Evolution von Spielstrategien für das Spiel "The Resistance"

Institut für Mathematik und Informatik
der Universität Greifswald

Abschlussarbeit

zur Erlangung des akademischen Grades
Bachelor of Science

eingereicht von

Johanna Lange

am 30. September 2020

Erstgutachter:	Prof. Dr. M. Ebner
Zweitgutachter:	Prof. Dr. M. Stanke

Danksagungen

An dieser Stelle möchte ich mich ganz herzlich bei Prof. Dr. Ebner für die sehr gute Betreuung der Bachelorarbeit bedanken. Er hat das Thema dieser Arbeit vorgeschlagen und stand bei Fragen stets zur Verfügung. Die regelmäßigen Besprechungen und das Engagement von Prof. Dr. Ebner waren eine große Hilfe.

Weiterer Dank gilt Florian Perner für das Korrekturlesen der Ausarbeitung. Zu guter Letzt danke ich auch meinen Eltern, die mich während des Bachelorstudiums in jeglicher Hinsicht unterstützt haben.

Inhaltsverzeichnis

1 Einleitung

„The Resistance“ ist ein spannendes, kartenbasiertes Brettspiel mit geheimen Identitäten. Täuschungen, Enthüllungen und hitzige Diskussionen sind wesentliche Bestandteile des Spiels. Jeder Spieler erhält eine Rolle und ist entweder ein Mitglied des Widerstands oder ein Spion. Nun heißt es: Der Widerstand gegen die Spione. Wer gewinnt? Basierend auf Diskussionen und den Spielzügen der Mitspieler versuchen die Widerstandskämpfer, die Spione unter ihnen zu entlarven. Die Spione jedoch können die gegnerische Partei täuschen, manipulieren und die Widerstandsmissionen sabotieren. In bis zu fünf Runden versuchen die Spieler, den Sieg für ihre Partei zu erringen. Dabei besteht jede Runde aus drei elementaren Handlungen: der Zusammenstellung eines Missionsteams, der demokratischen Abstimmung und der Mission. Nur wenn die Widerstandskämpfer drei Missionen erfolgreich durchführen, gehört die hart erkämpfte Freiheit und somit der Sieg ihnen.

Das Ziel dieser Bachelorarbeit ist es, erfolgversprechende Spielstrategien für Widerstandskämpfer und Spione zu entwickeln. Hierfür soll ein geeignetes Modell für die Strategien erstellt und das Spiel am Computer simuliert werden. Wie sollte der Spieler jeweils als Spion oder Widerstandskämpfer handeln, um seine Gewinnchancen zu maximieren? Es gilt herauszufinden, ob es eine auf dem Modell basierende optimale Spielstrategie für die beiden Rollen gibt. Um dieses Ziel zu erreichen, wird ein evolutionärer Algorithmus verwendet, welcher die Strategien optimiert. Die Evolution der Spielstrategien ist erfolgreich, wenn ein Lernfortschritt der Strategien nachgewiesen werden kann. Eine kontinuierliche Verbesserung der Strategien zu erzeugen, ist somit ebenfalls ein wichtiges Ziel dieser Bachelorarbeit.

Evolutionäre Algorithmen sind vor allem für komplexe Optimierungsprobleme gefragt, welche nicht mit anderen Optimierungstechniken gelöst werden können. Sie erzeugen gute Näherungslösungen für das Problem oder im besten Fall auch die optimale Lösung, sofern diese existiert. Ihre Funktionsweise ist von der natürlichen Evolution der Lebewesen inspiriert. Im Sinne der Darwin‘schen Evolutionstheorie beschreibt „Survival of the Fittest“ das Überleben der am besten an die Umwelt angepassten Individuen. Dieses Prinzip wird auf abstrakte Weise mithilfe der evolutionären Algorithmen am Computer simuliert und somit werden immer bessere Lösungen für das Optimierungsproblem evolviert. In der Natur besitzt jedes Individuum eine sogenannte Fitness, welche als Maß für den Fortpflanzungserfolg des Individuums steht. Die am besten angepassten Individuen besitzen eine hohe Fitness. Ihre Gene werden

sich langfristig gesehen in den nachfolgenden Generationen durchsetzen. Für den evolutionären Algorithmus sind die Individuen mögliche Lösungen für das Optimierungsproblem, also in diesem Fall die Spielstrategien. Anhand der Fitness, welche hier die Bewertung einer Strategie darstellt, werden die besten Individuen ausgewählt, um Nachkommen zu erzeugen. Genetische Variation der Nachkommen entsteht durch Rekombination der Eltern sowie durch Mutationen im Genmaterial. Die besten Spielstrategien sollen immer wieder verändert und miteinander kombiniert werden, um neue Strategien zu erzeugen, welche besser „angepasst“ sind als ihre Vorfahren.
Bevor die Evolution am Computer simuliert werden kann, wird eine geeignete Modellierung der Spielstrategien benötigt. Evolutionsstrategien, welche ein Teilgebiet der evolutionären Algorithmen darstellen, optimieren reellwertige Parameter. Die hier verwendete Parametrisierung der Strategien ist auf dem Raum der reellen Zahlen, beziehungsweise einer Teilmenge von diesem, definiert worden. Die Evolution der Spielstrategien erfolgt daher mit einer Evolutionsstrategie, wobei die zu optimierenden Strategien als Parameterlisten aufgefasst werden können.

In dieser Arbeit werden zunächst die Spielregeln von „The Resistance“ und die theoretischen Grundlagen der evolutionären Algorithmen genauer erklärt. Im Speziellen wird dabei auf Evolutionsstrategien eingegangen und außerdem die Thematik der Co-Evolution eingeführt. Anschließend wird das verwendete Modell für die Spielstrategien sowie die Implementation der Spielumgebung erläutert. Diese ermöglicht es, das Spiel mit fünf Spielern durchzuführen, wobei die Spielzüge der Spieler durch die Parameter der Strategie bestimmt werden. Anhand vieler simulierter Spiele können die einzelnen Strategien bewertet werden. Diese Bewertung dient als Fitness für die Evolution. Nachdem die Implementation der Evolutionsstrategie beschrieben wird, werden im letzten Teil der Arbeit die durch Evolution erzeugten Strategien ausgewertet und mit geeigneten Messmethoden geprüft, ob ein Lernfortschritt nachgewiesen werden kann.

2 Die Spielregeln von „The Resistance“

Das von Don Eskridge entwickelte Brettspiel „The Resistance“ wurde 2010 durch Indie Boards and Cards veröffentlicht. Das Spiel ist für fünf bis zehn Spieler ausgelegt und basiert auf geheimen Spieleridentitäten. Logische Schlussfolgerungen, Täuschungen, Kommunikation und auch Intuition sind Bestandteile des Spiels. Im Folgenden werden die Spielregeln (siehe [1]) genau erklärt.

Eine Gruppe von Widerstandskämpfern hat es sich zum Ziel gesetzt, ihre mächtige Regierung zu bezwingen, um sich Freiheit zu erkämpfen. Die Widerstandskämpfer treten in kleinen Teams verschiedene Missionen an, um ihren Plan in die Tat umzusetzen. Unter ihnen befinden sich jedoch auch Spione der Regierung, welche versuchen werden, die Missionen zu sabotieren. Dessen sind sich die Widerstandskämpfer bewusst. Wem können sie trauen und wem nicht? Sie müssen bei der Teamzusammenstellung sehr vorsichtig sein, denn schon ein einziger Spion im Team kann die komplette Mission gefährden.
Im Spiel „The Resistance“ gibt es zwei Gruppen: Die Widerstandskämpfer und die Spione. Es gewinnen die Spieler derjenigen Gruppe, welche als erste drei Punkte erzielt hat. Um einen Punkt zu erhalten, müssen die Widerstandskämpfer eine Mission erfolgreich durchführen. Die Spione bekommen einen Punkt, wenn sie eine Mission sabotieren. Es kann bis zu fünf Missionen geben, bis eine Gruppe den Sieg erlangt hat.
Zu Beginn einer Partie erhalten alle Spieler zunächst eine geheime Identität, welche die Zugehörigkeit zu einer der beiden Gruppen bestimmt. Die Zusammensetzung der Spione und Widerstandskämpfer ist für jede Spieleranzahl genau festgelegt (siehe Tabelle 2.1).

Spieleranzahl	5	6	7	8	9	10
Widerstandskämpfer	3	4	4	5	6	6
Spione	2	2	3	3	3	4

Tabelle 2.1: Rollenzusammensetzung von Spionen und Widerstandskämpfern nach der Spieleranzahl

Eine Spielpartie besteht aus drei bis fünf Runden. Die Runden werden dabei in drei Phasen unterteilt: die Teamauswahl, Abstimmung und die Durchführung der Mission. Bevor die erste Runde beginnt, erfahren die Spione, wer ihre Kollegen sind und wer zu den Widerstandskämpfern gehört. Sie erhalten die volle Information über die Rollen aller Spieler. Als Spion besteht die Aufgabe darin, nicht als solcher identifiziert zu werden. Er muss seine Spielzüge

gut abwägen und versuchen, sich als Widerstandskämpfer zu verkaufen. Die Widerstandskämpfer bleiben jedoch im Ungewissen und kennen nur ihre eigene Identität. Für sie gilt es während des Spielverlaufs herauszufinden, wer die Spione sind. Anhand von Spielzügen der Mitspieler können oftmals Schlussfolgerungen über die Identitäten gezogen werden. Außerdem ist die Kommunikation ein wesentlicher Bestandteil des Spiels. Allen Spielern ist es erlaubt, sich jederzeit öffentlich über ihre Vermutungen zu äußern und mit anderen zu diskutieren. Dabei wissen die Widerstandskämpfer nicht, wessen Aussagen und Anschuldigungen sie trauen können.

2.1 Ablauf

Nachdem die geheime Rollenzuweisung und die Enthüllung der Identitäten für die Spione erfolgt sind, ist der Ablauf des Spiels durch die folgenden drei Phasen festgelegt:

Teamauswahl:
Der Missionsleiter hat die Aufgabe, ein Team vorgegebener Größe für die Mission zusammenzustellen. In der ersten Runde wird der Missionsleiter zufällig ausgewählt. Danach ist immer der nächste Spieler im Uhrzeigersinn der neue Missionsleiter. Auch hier gilt, dass die Größe der Teams je nach Spieleranzahl variiert. In Tabelle 2.2 sind die Zahlen für die jeweiligen Missionen aufgelistet.

	1. Mission	2. Mission	3. Mission	4. Mission	5. Mission
5 Spieler	2	3	2	3	3
6 Spieler	2	3	4	3	4
7 Spieler	2	3	3	4*	4
8 Spieler	3	4	4	5*	5
9 Spieler	3	4	4	5*	5
10 Spieler	3	4	4	5*	5

Tabelle 2.2: Größe des Missionsteams
* mindestens zwei Spione müssen sabotieren, damit die Mission fehlschlägt

Der Missionsleiter darf sich auch selbst für das Team vorschlagen. Die Widerstandskämpfer wollen möglichst keine Spione im Team dabei haben. Die Spione aber versuchen, die Widerstandskämpfer zu täuschen, um möglichst als Mitglied für das Team ausgewählt zu werden.

Abstimmung:
Nun stimmen die Spieler ab, ob sie mit dem vorgeschlagenen Team einverstanden sind oder nicht. Dafür werden alle Spieler gleichzeitig aufgefordert, entweder mit „Ja“ oder „Nein“ zu stimmen. Nur mit einer Wahlmehrheit wird das Team für die Mission zugelassen. Bei zum

Beispiel fünf Spielern werden mindestens drei „Ja“-Stimmen benötigt, um das Team auf die Mission zu schicken. Bei einer geraden Spieleranzahl wird das Team bei einem Unentschieden abgelehnt. Falls das Team nicht mit der Wahlmehrheit bestätigt wird, muss ein neues Team vorgeschlagen werden. Dies ist die Aufgabe des nächsten Missionsleiters. Dann wird erneut für oder gegen das Team abgestimmt. Wenn die Abstimmung in einer Runde fünf mal in Folge zu keiner Mehrheit geführt hat, dann haben die Spione in diesem Fall automatisch gewonnen.

