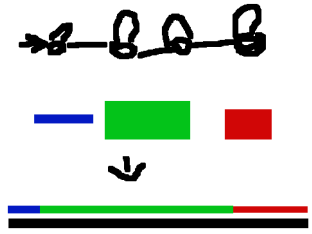
## 2.1 Konstruktion eines GAHMM

In diesem Kapitel werde ich beschreiben wie man einen Genetischen Algorithmus für Hidden Markov Modelle erstellen kann und dann auf die in meiner Arbeit verwendeten konkreten Implementationen der genetischen Operatoren eingehen.

### Representation

Ähnlich wie in den Papers [hier papers einfügen] Ist die Chromosonale Representation eines HMMs ein Vektor welcher die Reihen aller Matrizen hintereinander enthält.

Um die Struktur von nicht ergodischen Hidden Markov Modellen im Mutations-schritt nicht zu Verändern muss man die Gene Welche Startwahrscheinlichkeiten und Transitionswahrscheinlichkeiten beschreiben von der Mutation ausschließen oder man kann eine Maske definieren welche eine Mutation Gene welche initial 0 oder 1 sind verhindert.



### Fitness Operator

Die Fitness eines Chromosoms ist die durchschnittliche Log Wahrscheinlichkeit des Hidden Markov Models, welchen das Chromosom representiert. Als Observations Sequenzen werden Äußerungen der Zahl 0 aus dem Free Spoken Digit Dataset verwendet

### Mutationsoperatoren

Mutationsoperatoren wird dies das annanas gewählt

Belegung der Parameter:

Hidden Markov Parameter Bei den Hidden Markov Modellen welche von den genetischen Algorithmus optimiert werden handelt es sich ausschließlich um Left-right Modelle mit 4 Zuständen und 128 Observationssymbolen. Diese Werte sind relativ Arbiträr gewählt und orientieren sich an existierender Literatur zu Spracherkennung mit Hidden Markov modellen.

Als Observations-Sequenzen werden Äußerungen der Zahl 0 aus der Free Spoken Digit Database gewählt. Welche mittels eines k-means algorithmus mit k=128 quantisiert wurden.

Genetischer Algorithmus Parameter:

Die Populationsgröße wird auf 50 beschränkt und die Anzahl der Generationen auf 100. Das Rational für diese Entscheidung ist, dass ein Ablauf des Genetischen Algorithmus mit diesen Parametern auf meinem Klapprechner in unter 5 minuten abläuft und 100 Generationen meißt ausreichen um den Genetischen Algorithmus konvergieren zu lassen Die Anzahl der Observationssequenzen ist 10

### Erweiterung des GA

- Der genetische Algorithmus muss um einen Normalisierungsschritt erweitert werden, falls die Crossover oder Mutationsoperatoren keine Reihenstochastizität garantieren.

## 2.2 Literaturübersicht

TODO: in diesem Chapter gebe ich eine übersicht über die unternommenen Ansätze das Training von Hidden Markov Modellen mit hybriden metaheuristischen Algorithmen.

Wir können drei Kategorien für die verwendung von genetischen Algorithmen für das trainieren von Hidden Markov Modellen erkennen. - Genetischer Algorithmus ohne Baum-Welch (Chau 1997) - Genetischer Algorithmus vor Baum-Welch fürs finden solider initialer Belegungen der Parameter (Slimane 1996)

- GA wird hybrid mit BW angewendet, so dass alle n Generation BW k mal angewendet wird.

Optimization of HMM By A Genetic Algorithm: - Genetischer Algorithmus ohne Baum Welch - 20000 GA Iterationen - population size - crossover-rate 0.01 - mutation rate 0.0001 - crossover type single point crossover - mutation type N-Point

Baum-welch - max 200 iterationen - 1 HMM

# Chapter 3

# Evaluation

## 3.1 Vergleich genetischer Operatoren

## 3.2 Berechnungskost eines GA

- Wallah Teuer - Lohnt net

# Bibliography


[1] Ehrhard Behrends. *Elementare Stochastik*. 2012.

[2] *Grey Wolf Firefly and Bat Algorithms Three Widespread Algorithms that Do Not Contain Any Novelty*. 2020.

[3] Dominik Wied Karsten Webel. *Stochastische Prozesse*. 2016.

[4] Christian Kohlschein. *An introduction to Hidden Markov Models*.

[5] Sivanandam. *Genetic Algorithms. In: Introduction to Genetic Algorithms. Springer*. 2008.

[6] Sivanandam. *Terminologies and Operators of GA. In: Introduction to Genetic Algorithms. Springer*. 2008.

[7] Kenneth Sorensen. *Metaheuristics—the metaphor exposed*. 2012.

[8] Springer. *Metaheuristics*. 2008.

[9] Springer. *Metaheuristics and Evolutionary Computation: Algorithms and Applications*.

[10] Dennis Weyland. *A rigoruous analysis vong dem harmony moped*. 2010.

[11] Gordan Žitković. *Introduction to Stochastic Processes - Lecture Notes*. 2010.

**Eigenständigkeitserklärung**

Hiermit versichere ich, dass ich diese Arbeit bzw. im Fall einer Gruppenarbeit den von mir entsprechend gekennzeichneten Anteil an der Arbeit selbständig verfasst habe. Ich habe keine unzulässige Hilfe Dritter in Anspruch genommen. Zudem habe ich keine anderen als die angegebenen Quellen und Hilfsmittel benutzt und alle Ausführungen (insbesondere Zitate), die anderen Quellen wörtlich oder sinngemäß entnommen wurden, kenntlich gemacht. Ich versichere, dass die von mir in elektronischer Form eingereichte Version dieser Arbeit mit den eingereichten gedruckten Exemplaren übereinstimmt. Mir ist bekannt, dass im Falle eines Täuschungsversuches die betreffende Leistung als mit "nicht ausreichend" (5,0) bewertet gilt. Zudem kann ein Täuschungsversuch als Ordnungswidrigkeit mit einer Geldbuße von bis zu 50.000 Euro geahndet werden. Im Falle eines mehrfachen oder sonstigen schwerwiegenden Täuschungsversuchs kann ich zudem exmatrikuliert werden. Mir ist bekannt, dass sich die Prüferin oder der Prüfer bzw. der Prüfungsausschuss zur Feststellung der Täuschung des Einsatzes einer entsprechenden Software oder sonstiger elektronischer Hilfsmittel bedienen kann.

Duisburg, 28. Februar 2023 _____

(Ort, Datum) (Vorname Nachname)