Mission:
Das durch die Wahl bestätigte Team tritt nun die Mission an. Diese ist so aufgebaut, dass sich jedes Mitglied des Teams entweder für den Erfolg oder Misserfolg der Mission entscheidet. Hierfür erhalten die Teammitglieder je eine Erfolg- und Misserfolg-Karte. Jeder wählt im Geheimen eine der beiden Karten aus. Ist das Teammitglied ein Widerstandskämpfer, so muss die Erfolg-Karte ausgewählt werden (ansonsten würden die Widerstandskämpfer ihr eigenes Spielziel verfehlen). Als Spion jedoch hat man die Wahl. Ein Spion muss sich entscheiden, ob er die Mission sabotieren möchte oder nicht. Durch Sabotage würde er das Spielziel der Spione voranbringen. Indem er nicht sabotiert, kann er aber vielleicht den Anschein wecken, selbst ein Widerstandskämpfer zu sein. Er muss seine Handlungen auch abwägen, falls sich mehrere Spione in dem Team befinden. Die Auswahl findet anonym statt, sodass kein Spieler weiß, wer sich wie entschieden hat. Sobald eine Misserfolg-Karte dabei ist, gilt die Mission als gescheitert und die Spione erhalten einen Punkt. Wenn alle Karten den Erfolg zeigen, geht der Punkt an die Widerstandskämpfer. Wenn noch keine der beiden Gruppen insgesamt drei Punkte gesammelt hat, startet die nächste Runde. Ansonsten ist das Spiel vorbei und die Siegergruppe steht fest.

3 Evolutionäre Algorithmen

Die Anfänge der evolutionären Algorithmen liegen in den 1950er Jahren. Sie sind ein sehr wichtiger Bestandteil im Bereich von Computational Intelligence und werden verwendet, um Lösungen für verschiedene Optimierungsprobleme zu finden [2]. Evolutionäre Algorithmen sind insbesondere dann gefragt, wenn das Problem nicht mit anderen Optimierungsverfahren gelöst werden kann. Beispielsweise ist die Zielfunktion nicht differenzierbar und ein Gradientenverfahren kann dementsprechend nicht angewandt werden. Evolutionäre Algorithmen arbeiten direkt mit der Zielfunktion und benötigen keine Ableitungen [3]. Die Optimierungsprobleme dürfen sehr komplex sein.
Die evolutionären Algorithmen werden heute in drei Hauptbereiche unterteilt: Genetische Algorithmen, genetische Programmierung und Evolutionsstrategien [2]. Die ersten beiden Ansätze eignen sich zur Parameteroptimierung, wobei Evolutionsstrategien zur Optimierung kontinuierlicher Parameter verwendet und genetische Algorithmen häufig für kombinatorische Optimierungsprobleme eingesetzt werden. Mit genetischer Programmierung werden Computerprogramme evolviert. Die Grundidee hinter diesen drei Ansätzen ist die gleiche. Evolutionäre Algorithmen ahmen den Prozess der natürlichen Evolution nach und suchen auf diese Art und Weise die optimale Lösung für das Problem. Nach dem Vorbild der Natur wird die Evolution auf vereinfachte Weise am Computer simuliert und dadurch werden immer bessere Lösungen ermittelt. Die Funktionsweise der evolutionären Algorithmen basiert daher auf zufälligen Prozessen. Aufgrund dieser Zufälligkeit kann allerdings nicht immer garantiert werden, dass der Algorithmus letztendlich die optimale Lösung findet, sofern diese existiert. Jedoch können gute Näherungslösungen evolviert werden.

Für diese Bachelorarbeit wurden die Strategien für das Spiel „The Resistance" mit Hilfe einer Evolutionsstrategie optimiert. Im Folgenden wird kurz auf den biologischen Hintergrund eingegangen, dann die grundlegende Funktionsweise der evolutionären Algorithmen erläutert und anschließend auf die Evolutionsstrategien eingegangen. Von weiterer Bedeutung ist außerdem die sogenannte kompetitive Co-Evolution, welche am Ende dieses Kapitels eingeführt wird.

3.1 Biologischer Hintergund

„Survival of the fittest" - Dieser Ausdruck ist vielen bekannt. Im Sinne der Darwin'schen Evolutionstheorie wird damit das Überleben der am besten an die Umwelt angepassten Individuen beschrieben.
Lebewesen stehen stets im Wettbewerb miteinander. Es herrscht sowohl Konkurrenz zwischen Individuen derselben Art, als auch zwischen Individuen verschiedener Arten. Sie konkurrieren dabei zum Beispiel um Ressourcen in ihrem Lebensraum, da diese nämlich nicht unbegrenzt für alle verfügbar sind. Aus diesem Grund können nur diejenigen überleben, die am besten an ihre Umwelt angepasst sind. Die Angepasstheit eines Individuums wird durch seine sogenannte Fitness beschrieben. Darunter versteht man in der Biologie ein Maß für den Fortpflanzungserfolg eines Individuums und dieser ist relativ zu dem der anderen Individuen in der Population zu betrachten. Anders ausgedrückt, ist die Fitness ein Maß für den genetischen Beitrag, den ein Individuum zur nächsten Generation leistet. Individuen mit einer hohen Fitness sind besser angepasst und haben dementsprechend mehr überlebens- und fortpflanzungsfähige Nachkommen im Vergleich zu anderen Individuen. Dadurch setzen sich bestimmte Eigenschaften, wie zum Beispiel Verhaltensstrategien und auch das Aussehen, in der Evolution durch, wenn es dem Lebewesen zu einer hohen Fitness und damit zu vielen Nachkommen verhilft. Der dahinter stehende Prozess ist die Selektion, welche als „natürliche Auslese" bezeichnet werden kann.
Evolution durch Selektion lässt sich demnach wie folgt erklären: Auf natürliche Art und Weise erfolgt genetische Variation durch Mutationen in der DNA und Rekombinationen bei der Fortpflanzung. Die natürliche Evolution basiert auf dem Zufall, da die Variationen in den Genen zufällig und ungerichtet entstehen [4]. Variationen, welche zu vorteilhaften Eigenschaften für das Überleben in der Umwelt führen, werden durch die Selektion gefördert, indem diese Individuen viele Nachkommen haben. Andere sterben langfristig gesehen aus. Über einen großen Zeitraum hinweg verbessert sich langsam die Angepasstheit der Individuen durch den Prozess der Selektion.

3.2 Grundlegende Funktionsweise evolutionärer Algorithmen

„Die Evolution selbst ist ein ständiger Kreislauf von Neuschaffung, Überlebensprüfung, Vermehrung der Besten und Veränderung des Vorhandenen, um sich immer wieder neu anzupassen und zu verbessern [2, S. 35]." Evolutionäre Algorithmen simulieren diesen Kreislauf zur Lösung eines Optimierungsproblems. Die Funktionsweise basiert also auf den Prinzipien von Variation, Reproduktion und Selektion. Eine möglichst genaue Modellierung der natürlichen Evolution ist dabei nicht das Ziel. Evolutionäre Algorithmen ahmen die natürliche Evolution

nur auf sehr abstrakte Art und Weise nach [2].

3.2.1 Grundbegriffe

Im Kontext der evolutionären Algorithmen werden einige biologische Begriffe verwendet. In Tabelle 3.1 wird die biologische Bedeutung dieser Begriffe kurz eingeführt und außerdem die Bedeutung für den evolutionären Algorithmus erklärt. Eine Vertiefung der biologischen Kenntnisse ist für das Verständnis der evolutionären Algorithmen nicht notwendig, da die Begriffe nur in einem sehr abstrakten Sinn verwendet werden. Die Veranschaulichung einiger Begriffe ist in Abbildung 3.1 zu sehen.

Begriff	Biologische Bedeutung	Evolutionäre Algorithmen
Chromosom	enthält die Erbinformationen	eine Zeichen- oder Zahlenkette
Gen	Abschnitt im Chromosom (legt bestimmte Eigenschaft des Lebewesens fest)	ein Zeichen oder eine Zahl im Chromosom
Allele	alle möglichen Ausprägungen eines Gens	alle gültigen Zeichen bzw. Zahlen
Individuum	ein Lebewesen	ein Lösungskandidat bestehend aus (meistens) einem Chromosom
Population	Menge von Lebewesen	Menge von Lösungskandidaten
Generation	Population zu einem Zeitpunkt	analog
Genotyp	genetische Zusammensetzung = die Kodierung	eine Kombination der Zeichen- oder Zahlenabfolge im Chromosom = die Kodierung
Phänotyp	Erscheinungsbild des Lebewesens (vom Genotyp und der Umwelt bestimmt)	Verhalten des Lösungskandidaten in der Implementierung, welches vom Genotyp bestimmt wird
Fitness	Fortpflanzungserfolg	Güte einer Lösung

Tabelle 3.1: Biologische Begriffe im Kontext der evolutionären Algorithmen [2] [4]

3.2.2 Funktionsweise

Das Flussdiagramm in Abbildung 3.2 stellt die Funktionsweise eines evolutionären Algorithmus schematisch dar. Die Realisierung der einzelnen Schritte sieht je nach verwendetem evolutionären Algorithmus etwas anders aus.

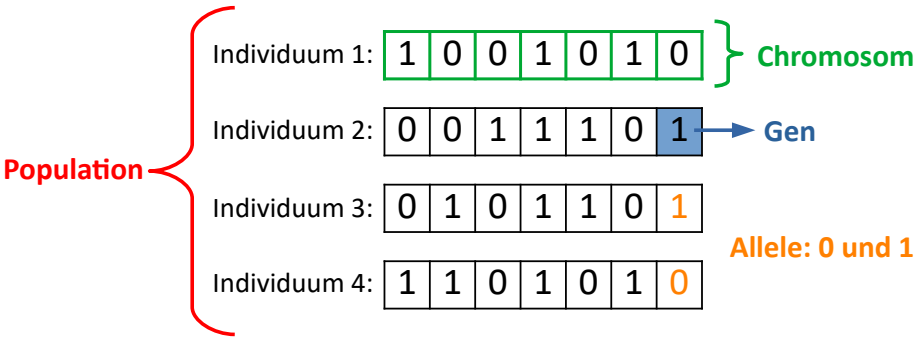


Abbildung 3.1: Veranschaulichung der Begriffe anhand binärer Zeichenketten, welche üblicherweise bei genetischen Algorithmen verwendet werden

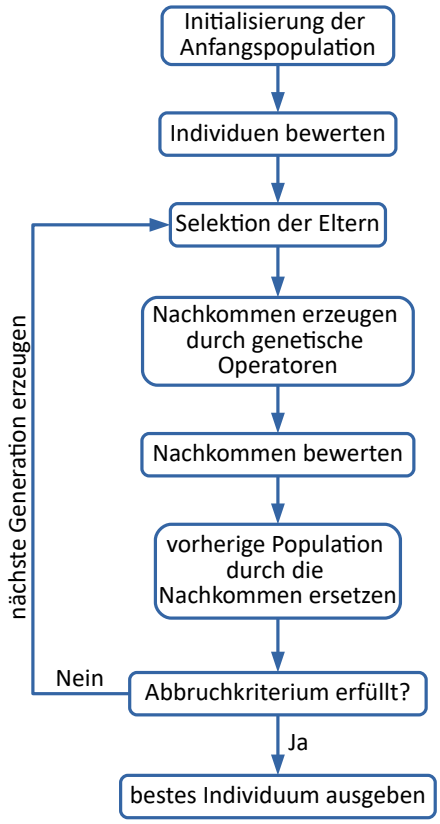


Abbildung 3.2: Flussdiagramm eines evolutionären Algorithmus

Der Algorithmus startet mit einer meist zufällig initialisierten Anfangspopulation von Individuen. Die Größe der Population kann selbst gewählt werden. Die Individuen sind mögliche Lösungen für das Optimierungsprobem und sind problemspezifisch (zum Beispiel als Zahlenkette) kodiert [3]. Alle Individuen der Population werden mit der sogenannten Fitnessfunktion bewertet, um zu ermitteln, wie gut ein Individuum das Optimierungsproblem löst. Als nächstes soll eine neue Generation von Individuen erzeugt werden. Hierfür müssen zunächst Eltern-Individuen ausgewählt werden, was mithilfe einer geeigneten Selektionsmethode geschieht. In der Regel wird durch die Selektion ein Teil der am besten bewerteten Individuen als Eltern beibehalten und die restlichen Individuen verworfen. Ganz im Sinne der Evolutionstheorie überleben überwiegend die besser angepassten Individuen. Die Individuen aus dem Eltern-Pool werden nun verwendet, um Nachkommen zu erzeugen, was durch die genetischen Operatoren geschieht. Zwei Eltern-Individuen werden zum Beispiel durch ein Crossover rekombiniert, wobei zufällige Bestandteile der zwei Lösungen vertauscht werden. Eine weiterer genetischer Operator ist die Mutation. Hierbei wird das Chromosom eines Eltern-Individuums auf zufällige Art und Weise verändert, wodurch eine neue Lösung entsteht. Abhängig davon, welcher evolutionäre Algorithmus angewandt wird, unterscheidet sich die Umsetzung der jeweiligen genetischen Operatoren. Die Nachkommen entsprechen den variierten Eltern-Chromosomen. Ein Eltern-Individuum kann auch an der Produktion mehrerer Nachkommen beteiligt sein. Es kann außerdem festgelegt werden, dass manche Lösungen unverändert als Nachkommen verwendet werden. In der Regel werden solange Eltern zur Nachkommen-Produktion herangezogen, bis die Zahl der Nachkommen so groß ist, wie die Größe der Ursprungspopulation. Die Nachkommen werden anschließend mit der Fitnessfunktion bewertet und die komplette vorherige Population durch die Nachkommen ersetzt. Nun wird geprüft, ob die neue Generation das Abbruchkriterium erfüllt. Dieses könnte zum Beispiel so lauten: Ist in der neuen Generation eine Lösung vorhanden, welche eine geforderte Mindestgüte erfüllt? Ist dies nicht der Fall, wird eine neue Generation von Individuen nach demselben Schema erstellt. Der evolutionäre Algorithmus simuliert eine Generation nach der anderen, bis das Abbruchkriterium erfüllt wird. Zuletzt kann das beste Individuum ausgegeben werden.

Durch die Variation vorhandener Lösungen können zufällig bessere Lösungen entstehen. Diese würden mit einer höheren Fitness bewertet werden und wären demnach höchstwahrscheinlich an der Produktion von Nachkommen beteiligt. Das Ziel in einem evolutionären Algorithmus ist es, durch diesen Kreislauf immer bessere Lösungen zu konstruieren, bis eine hinreichend gute oder optimale Lösung gefunden wurde oder ein anderes Abbruchkriterium den Algorithmus beendet. Der Fortschritt in der Evolution lässt sich anhand ansteigender Fitness-Werte im Laufe der Generationen nachvollziehen.

Die folgenden Elemente werden für einen evolutionären Algorithmus benötigt [4]:

- eine Struktur zur Repräsentation der möglichen Lösungen für das Optimierungsproblem
- eine Anfangspopulation von Lösungskandidaten
- eine vom Anwender problemspezifisch definierte Fitnessfunktion $f : S \to \mathbb{R}$ zur Bewertung aller Lösungen des Suchraums S
- eine entsprechend geeignete Selektionsmethode
- ein oder mehrere genetische Operatoren (in der Regel Mutation und Crossover)
- einzustellende Parameter (zum Beispiel die Populationsgröße)
- ein Abbruchkriterium, welches die Evolution beendet

Die Bestandteile werden im nachfolgenden Abschnitt für die Evolutionsstrategien konkretisiert.

3.3 Evolutionsstrategien

Evolutionsstrategien werden für die Optimierung reellwertiger Parameter verwendet. Der kontinuierliche Suchraum S, welcher alle möglichen Lösungen enthält, ist der $\mathbb{R}^n$ oder eine Teilmenge von diesem [2]. Insgesamt gibt es für das Optimierungsproblem n zu optimierende Parameter. Das Chromosom eines Individuums x enthält dann n Gene, so dass $x = (x_1, x_2, \ldots, x_n)$, wobei $x_i \in \mathbb{R}$ für $i = 1, \ldots, n$ die zu optimierenden reellen Parameter sind [3].
Die Fitnessfunktion $f : S \to \mathbb{R}$ dient zur Bewertung der Individuen und ordnet jedem Element aus dem Suchraum $S \subseteq \mathbb{R}^n$ eine Fitness zu, welche die Güte der Lösung angibt. Demnach ist $f(x)$ die Fitness des Individuums x. Der Anwender definiert selbst, in welchem Wertebereich die Fitness liegen soll. Es ist üblich, positive reelle Zahlenwerte zu verwenden [2]. Die Fitnessfunktion entspricht der Zielfunktion des Optimierungsproblems. Da im biologischen Kontext stets eine große Fitness im Sinne der Angepasstheit von Vorteil ist, wird meistens von einem Maximierungsproblem ausgegangen. Dann gilt: Je höher die Fitness, desto besser ist die Lösung und gesucht wird das Individuum $x^\star$, sodass $f(x^\star) > f(x)$ für alle $x \in S$. Minimierungsprobleme können aber ebenfalls mit evolutionären Algorithmen gelöst werden.

Die Anfangspopulation wird meistens mithilfe des Zufalls initialisiert, indem zufällige Punkte aus dem Suchraum erzeugt werden. Jeder dieser Punkte entspricht einer Chromosomenbelegung eines Individuums. Wenn Vorwissen zu dem Optimierungsproblem besteht, können

manche Punkte auch gezielt im Suchraum gesetzt werden [2].
Wie viele Individuen eine Population umfasst, entscheidet der Anwender selbst. Bei den Evolutionsstrategien wird die Anzahl der Eltern-Individuen im Allgemeinen mit μ bezeichnet und die Anzahl der zu erzeugenden Nachkommen mit λ. Eine Population umfasst demnach λ Individuen, von denen μ Individuen als Eltern für die nächste Generation ausgewählt werden, wobei $\lambda > \mu$ ist. Die Selektion der Eltern erfolgt nach dem Eliteprinzip. Dies bedeutet, dass die diejenigen μ Individuen, welche die besten Bewertungen durch die Fitnessfunktion erhielten, als Eltern ausgewählt werden. Die restlichen Individuen werden verworfen. Aus dem Eltern-Pool wird zufällig ein Individuum ausgewählt, welches dann einen Nachkommen erzeugt. Dies geschieht solange, bis λ Nachkommen erstellt wurden. Der Zufallsoperator unterliegt dabei der Gleichverteilung, sodass im Durchschnitt alle Eltern-Individuen gleich viele Nachkommen erzeugen. Dabei entsteht ein Nachkomme in der Regel durch Mutation des Eltern-Chromosoms, wobei das Vorgehen später noch genauer erklärt wird.
Man unterscheidet zwei grundlegende Varianten der Evolutionsstrategien:

$(\mu + \lambda)$-Strategie

Nachdem μ Eltern insgesamt λ Nachkommen erzeugt haben, werden die neu generierten Lösungen mit der Fitnessfunktion bewertet. Für die Erstellung der nächsten Generation müssen erneut μ Eltern selektiert werden, wobei für die Auswahl auch die vorherigen Eltern mit einbezogen werden. Es werden demnach die μ besten Individuen aus insgesamt $\mu + \lambda$ Individuen ausgewählt, nämlich aus den Eltern und deren Nachkommen. Deswegen wird diese Variante als $(\mu+\lambda)$-Strategie bezeichnet. Weil die besten Individuen selektiert werden, ist der Vorteil, dass sehr guten Eltern-Individuen nicht verworfen werden, falls die erzeugten Nachkommen vielleicht sogar schlechter sind. Allerdings wird dadurch unter Umständen eher ein lokales Optimum erreicht [2].

(μ, λ)-Strategie

Bei der (μ, λ)-Strategie erzeugen μ Eltern λ Nachkommen. Nur von den Nachkommen werden dann μ Individuen selektiert, um Eltern für die nächste Generation zu werden. Die vorherigen Eltern werden dabei also ignoriert und verworfen. Mit dieser Variante wird eine vorzeitige Konvergenz zu einem lokalen Optimum häufig vermieden.

Bei der (μ, λ)-Strategie ergibt sich der Selektionsdruck durch $\frac{\mu}{\lambda}$ [3] und gibt an, wie stark die Individuen mit einer hohen Fitness bevorzugt werden [5]. Die Zahl $\frac{\mu}{\lambda}$ entspricht dem prozentualen Anteil der als Eltern ausgewählten Individuen. Je kleiner dieser Anteil ist, desto größer ist der Selektionsdruck für die Individuen [3]. Ein sehr niedriger Selektionsdruck tastet den Suchraum sehr breit ab, erschwert jedoch ein Konvergenzverhalten zu der optimalen Lösung [5]. Ein hoher Selektionsdruck sorgt dafür, dass nur die Umgebung sehr weniger Punkte im

Suchraum betrachtet wird und erzielt damit Konvergenz, wobei die Gefahr bestehen kann, dabei ein lokales Optimum zu erreichen. Das Verhältnis von μ und λ wird häufig eins zu sieben gewählt, was bedeutet, dass ein Eltern-Individuum dann im Mittel sieben Nachkommen erzeugt [2]. Der Selektionsdruck liegt damit bei $\frac{1}{7}$.

Mutation

Die Mutation stellt bei Evolutionsstrategien den Hauptoperator der Variation dar und wird für die Erzeugung aller Nachkommen verwendet (sofern keine Rekombination mit einfließen soll). Das Chromosom eines Eltern-Individuums $x = (x_1, x_2, \ldots, x_n)$ wird mutiert und damit ein Nachkomme erzeugt, indem zu jedem Parameter x_i, $i = 1, \ldots, n$, eine normalverteilte Zufallszahl addiert wird. Das heißt: $x_{Nachkomme} = (x_1, x_2, \ldots, x_n) + (z_1, z_2, \ldots, z_n)$, wobei z_i für $i = 1, \ldots, n$, eine generierte Zufallszahl ist, welche der Normalverteilung mit Mittelwert 0 und kleiner Standardabweichung σ unterliegt. Durch die Mutation wird ermöglicht, dass das Chromosom in alle Richtungen im Suchraum verändert werden kann. Die Stärke der Mutation wird durch die Standardabweichung σ bestimmt. Der Wert für σ sollte problemspezifisch und so gewählt werden, dass nach der Mutation noch Ähnlichkeit zum Eltern-Chromosom besteht [2]. Ist σ zu groß, entsteht ein komplett anderes Chromosom, welches einer willkürlichen Chromosomen-Erzeugung entsprechen würde. Bei einem zu kleinen Wert würde die Rechenzeit unter Umständen verlängert werden, bis ein hinreichend gute Lösung evolviert wurde.
Für eine simple Evolutionsstrategie kann σ für alle Individuen konstant gewählt werden. Für manche Probleme bietet es sich aber an, die Mutationsstärke während der Evolution anzupassen. Man spricht dann von einer Mutationsschrittweiten-Adaption. Hierbei unterscheidet man zwei mögliche Ansätze:

deterministische Adaption:
Bei dieser Variante ist die Mutationsstärke σ zu Beginn der Evolution größer, um den Suchraum möglichst breit zu durchsuchen. Im Laufe der Evolution wird die Schrittweite immer kleiner, damit eine Verfeinerung der generierten Lösungen eintritt und ein Konvergenzverhalten erzielt wird, indem nur noch lokal optimiert wird. Man kann die Mutationsstärke nach einem festen Schema anpassen. Wenn $\sigma(t)$ die Mutationsstärke zum Zeitpunkt t in der Evolution ist, dann ist $\sigma(t+1) = \alpha * \sigma(t)$, wobei $0 < \alpha < 1$ [2]. Der Faktor α lässt die Mutationsschrittweite von Generation zu Generation sinken und sollte möglichst nah an 1 gewählt werden.

dynamische Adaption:
Bei der dynamischen Mutationsschrittweiten-Adaption kann σ während der Evolution nicht nur kleiner, sondern auch größer werden. Jedes Individuum erhält dabei eine eigene Mutationsstärke. Es ist auch möglich für jedes der Gene im Chromosom eine eigene Mutations-

schrittweite zu verwenden. Bei Erzeugung eines Nachkommen werden die Mutationsschrittweiten an den Nachkommen vererbt, wobei sie dabei auch leicht verändert werden. Wenn die besten Individuen aufgrund ihrer Fitness selektiert werden, werden dadurch auch indirekt die Mutationsstärken evolviert, da die Individuen mit geeigneteren Mutationsstärken auch bessere Nachkommen erzeugen. Manche Punkte im Suchraum befinden sich vielleicht in einer Umgebung, in der die umliegenden Punkte alle eine sehr ähnliche Fitness haben. Dann kann ein größerer Mutationsschritt erfolgen. Ansonsten sind kleinere Schritte besser, um nicht über das Optimum hinweg zu springen. Es gibt verschiedene Methoden für die dynamische Adaption, auf welche hier nicht weiter eingegangen wird, da diese nicht relevant für die verwendete Evolutionsstrategie in dieser Bachelorarbeit ist. Warum das der Fall ist, wird in Abschnitt 5.3.1 erklärt.

Rekombination

Das Crossover ist bei Evolutionsstrategien ein optionaler genetischer Operator, da ein Crossover lediglich einer Mutation um den Schwerpunkt der Elternindividuen entspricht [3]. Falls ein Crossover angewandt wird, verwendet man in der Regel das uniforme Crossover, was bei den Evolutionsstrategien häufig als diskrete Rekombination bezeichnet wird [2]. Aus zwei Eltern-Chromosomen wird ein Nachkomme erzeugt, indem für jedes Gen zufällig eines der beiden Eltern-Allele ausgewählt wird. Das heißt:
$x = (x_1, x_2, \ldots, x_n)$ und $y = (y_1, y_2, \ldots, y_n)$ erzeugen den Nachkommen $v = (v_1, \ldots, v_n)$, wobei $v_i = x_i$ oder $v_i = y_i$ für $i = 1, \ldots, n$.

Bei den genetischen Operatoren ist darauf zu achten, dass keine Nachkommen erzeugt werden, deren Chromosomen unzulässige Lösungen darstellen. Falls dies auftritt, können zum Beispiel Reperaturmechanismen angewandt werden [2].

Abbruchkriterium

Für evolutionäre Algorithmen im Allgemeinen wird immer ein Abbruchkriterium benötigt. Die Evolution kann zum Beispiel beendet werden,

- wenn eine bestimmte Anzahl von Generationen erreicht wurde.
- wenn eine bestimmte Mindestfitness erreicht wurde.
- wenn seit einer bestimmten Anzahl von Generationen keine Verbesserung mehr stattgefunden hat.

Parameterwerte

Ein evolutionärer Algorithmus bietet einige Freiheiten und Parameter, mit denen experimentiert werden kann. Das gibt dem Anwender die Möglichkeit, den Algorithmus spezifisch

für sein Optimierungsproblem anzupassen. Bevor eine Evolution durchgeführt werden kann, müssen verschiedene Parameter festgelegt werden. Dazu gehören bei einer Evolutionsstrategie in jedem Fall μ und λ sowie die Mutationsschrittweite. Eventuell werden auch noch Parameter für die Mutationsschrittweiten-Adaption und für die Anwendung eines Crossovers benötigt.

3.4 Kompetitive Co-Evolution

Als kompetitive Co-Evolution bezeichnet man die Situation, in der die Fitness eines Individuums nicht nur vom eigenen Genotypen und der Umgebung abhängt, sondern auch von anderen Individuen beeinflusst wird [6]. In der Regel sind dies Individuen einer anderen Population, welche parallel evolviert werden und die Konkurrenten darstellen. Je besser also die Konkurrenten sind, desto schlechter ist die Fitness des zu bewertenden Individuums. Eine Verbesserung der Fitness der einen Population ist immer zum Nachteil für die Fitness der anderen Population [7].
Im Fall des Spiels „The Resistance" konkurrieren Widerstandskämpfer und Spione um den Spielsieg. Die Strategien beider Rollen sollen mithilfe einer Evolutionsstrategie optimiert werden. Dabei sind ihre Gewinnchancen stets auch von der Strategie des Gegners sowie von den eigenen Teammitgliedern abhängig, denn immer drei Widerstandskämpfer und zwei Spione treten in einer fünf-Spieler-Partie gegeneinander an. Da die Co-Evolution also relevant für die Evolution der Spielstrategien ist, wird die Thematik hier eingeführt.

3.4.1 Fitnesslandschaft

Die Fitnessfunktion stellt die Umgebung für die Individuen dar. Würde man alle Fitnesswerte über dem zugehörigen Raum der Genotypen auftragen, so erhält man die Fitnesslandschaft [3]. Für einen zweidimensionalen Suchraum S lässt sich dies gut veranschaulichen. Die X- und Y-Achse entsprechen jeweils einem der beiden Parameter. Auf der Z-Achse lässt sich nun für alle zulässigen Parameterkonstellationen die zugehörige Fitness auftragen. Das Bild einer Landschaft mit Hügeln und Tälern entsteht. Für jeden Punkt im Suchraum ist die Fitness $f : S \to \mathbb{R}$ in der Fitnesslandschaft abgebildet. Ohne Co-Evolution bleibt die Fitnesslandschaft in der kompletten Evolution in der Regel konstant. Mit Co-Evolution, verändert sich die Fitnesslandschaft jedoch dynamisch, da die Fitness eines Individuums stets auch von anderen Individuen abhängt. Eine bestimmte Parameterkonstellation für den Widerstandskämpfer kann also je nach mitspielenden Spionen und den anderen Widerstandskämpfern eine unterschiedliche Fitness liefern. Die konkurrierenden Individuen sind Bestandteil der Umgebung eines Individuums. Im Laufe der Co-Evolution verändern sich auch die Konkurrenten, wodurch eine dynamische Fitnesslandschaft entsteht.

3.4.2 Der Red-Queen-Effekt

Das Verhalten von Räuber und Beute ist ein klassisches Beispiel für die Co-Evolution [6]. Es liegen zwei verschiedene Populationen vor, die gegenseitig Einfluss auf die Fitness der Individuen aus der jeweils anderen Population nehmen. Das Ziel beider Populationen ist es, ihre Strategie zu optimieren. Während der Evolution lernen die Beutetiere, immer schneller zu rennen, um möglichst vielen Räubern zu entkommen. Die Räuber jedoch lernen ebenfalls immer schneller zu sein, um möglichst viele Beutetiere zu fangen. Es beginnt also ein Wettrüsten in der Schnelligkeit: Die Individuen aus Räuber- und Beute-Population werden immer schneller, um sich an ihre Umwelt anzupassen. Wenn man sich die Entwicklung der Fitness von Räuber- und Beutepopulation über die Generationen ansieht, wird man eventuell feststellen, dass diese im Mittel konstant bleibt. Es sieht so aus, als hätten sich weder Beute noch Räuber in der Evolution verbessert. Die Optimierung hat aber trotzdem stattgefunden, man kann diese jedoch nicht anhand der Entwicklung der Fitness nachvollziehen. Die Fitness wird in Abhängigkeit der Konkurrenten gemessen und da diese ebenfalls stets dazu lernen, bleibt die Fitness im Durchschnitt gleich.
In einer Co-Evolution kann trotz konstant bleibender oder sogar fallender Fitness eine Verbesserung der Individuen stattgefunden haben. Dies wird als Red-Queen-Effekt bezeichnet [8] [9]. In Lewis Carroll's Buch „Through the Looking-Glass" sagt die „Red Queen" zu Alice, dass sich die Welt unter ihren Füßen schnell bewegt, sodass sie stets weiter rennen muss, um an der gleichen Position zu bleiben. Weil die Umgebung der Individuen bei einer Co-Evolution stets im Wandel ist, kann oftmals eine konstant bleibende Fitness beobachtet werden, obwohl die Individuen eigentlich permanent „weiter rennen", also sich mit der Evolution verändern. Da die Fitness bei einer Co-Evolution den Lernerfolg also nicht unbedingt widerspiegelt, werden andere Messmethoden benötigt, um den Fortschritt und damit den Red-Queen-Effekt nachzuweisen [7] [10].

3.4.3 Messmethoden

Die besten Individuen aus jeder Generation können im Anschluss an die Co-Evolution verwendet werden, um Daten zu erstellen, die Aufschluss über den Lernprozess geben können. Im Folgenden werden drei Messmethoden vorgestellt, welche im späteren Verlauf dieser Arbeit verwendet werden. Abbildungen zum genaueren Verständnis der jeweiligen Methoden können daher im Kapitel 5.4.2 eingesehen werden.

CIAO

Cliff & Miller schlugen 1995 eine Messmethode vor, mit welcher der Fortschritt in einer Co-Evolution von zwei konkurrierenden Populationen visualisiert werden kann. Die Idee ist, dass ein Individuum aus einer späteren Generation besser sein sollte als die Konkurrenten der

anderen Population aus früheren Generationen [10]. Dann nämlich hat ein Lernfortschritt stattgefunden. Um diesen zu ermitteln, wird das beste Individuum aus Generation g in Abhängigkeit jedes besten Konkurrenten aus den vorherigen Generationen $1, \ldots, g$ getestet. Pro Konkurrent wird eine bestimmte Anzahl an Evaluierungen durchgeführt und darüber die durchschnittliche Fitness ermittelt. Das heißt, für ein Individuum aus Generation g werden auch g Fitness-Werte berechnet (für $g = 1, \ldots, m$, wenn m die Anzahl der Generationen ist). Den entstehenden Datensatz nennen Cliff & Miller CIAO, was für „Current Individual vs. Ancestral Opponents" steht. Zur Visualisierung werden die Daten in einem 2D-Bild angezeigt, wobei in X-Richtung die Generationen der einen Population und in Y-Richtung die der anderen Population aufgetragen werden. Im Beispiel von Räuber und Beute zeigt ein Pixel bei $(50; 100)$, mit welcher Wahrscheinlichkeit das beste Beute-Individuum aus Generation 100 dem besten Räuber aus Generation 50 entkommt. Die Pixel werden in Graustufen angezeigt, das heißt je dunkler ein Pixel ist, desto besser war das Beute-Individuum im Test gegen den Räuber. Je heller er ist, desto siegreicher war der jeweilige Räuber.
Das CIAO-Bild zeigt die Veränderungen der besten Individuen auf Ebene der Phänotypen [10]. In Abbildung 3.3 ist ein idealer Datensatz dargestellt, bei dem alle Beute-Individuen allen Räubern aus den vorherigen Generationen entkommen und bei dem ebenfalls alle Räuber in der Lage ist, alle Beute-Individuen aus den vorherigen Generationen zu fangen [7]. Dies zeigt also, dass die Individuen stets besser waren als die gegnerischen Vorfahren. Um solch einen Lernfortschritt festzustellen, würde in der Graphik eine diagonale Aufteilung zu sehen sein, wobei die eine Seite viele dunkle Pixel enthält und die andere überwiegend helle.

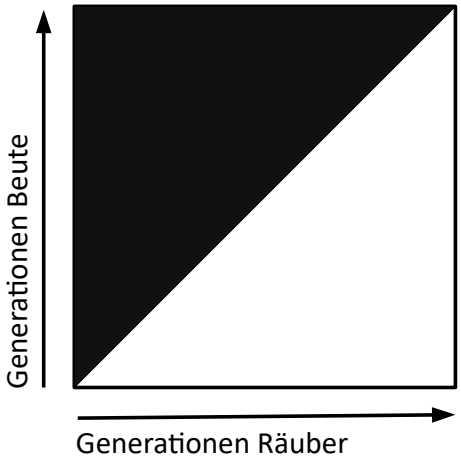


Abbildung 3.3: Darstellung eines idealen CIAO-Datensatzes (in Anlehnung an [7])

Elite Bitmap

Eine Möglichkeit, den Prozess der Evolution auf Ebene der Genotypen nachzuvollziehen, ist die Visualisierung der einzelnen Gene in einer sogenannten Elite Bitmap. Mithilfe der Elite Bitmap können Veränderungen im Genotypen einer Population aufgezeigt werden, welche möglicherweise bestimmte Muster im CIAO-Bild erklären [10]. Diese Methode wurde von Cliff & Miller für genetische Algorithmen vorgeschlagen, für welche die Parameter in der Regel binär kodiert sind und das Chromosom einem Bitstring entspricht. Daher wird die Bezeichnung „Bitmap" verwendet. Die Genotypen des besten Individuums aus jeder Generation (also der Elite) werden in einem 2D-Bild dargestellt. Im Sinne der genetischen Algorithmen ergibt eine Null im Chromosom einen weißen Pixel in der Graphik und eine Eins einen schwarzen Pixel. In X-Richtung werden die Genpositionen des Chromosoms aufgetragen und in Y-Richtung die Generationen. Diese Visualisierung ermöglichst es, die Veränderungen der Genotypen im Laufe der Generationen zu beobachten. Wenn vertikale Bänder zu erkennen sind, so hat sich das jeweilige Allel über mehrere Generationen hinweg bei den besten Individuen erfolgreich durchgesetzt. Signifikante Veränderungen im Genotyp der Elite-Individuen liegen vor, wenn eine neue Bänderung beginnt, aufhört oder durch ein anderes Muster unterbrochen wurde [10].

Master-Tournament

Floreano & Nolfi schlugen 1997 das Master-Tournament als weitere Messmethode des Lernfortschritts vor. Diese Technik wird ebenfalls erst angewandt, wenn die Evolution bereits abgeschlossen ist und die besten Individuen aus allen Generationen vorliegen. Das beste Individuum jeder Generation wird in Abhängigkeit aller besten Gegner aus allen Generationen getestet und daraus eine gemittelte Fitness pro Individuum berechnet [11]. Diese wird als Master-Fitness bezeichnet. Die resultierenden Werte werden in einem Graph als Kurve veranschaulicht, wobei die X-Achse die Generation anzeigt und auf der Y-Achse kann die zugehörige Master-Fitness für das beste Individuum der jeweiligen Generation abgelesen werden. Für jede Population wird eine eigene Kurve erstellt. Im Idealfall wächst die Kurve kontinuierlich an, was einen permanenten Lernfortschritt bedeuten würde [12]. Dann würde man das beste Individuum der gesamten Evolution in den letzten Generation auffinden. Ansonsten findet man die besten Individuen im Graph des Master-Tournaments dort, wo die Kurven beider konkurrierender Populationen die höchste Master-Fitness zeigen [11]. In diesem Fall lag keine kontinuierliche Verbesserung der Lösungen vor.

3.4.4 Vor- und Nachteile von Co-Evolution

Durch kompetitive Co-Evolution und die dadurch entstehende dynamische Fitnesslandschaft kann verhindert werden, dass die Evolution in einem lokalen Optimum verharrt [13]. Jede Population versucht stets besser als die Konkurrenz zu werden, wodurch potenziell immer

wieder neue Herausforderungen für die konkurrierende Population entstehen. Durch dieses Wettrüsten in der Co-Evolution können in manchen Fällen komplexere Lösungen evolviert werden als bei einer einfachen Evolution, bei welcher die Fitnesslandschaft konstant bleibt [7]. Leider ist dies aber nicht garantiert, denn es kann auch passieren, dass die Anpassungen der Individuen in einen immer wiederkehrenden Kreislauf verfallen. Wenn dies der Fall ist, dann werden meistens nur sehr simple Lösungen evolviert. Bei dem Kreisen werden die Lösungen, die zu einem bestimmten Zeitpunkt effektiv gegen die Konkurrenten waren, zu späteren Zeitpunkten in der Evolution wiederentdeckt [7]. Zum Beispiel verfolgt Population A die Strategie A_1, und Population B die Strategie B_1. Angenommen B_1 ist sehr gut gegen A_1. Eine neue Strategie A_2 wird evolviert, welche effektiv gegen B_1 ist. Daraufhin entsteht die Strategie B_2, welche wiederum A_2 schlägt. Es stellt sich heraus, dass die ursprüngliche Strategie A_1 sehr effektiv gegen B_2 ist, also kehrt Population A zu Strategie A_1 zurück und im Folgenden wird auch Population B wieder zu B_1 zurückkehren [7]. Es entsteht ein Kreislauf.

Hall-of-Fame

Eine Möglichkeit, das Kreisen der Lösungen bei der Co-Evolution zu verhindern, kann mithilfe der sogenannten Hall-of-Fame erreicht werden [7]. Rosin & Belew schlugen 1997 vor, das beste Individuum aus jeder Generation in der Hall-of-Fame zu speichern und diese Individuen zum Testen der aktuellen Generation zu verwenden [14]. Das heißt, alle Individuen der aktuellen Generation sollen in Abhängigkeit aller Konkurrenten aus der Hall-of-Fame bewertet werden. Die Fitness ergibt sich aus dem Durchschnitt der Ergebnisse. Da mit jeder Generation ein weiteres Individuum pro Population in die Hall-of-Fame aufgenommen wird, steigt auch die Anzahl an Berechnungsschritten. Es stellte sich aber heraus, dass es effektiv ist, wenn die Individuen nur mit einer vorgegebenen festen Anzahl von Gegnern aus der Hall-of-Fame evaluiert werden [12]. Die Gegner werden dann zufällig aus der Hall-of-Fame ausgewählt. Bei der Bewertung der Anfangspopulation werden die Konkurrenten zufällig aus der gegnerischen Startpopulation ausgewählt. Die Hall-of-Fame-Variante ist manchmal effektiver als die Standard-Co-Evolution, bei der die aktuellen Individuen anhand der Konkurrenten aus der aktuellen Generation bewerten werden [7]. Die Dynamik ist bei der Hall-of-Fame-Methode allerdings nicht so groß, da immer wieder die gleichen Gegner zur Evaluierung verwendet werden. Es kann kein starkes Wettrüsten wie bei der klassischen Co-Evolution einsetzen, wodurch aber das Kreisen der Lösungen verhindert wird. Es werden Lösungen evolviert, die gegen die vergangenen Konkurrenten effektiv sind.

4 Modellierung der Spielstrategien

In diesem Kapitel geht es darum, ein Modell für die Spielstrategien zu erstellen. Hierfür muss zunächst festgelegt werden, mit welcher Art von evolutionären Algorithmen die Strategien später optimiert werden sollen. Dementsprechend kann entschieden werden, welche Parameter das Modell für die Spielstrategien enthalten soll. Dabei wird die Spielstrategie in die Handlungen als Widerstandskämpfer und als Spion unterteilt.

4.1 Vereinfachungen

Simuliert man ein Spiel mit fünf Spielern, in welchem Spione und Widerstandskämpfer jeden Spielzug bei der Teamauswahl, Abstimmung und Mission komplett zufällig durchführen, dann liegt die Gewinnwahrscheinlichkeit für beide Parteien bei 50%. Daher könnte man das Spiel als fair einstufen, wobei in diesem Testfall die Spieler ihr Wissen nicht in ihre Entscheidungen mit einfließen lassen. Wenn „The Resistance" unter Menschen gespielt wird, ist zudem verbale und nonverbale Kommunikation von großer Bedeutung. Da die Spione versuchen müssen, den Anschein zu erwecken, selbst ein Widerstandskämpfer zu sein, werden sie unter Umständen in Diskussionen mit anderen Spielern lügen. Wer die Mitspieler gut kennt oder generell die Mimik und Gestik anderer Menschen sehr gut deuten kann, kann oft leicht sagen, dass ein Spieler lügt. Es ist als Spion also oftmals nicht so leicht, die anderen glaubwürdig zu überzeugen, ein Widerstandskämpfer zu sein. Einige Spieler neigen dazu, als Widerstandskämpfer viel aktiver am Spiel teilzunehmen, da sie in keiner Situation lügen müssen und so besser mit den anderen diskutieren können. Sobald sie allerdings als Spion spielen, werden sie zum eher passiven Beobachter, damit sie nichts falsches sagen oder ausstrahlen, was sie als Spion entlarven könnte. Wer schon mehrere Partien mit diesem Mitspieler gespielt hat, merkt dann sehr schnell, dass die Person höchstwahrscheinlich ein Spion ist.
In einer Partie unter Menschen zählen also nicht nur die logischen Schlussfolgerungen anhand der Spielzüge. Ein Widerstandskämpfer formt seine Meinung über die Identitäten der Mitspieler auch anhand dessen, was sie sagen, sowie deren Mimik und Gestik. Dadurch bekommen sie eine weitere Möglichkeit, Spione zu identifizieren. Vernachlässigt man die Komponente der Kommunikation und verwendet nur die aus den Spielrunden erhaltenen Informationen, ist es für den Widerstandskämpfer durchaus schwieriger zu gewinnen. Der Spion hingegen hat zu Beginn des Spiels bereits alle Informationen, die er benötigt und daher

keinen so großen Nachteil in einem Spiel ohne Kommunikation.
Das in dieser Bachelorarbeit simulierte Spiel verzichtet aus Vereinfachungsgründen auf die Kommunikation zwischen den Spielern und konzentriert sich auf die reinen Spielzüge. Besonders interessant ist es daher für den Widerstandskämpfer, eine gute Strategie zu finden. Des Weiteren wird lediglich das Spiel mit fünf Spielern betrachtet, bei welchem es drei Widerstandskämpfer und zwei Spione gibt.

4.2 Modell-Ideen

Die Aufgabe der Evolution von Spielstrategien für das Spiel „The Resistance" kann auf ganz verschiedene Arten bearbeitet werden. Eine Möglichkeit ist es, mit einem genetischen Algorithmus binäre Zeichenketten zu evolvieren. Die Idee bietet sich an, da die im Spiel zu treffenden Entscheidungen bei der Abstimmung und Sabotage jeweils nur zwei Möglichkeiten umfassen: Wählt der Spieler für das Team oder nicht? Sabotiert der Spion oder nicht? Man kann jeweils eine binäre Zeichenkette für die Strategie des Widerstandskämpfer und des Spions verwenden, wobei die Nullen und Einsen im Chromosom die Antwort auf eine Entscheidung in einer bestimmten Spielsituation darstellen. Null repräsentiert die Antwort „Nein" und Eins repräsentiert „Ja". In der Evolution sollen dann die Belegungen der einzelnen Gene gelernt werden. Mit dieser Herangehensweise ist es jedoch nicht möglich, für den Widerstandskämpfer eine brauchbare Repräsentation der Strategie zu entwerfen. Aus kombinatorischen Gründen funktioniert es nicht, die bereits erworbenen Informationen über den kompletten Spielverlauf hinweg zu erhalten, ohne dabei ein Chromosom mit mehreren Millionen Genen zu benötigen. Man stelle sich die folgende Situation vor: Der Widerstandskämpfer erfährt, welche Spieler in welcher Mission als Teammitglieder teilnahmen und wie viele Sabotagen in der jeweiligen Mission auftraten. Anhand dieser Informationen soll er kombinieren und schlussfolgern, wie er für das aktuelle Team abstimmt. Insgesamt gibt es für die Erstellung eines Teams der Größe zwei und drei jeweils zehn mögliche Spielerkonstellationen, da $\binom{5}{3} = \binom{5}{2} = 10$. Für die erste Mission liegen noch keine Spielinformationen vor. Für die zweite Mission ist bekannt, welches Team die erste Mission antrat und mit welchem Ergebnis (keine, eine oder zwei Sabotagen). Unter Berücksichtigung des Teams und des Ergebnisses aus der ersten Mission kann für das in Mission zwei zur Wahl stehende Team entschieden werden, ob es unterstützt oder abgelehnt werden soll. Für alle möglichen Konstellationen aus der ersten Mission, das heißt für $3 * 10 = 30$ Möglichkeiten, soll für alle zehn möglichen aktuellen Teamkonstellationen eine Entscheidung getroffen werden. Allein für die zweite Mission würden dadurch $3 * 10 * 10 = 300$ Gene benötigt werden. Für alle 300 Konstellationen gibt es jeweils ein Gen, welches den Spielzug für die Situation mit der Belegung Null oder Eins entscheidet. Für die folgenden Missionen, müssen alle vorherigen Missionen mitberücksichtigt werden. Das heißt, bei Mission fünf gäbe es 30^4 Kombinationen für sämtliche vorher eingetroffenen Spielsituationen und daher $30^4 * 10 = 8.1$ Millionen

Möglichkeiten abzuwägen, wie für das aktuelle Team entschieden werden soll. Jegliche Kombinationen für alle Spielsituationen in der Zeichenkette zu berücksichtigen, würde demnach mehrere Millionen Gene im Chromosom des Widerstandskämpfers erfordern. Dies wäre in der Implementierung schwer umzusetzen und es würde sehr lange dauern, bis die Evolution zu einem guten Ergebnis führt. Daher ist dieser Ansatz für den Widerstandskämpfer ungeeignet.
Für die Widerstandskämpfer ist aber sehr wichtig, logische Schlussfolgerungen aus dem gesamten Spielverlauf zu ziehen und die erhaltenden Informationen zu kombinieren und zu verwenden, um die Spione ausfindig zu machen. Nur dann haben sie eine Chance zu gewinnen.

Wesentlich erfolgversprechender ist der Ansatz, bei dem die Widerstandskämpfer ihre Mitspieler nach ihrer Vertrauenswürdigkeit bewerten. Hierfür legt jeder Widerstandskämpfer für sich eine Buchführung an, in der für jeden Spieler ein Score eingetragen wird. Der Score eines Spielers soll aussagen, wie verdächtig dieser ist, ein Spion zu sein. Anhand der Geschehnisse im Spiel wird der Score eines Spieler nach jeder Mission aktualisiert, indem situationsabhängige Parameterwerte hinzuaddiert oder abgezogen werden. Diejenigen zwei Spieler mit den höchsten Scores sind in den Augen des Widerstandskämpfers am wahrscheinlichsten die Spione und damit in den Missionsteams nicht erwünscht. Mithilfe einer Evolutionsstrategie soll die Parameterwahl, welche in diesem Kontext als Bewertungsschema aufgefasst werden kann, optimiert werden. Es soll gelernt werden, welche Spielzüge wie stark belohnt oder bestraft werden müssen, um die Spione möglichst sicher zu identifizieren und dadurch den Spielsieg herbeizuführen.
Da der Spion alle Spielerdidentitäten kennt, erhält dieser keine Buchführung. Der Spion soll ein möglichst effizientes Verhalten lernen, um nicht so schnell als Spion entlarvt zu werden. Um dies zu erreichen, werden für den Spion Wahrscheinlichkeiten evolviert, mit welchen er in gewissen Situationen über Teamauswahl, Abstimmung und Sabotage entscheidet, um seiner Gruppe zum Sieg zu verhelfen.
Diese Idee wird in diesem Kapitel weiterhin verfolgt und ein entsprechendes Modell erstellt. Mithilfe einer Evolutionsstrategie werden im Kapitel 5 die Spielstrategien anschließend optimiert.

4.3 Spielumgebung

In der Simulation am Computer können ohne die Spielumgebung keine Spieler mit ihren Strategien gegeneinander antreten. Das Spiel selbst stellt das Grundgerüst für das Modell der Spielstrategien dar. Die Implementation des Spiels erfolgt nach den Spielregeln aus Kapitel 2.
Im Algorithmus 1 kann der komplette Spielablauf noch einmal nachvollzogen werden.

Algorithmus 1 : Spiel

```
Function Spiel(Spieler_1, Spieler_2, Spieler_3, Spieler_4, Spieler_5):
    verteile die Rollen 3 x Wiederstandskämpfer und 2 x Spion zufällig an die Spieler
    Missionsleiter ← Spieler_1
    sabotierte Missionen ← 0
    erfolgreiche Missionen ← 0
    while Spiel noch nicht vorbei                          // nächste Mission antreten
        Wahlversuch ← 1
        Ja-Stimmen ← 0
        while Ja-Stimmen < 3 und Wahlversuch ≤ 5
            Team ← Teamauswahl durch Missionsleiter
            for i ← 1 bis 5
                Spieler_i stimmt ab
            end for
            Ja-Stimmen ← Ergebnis der Abstimmung
            if Ja-Stimmen < 3
                Wahlversuch ← Wahlversuch + 1
                Missionsleiter ← nächster Spieler im Uhrzeigersinn
            end if
        end while
        if Wahlversuch = 6          // nach 5. Wahlversuch keine Wahlmehrheit
            Spiel ist vorbei: Spione haben gewonnen
        else                                                // Wahlmehrheit erzielt
            for each Mitglied im Team
                Mitglied entscheidet, ob es die Mission sabotiert      // falls Spion
            end for
            if Mission wurde sabotiert
                sabotierte Missionen ← sabotierte Missionen + 1
            else
                erfolgreiche Missionen ← erfolgreiche Missionen + 1
            end if
            if erfolgreiche Missionen = 3 oder sabotierte Missionen = 3
                Spiel ist vorbei
            else
                Widerstandskämpfer aktualisieren ihre Buchführung
                Missionsleiter ← nächster Spieler im Uhrzeigersinn
            end if
        end if
    end while
end
```

Das Spiel erfordert eine Eingabe von fünf Spielern. Was sich hinter den Spielern verbirgt, sind lediglich die Parameter ihrer Spielstrategie. Funktionen für die Teamauswahl, Abstimmung und Sabotage verwenden dann die Parameter des Spielers, um den Spielzug auszuführen, welcher nach der Strategie vorgesehen ist. Genauer gesagt, besteht jeder Spieler aus zwei Parametersätzen: einen für den Widerstandskämpfer und einen für den Spion. Je nach zugewiesener Rolle, erfährt der Spieler, welche Parameter und Funktionen er im Spiel verwenden muss. Die Parameter des Spion sind Wahrscheinlichkeiten, welche bei der Teamauswahl, Abstimmung und Sabotage bei den Entscheidungen mit einfließen. Für den Widerstandskämpfer entspricht die Wahl der Parameter einem Bewertungsschema, anhand welchem die Buchführung nach jeder Mission aktualisiert wird.
Jeder Spieler erhält einen Index i, $i = 1, \ldots, 5$. Man kann es sich so vorstellen, dass $Spieler_1$ bis $Spieler_5$ in dem simulierten Spiel in einem Kreis sitzen. $Spieler_{i+1}$ sitzt im Uhrzeigersinn betrachtet links neben $Spieler_i$ für $i = 1, \ldots, 4$ und $Spieler_1$ sitzt wiederum links von $Spieler_5$. Die Indizes geben also den Sitzplatz an und damit auch die Reihenfolge, mit welcher das Amt des Missionsleiters weitergegeben wird. $Spieler_1$ beginnt stets als Missionsleiter. Wenn mit denselben fünf Spielern mehrere Spieldurchläufe durchgeführt werden, sollten die Spieler-Indizes zunächst zufällig permutiert werden, damit die Reihenfolge verändert wird und nicht immer derselbe Spieler als Missionsleiter beginnt.

4.4 Modell und Parameter

Eine Evolutionsstrategie eignet sich besonders dann, wenn die zu optimierenden Parameter aus dem Bereich der reellen Zahlen stammen. Für die Evolution der Spielstrategien mit dem hier vorgestellten Modellansatz liegen die zu optimierenden Parameter für die Spione im Intervall $[0, 1]$. Die Bewertungs-Parameter der Widerstandskämpfer können ebenfalls als reelle Zahlen gewählt werden.
Ein Teil der Spielstrategie wird bereits fest vorgegeben. Nur dort wo die Parameter einfließen, kann die Spielstrategie modifiziert werden. Im Folgenden werden die vorgegebenen Anteile der Strategie und die verwendeten Parameter für beide Rollen erklärt.

4.4.1 Widerstandskämpfer

Für den Widerstandskämpfer gibt es insgesamt 14 Parameter. Das heißt, dass das Chromosom für ein Widerstandskämpfer-Individuum in der Evolutionsstrategie aus 14 Genen besteht. Jeder dieser Parameter darf beliebige positive und negative reelle Zahlenwerte annehmen. Anhand der Parameterwerte bewertet der Widerstandskämpfer seine Mitspieler für ihre Handlungen als Missionsleiter und bei der Abstimmung. Die Mitspieler, welche Teil des Missionsteams waren, werden außerdem je nach Resultat der Mission bewertet. Für verschiedene Spielszenarien werden unterschiedliche Parameter zur Bewertung der Spieler

herangezogen. Nach jeder Mission aktualisiert der Widerstandskämpfer seine Buchführung, indem der jeweilige Parameterwert zum bisherigen Score des Spielers hinzuaddiert wird. Da die beiden Spieler mit den höchsten Scores von dem Widerstandskämpfer als potenzielle Spione angesehen werden, werden die Spieler durch positive Parameterwerte bestraft und durch negative für ihre Handlungen belohnt. In der Evolution müssen die Vorzeichen und das Verhältnis der Parameterwerte gelernt werden.
Die Buchführung ist in diesem Modell für den Widerstandskämpfer entscheidend, um die Spione ausfindig zu machen. Die Handlungen des Widerstandskämpfers orientieren sich größtenteils an den Scores der Spieler. Die Buchführung entscheidet darüber, welches Team er zusammenstellt und unter anderem auch, wie er für ein Team abstimmt. Er versucht stets zu verhindern, dass beiden Spieler mit den höchsten Scores Teammitglieder für die Mission sind.

Teamauswahl

Der Widerstandskämpfer wählt das Team nach Algorithmus 2. Die vertrauenswürdigsten Spieler sind stets diejenigen, welche die niedrigsten Scores in der Buchführung besitzen. Demnach werden die k Spieler mit den niedrigsten Scores als Teammitglieder ausgewählt. Da der eigene Score des Widerstandskämpfers stets der kleinste Wert in der Buchführung ist und bleibt (wird im Abschnitt *Buchführung* erläutert), wird er als Missionsleiter auch immer sich selbst für das Team aufstellen.
In der ersten Mission besitzen alle Mitspieler den Initialisierungswert 0 in der Buchführung und daher wird eine zufällige Auswahl getroffen.

Algorithmus 2 : Widerstandskämpfer Teamauswahl

```
k ← Anzahl benötigter Teammitglieder          // 2 oder 3, abh. von Mission
sortiere die Spieler in der Buchführung nach aufsteigenden Scores
if Mission = 1
    wähle dich selbst als Teammitglied aus
    wähle zufällig weitere k − 1 Teammitglieder von allen Mitspielern aus
else                                          // bei allen anderen Missionen
    wähle die ersten k Spieler in der Buchführung als Teammitglieder aus
end if
```

Abstimmung

Das Verhalten des Widerstandskämpfers bei der Abstimmung erfolgt nach Algorithmus 3. Bevor der Widerstandskämpfer bei seiner Entscheidung die Scores der Spieler berücksichtigt, werden drei Fälle betrachtet. Wenn der Spieler selbst der Missionsleiter ist, dann hat

er das Team nach seiner Vorstellung erstellt und kann ohne Bedenken „Ja" wählen. Bei Scheitern des fünften Wahlversuches würden die Spione sofort gewinnen. Dies muss verhindert werden und demnach stimmt der Widerstandskämpfer mit „Ja". Bei der ersten Mission wird das Team immer unterstützt. Da für alle Spieler noch der Initialisierungs-Score in der Buchführung steht, ist es nicht sinnvoll bereits Spieler ausschließen zu wollen. Tritt keiner der eben beschriebenen Fälle ein, entscheidet der Widerstandskämpfer nach den Scores der vorgeschlagenen Teammitglieder. Er stimmt nur mit „Ja", wenn keiner der beiden Spieler mit den höchsten Scores im Team ist. Der Widerstandskämpfer, um dessen Buchführung es sich handelt, besitzt den niedrigsten Score in der Buchführung (siehe Abschnitt *Buchführung*). Dadurch stimmt der Widerstandskämpfer stets mit "Nein", wenn das Team aus drei Mitgliedern besteht, der Widerstandskämpfer selbst aber nicht im Team ist. In diesem Fall muss mindestens ein Spion im Team sein, weil es insgesamt nur drei Widerstandskämpfer gibt.
Es ist wichtig, die Reihenfolge der Fälle im Algorithmus 3 zu beachten. Falls also die Situation auftritt, dass ein fünfter Wahlversuch stattfindet und ein Team vorgeschlagen wird, in welchem der Widerstandskämpfer einen Spion vermutet, dann wird die Entscheidung anhand des fünften Wahlversuches priorisiert und trotzdem mit „Ja" gestimmt.

Algorithmus 3 : Widerstandskämpfer Abstimmung

```
sortiere die Spieler in der Buchführung nach aufsteigenden Werten
if selbst der Missionsleiter
    wähle „Ja"
else if Wahlversuch = 5
    wähle „Ja"
else if Mission = 1
    wähle „Ja"
else
    if die zwei Spieler mit den höchsten Scores sind nicht im Team
        wähle „Ja"
    else
        wähle „Nein"
    end if
end if
```

Da ein Widerstandskämpfer nie sabotiert, ist eine Funktion für das Antreten der Mission als Widerstandskämpfer nicht nötig.

Buchführung

Zu Beginn einer Spielpartie weist der Widerstandskämpfer zunächst einmal jedem Mitspieler einen Score von Null zu, da er zu diesem Zeitpunkt noch keine Informationen besitzt. Seinen eigenen Score initialisiert der Widerstandskämpfer mit einem negativen Wert, welcher im Betrag sehr hoch ist, um sich als eindeutiger Widerstandskämpfer von den Mitspielern abzuheben. In seiner Buchführung wird jedem Spieler anhand ihrer Indizes der entsprechenden Score zugeordnet. In Tabelle 4.1 ist die Initialisierung aus Sicht des Spielers mit Index $i = 2$ beispielhaft dargestellt. Wichtig zu betonen ist noch einmal, dass nur Spieler mit der Rolle als Widerstandskämpfer eine Buchführung anlegen.

Spieler-Index	1	2	3	4	5
Score	0	-1000	0	0	0

Tabelle 4.1: Beispielhafte Buchführung aus Perspektive von $Spieler_2$: Initialisierung der Scores zu Spielbeginn für alle Spieler

Im Folgenden wird beschrieben, wie die Buchführung eines Widerstandskämpfers nach jeder Mission aktualisiert wird. Dabei werden Parameterwerte r_j $(j = 0, \ldots, 13)$ zu dem Score eines Spielers hinzugefügt, wenn sich der Spieler in der beschriebenen Situation befand. Es kann vorkommen, dass ein, kein oder mehrere Parameterwerte zu dem bisherigen Score hinzuaddiert werden. Der Spieler, dem die Buchführung gehört, kann seinen eigenen Score ebenfalls aktualisieren. Wenn der Initialisierungswert ausreichend niedrig gewählt wird, behält der Spieler im gesamten Spielverlauf den niedrigsten Wert in der Buchführung bei.

Missionsteam:
Die Mitglieder eines Missionsteams werden je nach Resultat der Mission mit einem bestimmten Parameter bewertet. In Tabelle 4.2 sind die Parameter und die zugehörigen Spielszenarien aufgelistet. Unterschieden wird nach Teamgröße, Anzahl der Sabotagen und ob der Spieler, um dessen Buchführung es sich handelt (im Folgenden als Buchführungs-Spieler bezeichnet), ebenfalls Teil des Teams war.
Der Parameter r_0 wird zum Score der Teammitglieder hinzuaddiert, wenn die Mission mit einem Team der Größe zwei durch einen Spieler sabotiert wurde. Der Buchführungs-Spieler selbst war kein Mitglied dieses Teams. Mindestens einer der beiden Teammitglieder ist ein Spion. Bei einem Team der Größe drei, in dem der Buchführungs-Spieler selbst ein Mitglied war und eine Sabotage auftrat, muss ebenfalls mindestens einer der beiden anderen Teammitglieder ein Spion sein. Der Buchführungs-Spieler ist schließlich ein Widerstandskämpfer und sabotiert nie. Die anderen beiden Teammitglieder werden also auch hier mit dem Parameter r_0 bewertet.
Im Fall für den Parameter r_6 gab es ein Team der Größe drei, was die Mission ohne Sabotagen abschließt. Der Buchführungs-Spieler war nicht in diesem Team. Da es aber nur

Parameter	Teamgröße	inkl. Spieler selbst	#Sabotagen	Bemerkung
r_0	2	nein	1	mind. ein Spion
	3	ja	1	unter zwei Spielern
r_1	3	nein	1	
r_2	3	nein	2	
r_3	2	ja	0	
r_4	2	nein	0	
r_5	3	ja	0	
r_6	3	nein	0	mind. ein Spion im Team

Tabelle 4.2: Situationsabhängige Bewertung der Mitglieder des Missionsteams

drei Widerstandskämpfer gibt und der Buchführungs-Spieler einer von ihnen ist, muss also mindestens ein Spion in dem Missionsteam sein und die Bewertung erolgt mit Parameter r_6. Die Situationen der Parameter r_1 bis r_5 erfordern keine weiteren Erklärungen.

Drei weitere mögliche Kombinationen sind in Tabelle 4.2 nicht dargestellt. Es handelt sich um Situationen, in denen der Buchführungs-Spieler mit hundertprozentiger Sicherheit einen Spion identifizieren kann. In diesen Fällen werden die Scores der jeweiligen Teammitglieder nicht mit einem Parameterwert verrechnet. Stattdessen wird der Score auf einen fixen hohen Wert gesetzt. Der Wert sollte so hoch sein, dass sich der Score des Spielers stets deutlich von den anderen abhebt und immer der höchste Score in der Buchführung bleibt. Die Situationen können in Tabelle 4.3 nachvollzogen werden.

	Teamgröße	inkl. Spieler selbst	#Sabotagen	Bemerkung
Score $\leftarrow +1000$	2	ja	1	ein Spion identifiziert
Score $\leftarrow +1000$	2	nein	2	zwei Spione identifiziert
Score $\leftarrow +1000$	3	ja	2	zwei Spione identifiziert

Tabelle 4.3: Sonderfälle: Spion wird eindeutig identifiziert

Im Fall des Parameters r_2 kann außerdem mit hundertprozentiger Sicherheit ein anderer Widerstandskämpfer identifiziert werden. Das Team besteht aus drei Spielern und zwei von ihnen sabotieren die Mission. Demnach steht fest, dass beide Spione in dem Team waren. Der Buchführungs-Spieler gehörte nicht zu diesem Team. Er weiß nun zwar nicht, wer genau die beiden Spione sind, aber er weiß, dass der verbleibende Spieler, welcher ebenfalls nicht im Team war, ein Widerstandskämpfer sein muss. Der Score des identifizierten Widerstandskämpfers wird auf einen hohen negativen Wert gesetzt.

Missionsleiter:
Derjenige Spieler, der das Team erstellt hat, wird mit einem der Parameter r_7 bis r_9 bewertet.

In der ersten Mission wird allerdings auf die Bewertung des Missionsleiters verzichtet, da die Widerstandskämpfer zu diesem Zeitpunkt noch keine Informationen hatten und das zweite Teammitglied zufällig aus ihren Mitspielern auswählen (siehe Algorithmus 2). In Tabelle 4.4 sind die möglichen Szenarien für den Missionsleiter mit den zugehörigen Parametern erklärt.

Parameter	Erfolgreiche Mission?	Bemerkung
r_7	ja	Sonderfall: Teamgröße 3, ohne den Spieler selbst $\Rightarrow$ mindestens ein Spion im Team
r_8	ja	
r_9	nein	

Tabelle 4.4: Situationsabhängige Bewertung des Missionsleiters

In dem Fall für Parameter r_7 war der Buchführungs-Spieler nicht im Team, welches aus drei Mitgliedern bestand. In dem Missionsteam muss sich demnach mindestens ein Spion befinden. Die Mission wurde aber nicht sabotiert. Trotzdem hat der Missionsleiter ein Team zusammengestellt, in dem sich ein Spion befand und wird als Konsequenz mit dem Parameter r_7 bewertet. Ob der Leiter hierfür nun bestraft oder trotzdem belohnt werden sollte, wird in der Evolution gelernt. Ist dieser Sonderfall nicht eingetreten, dann wird einer der beiden Parameter verwendet. Für Parameter r_8 und r_9 ist nur das Ergebnis der Mission entscheidend. Der Wert r_8 wird verwendet, wenn es sich demnach nicht um den Sonderfall von Parameter r_7 handelt.

Wahlergebnis:
Zur Vereinfachung werden nur die Wahlen betrachtet, bei welchen mit einer Mehrheit für das Team gestimmt wurde und das Team zugelassen wurde, die Mission anzutreten. Spieler, die mit „Nein" gewählt haben, gehörten demnach zu der Minderheit und werden nach den Kriterien aus Tabelle 4.5 bewertet.

Parameter	Erfolgreiche Mission?	5. Wahlversuch	Bemerkung
r_{10}	egal	ja	5. Wahlversuch abgelehnt
r_{11}	ja	nein	erfolgreiches Team abgelehnt
r_{12}	nein	nein	Team mit Spion abgelehnt

Tabelle 4.5: Bewertung der Spieler, die mit „Nein" gestimmt haben

Der fünfte Wahlversuch wird nun auch zur Bewertung herangezogen. Spieler, welche beim fünften Wahlversuch „Nein" wählen, werden mit Parameter r_{10} bewertet. Widerstandskämpfer würden diese Wahl niemals ablehnen, da ansonsten der Sieg für die Spione folgen könnte. Für alle anderen Wahlversuche gilt, dass die Spieler entweder bewertet werden, weil sie ein erfolgreiches Team ablehnten (r_{11}) oder weil sie ein Team ablehnten, welches für eine sabotierte Mission verantwortlich ist (r_{12}). Derjenige Spieler, der ein erfolgreiches Team ablehnt,

ist vielleicht ein Spion und wählt „Nein", weil sich keine Spione im Team befinden. Wer gegen ein Team gestimmt hat, dass die Mission sabotierte, könnte ein Widerstandskämpfer sein, der bereits sehr nützliche Informationen gesammelt hat und in seiner Buchführung richtige Verdächtigungen dokumentiert hat.

Ein weiterer Fall bei den Wahlergebnissen ist, dass ein Mitspieler für ein Team der Größe drei stimmt, er selbst jedoch nicht Mitglied dieses Teams ist. Wichtig zu bemerken ist aber, dass es nur drei Widerstandskämpfer insgesamt im Spiel gibt. Ein Widerstandskämpfer würde solch ein Team also ablehnen, da er nicht selbst im Team ist und demnach mindestens ein Spion im Team sein muss. Derjenige Spieler wird mit Parameter r_{13} bewertet.

Parameter	Erfolgreiche Mission?	5. Wahlversuch	Bemerkung
r_{13}	egal	nein	für ein Team der Größe drei gewählt, ohne Mitglied dieses Teams zu sein

Tabelle 4.6: Bewertung der Spieler, die für ein Team der Größe drei mit „Ja" gestimmt haben, obwohl sie selbst nicht Mitglied dieses Teams sind

Beispiel:
Angenommen folgendes Szenario tritt in der zweiten Mission im zweiten Wahlversuch ein: Der Missionleiter ist $Spieler_3$ und dieser erstellt das Team bestehend aus sich selbst und $Spieler_1$ und $Spieler_4$. Die Wahl erreicht eine Wahlmehrheit, lediglich $Spieler_2$ und $Spieler_4$ haben die Wahl abgelehnt. Das Resultat der Mission ist, dass eine Sabotage durchgeführt wird. Aus Sicht von $Spieler_2$, welcher ein Widerstandskämpfer ist, wird die Buchführung wie in Tabelle 4.7 aktualisiert. Zu bemerken ist, dass der Score von $Spieler_2$ in der Implementierung ebenfalls verändert wird, hier aber zur besseren Übersicht bei -1000 bleibt, um zu verdeutlichen, dass dies nach wie vor der kleinste Score in der Buchführung ist.

Spieler	1	2	3	4	5
Score	$Score^1_{alt} + r_1$	-1000	$Score^3_{alt} + r_1 + r_9$	$Score^4_{alt} + r_1 + r_{12}$	$Score^5_{alt} + r_{13}$

Tabelle 4.7: Buchführung aus Perspektive von $Spieler_2$: Aktualisierung der Scores nach der zweiten Mission

Man könnte vermuten, dass r_3, r_4, r_5, r_8 und r_{12} zur Belohnung eingesetzt werden und die restlichen Parameter zur Bestrafung. Später wird man aber sehen, dass zum Teil andere Vorzeichen besser geeignet sind.

4.4.2 Spion

Für den Spion gibt es insgesamt zehn Parameter, welche nur die Werte aus dem Intervall $[0, 1]$ annehmen können, da es sich um Wahrscheinlichkeiten handelt. In bestimmten Situationen ist das Verhalten des Spions nicht fest vorgegeben, sondern wird mit der Wahrscheinlichkeit s_l $(l = 1, \ldots, 10)$ ausgeführt. Zum besseren Verständnis werden die Parameter direkt im Kontext der Funktionen Teamauswahl, Abstimmung und Sabotage erläutert.

Teamauswahl

Die Teamauswahl des Spions wird im Algorithmus 4 verdeutlicht.

Algorithmus 4 : Spion Teamauswahl

$k \leftarrow$ Anzahl benötigter Teammitglieder `// 2 oder 3, abh. von Mission`
wähle dich selbst als Teammitglied aus
`// Welche Mitspieler kommen als weitere Teammitglieder in Frage?`
if *Teamgröße* $= 2$
  mit Wahrscheinlichkeit s_0: alle anderen Mitspieler
  mit Wahrscheinlichkeit $1 - s_0$: nur die Widerstandskämpfer
else `// Teamgröße` $= 3$
  mit Wahrscheinlichkeit s_1: alle anderen Mitspieler
  mit Wahrscheinlichkeit $1 - s_1$: nur die Widerstandskämpfer
end if
wähle zufällig $k - 1$ Teammitglieder aus den in Frage kommenden Mitspielern aus

Der Spion wählt immer sich selbst in das Team. Welche Spieler als weitere Teammitglieder in Frage kommen, wird nach der Teamgröße unterschieden. Je nach den Wahrscheinlichkeits-Parametern s_0 oder s_1 bevorzugt der Spion, entweder den zweiten Spion als weiteres Teammitglied auszuschließen oder ihn als solches zuzulassen. Manchmal könnte es vielleicht von Vorteil sein, einen weiteren Spion mit in das Team aufzunehmen. Wenn keiner der beiden Spione sabotiert, erlangen beide vielleicht mehr Vertrauen bei den Widerstandskämpfern. Zwei Spione im Team bringt aber auch die Gefahr mit sich, dass vielleicht beide Spione auf die Idee kommen, die Mission zu sabotieren, was sie schnell entlarven würde. Vor allem bei einem Team der Größe zwei hätte dies schlechte Konsequenzen, da dann alle Widerstandskämpfer die Spione eindeutig identifizieren können. Um diese Situation zu umgehen, könnte der Spion auch nur Widerstandskämpfer als weitere Teammitglieder ernennen. Er erhält durch die Wahrscheinlichkeiten die Möglichkeit, abzuwägen, welche Variante er wie stark bevorzugt.

Abstimmung

Das Abstimmungsverhalten des Spions erfolgt nach Algorithmus 5.

Algorithmus 5 : Spion Abstimmung

if *selbst der Missionsleiter*
  wähle „Ja"
else if *Punkte für Widerstandskämpfer* $= 2$
  if *Spione im Team* > 0
    wähle „Ja"
  else
    wähle „Nein"
  end if
else if *Punkte für Spione* $= 2$ ***und*** *Spione im Team* > 0
  wähle „Ja"
else if *Mission* $= 1$
  wähle „Ja"
else if *Wahlversuch* $= 5$
  wähle „Ja" mit Wahrscheinlichkeit s_2
  wähle „Nein" mit Wahrscheinlichkeit $1 - s_2$
else if *Spione im Team* $= 1$
  wähle „Ja" mit Wahrscheinlichkeit s_3
  wähle „Nein" mit Wahrscheinlichkeit $1 - s_3$
else if *Spione im Team* $= 2$
  wähle „Ja" mit Wahrscheinlichkeit s_4
  wähle „Nein" mit Wahrscheinlichkeit $1 - s_4$
else `// keine Spione im Team`
  wähle „Ja" mit Wahrscheinlichkeit s_5
  wähle „Nein" mit Wahrscheinlichkeit $1 - s_5$
end if

Für manche Situationen ist die Entscheidung bereits fest vorgegeben, in anderen werden die Parameter s_2 bis s_5 verwendet. Die Reihenfolge der Fälle ist wichtig zu beachten, um manche Situationen vor anderen zu priorisieren. Wenn bereits zwei erfolgreiche Missionen durchgeführt wurden, fehlt den Widerstandskämpfern nur noch eine gelungene Mission, um zu gewinnen. In diesem Fall muss der Spion alles daran setzen, dies zu verhindern. Nur wenn mindestens ein Spion im Team ist, wählt er mit „Ja" und ansonsten mit „Nein". Ist die Situation umgekehrt und die Spione haben bereits zwei Punkte durch sabotierte Missionen erzielt, dann wählt der Spion auch mit „Ja", wenn mindestens ein Spion im Team ist. Die

aktuelle Mission kann dann nämlich zum Sieg der Spione führen. Ist in dieser Situation aber kein Spion im vorgeschlagenen Team, dann wählt der Spion nicht automatisch „Nein". Die Widerstandskämpfer werden in der aktuellen Mission noch nicht gewinnen können, da sie bisher weniger als zwei Punkte haben.
Wenn die Wahl bereits zum fünften mal durchgeführt wird, wählt der Spion mit der Wahrscheinlichkeit s_2 für das Team. Dies ist unabhängig von der Anzahl der Spione im Team. Bei den Wahrscheinlichkeits-Parametern s_2 bis s_5 gilt immer, dass ansonsten mit der entsprechenden Gegenwahrscheinlichkeit mit „Nein" gestimmt wird. Parameter s_3 gibt vor, mit welcher Wahrscheinlichkeit ein Team mit einem Spion als Mitglied unterstützt wird und s_4 wird analog für ein Team mit zwei Spionen verwendet. Sollte keine der Situationen eintreffen, wird mit Parameter s_5 über das Abstimmungsverhalten entschieden. Dann gibt es keinen Spion im Team und man würde vermuten, dass in der Evolution hier gelernt wird, eher mit „Nein" zu stimmen, was auch geschieht, wie man später sehen wird.

Sabotage

Bei der Entscheidung, ob der Spion sabotiert oder nicht, wird nach den Situationen im Algorithmus 6 vorgegangen.

Algorithmus 6 : Spion Sabotage

```
if Punkte für Spione = 2 oder Punkte für Widerstandskämpfer = 2
    sabotiere
else if Anzahl Spione im Team = 2 und Teamgröße = 2
    sabotiere mit Wahrscheinlichkeit s6
else if Anzahl Spione im Team = 2 und Teamgröße = 3
    sabotiere mit Wahrscheinlichkeit s7
else if Mission = 1                          // als einziger Spion im Team
    sabotiere mit Wahrscheinlichkeit s8
else                                         // als einziger Spion im Team
    sabotiere mit Wahrscheinlichkeit s9
end if
```

Wenn die Widerstandskämpfer oder die Spione (oder beide) bereits zwei Punkte im Spiel erzielt haben und demnach eine Partei kurz vor dem Sieg steht, dann sabotiert der Spion. Dadurch verhindert er entweder den Sieg der Widerstandskämpfer oder führt den Sieg für die Spione herbei. Dabei ist es egal, ob noch ein weiterer Spion im Team ist, der auch sabotiert. In den anderen Fällen wird anhand der Parameter s_6 bis s_9 über die Sabotage entschieden. Bei zwei Spionen im Team ist Vorsicht geboten, da es passieren könnte, dass beide Spione sich dazu entscheiden, zu sabotieren. Diese Entscheidung wird je nach der Teamgröße und

anhand der Parameter s_6 und s_7 getroffen. In der ersten Mission, bei der nur ein Spion im Team ist, wird gesondert mit Parameter s_8 entschieden. Diese Mission wird extra betrachtet, da hier denkbar wäre, nicht zu sabotieren, um vielleicht das Vertrauen bei den Widerstandskämpfern zu erhöhen. Oder der Spion sabotiert, um bereits einen Punkt für die Spione zu erzielen. Der Parameter ermöglicht es herauszufinden, was besser im Spiel gegen die Widerstandskämpfer funktioniert. In allen anderen Missionen, bei welchen nur ein Spion im Team ist, sabotiert dieser mit Wahrscheinlichkeit s_9.

5 Evolution der Spielstrategien

In diesem Kapitel wird erklärt, wie die verwendete Evolutionsstrategie zur Evolution der Spielstrategien aufgebaut wurde. Anschließend folgen die Ergebnisse aus der Evolution und die Auswertung.
Die Implementation des Modells und des evolutionären Algorithmus für diese Bachelorarbeit erfolgte in Python. Für den evolutionären Algorithmus wurde mit dem Framework DEAP (Distributed Evolutionary Algorithms in Python) gearbeitet, welches die Implementierung durch viele vorgefertigte Datenstrukturen und genetische Operatoren erleichtert [15]. DEAP erlaubt es, für das Problem spezifische Anpassungen vorzunehmen und die Datenstruktur der Individuen ganz einfach und individuell zu erstellen. Unter anderem können genetische Algorithmen, genetische Programmierung und Evolutionsstrategien mit DEAP umgesetzt werden.

5.1 Individuen und Populationen

Gesucht wird jeweils eine optimale Strategie für den Widerstandskämpfer und Spion auf Basis des Modells aus Kapitel 4, sofern diese existieren. Die optimale Spielstrategie für einen Spieler setzt sich dann aus der optimalen Strategie des Widerstandskämpfers und des Spions zusammen.
Die hier verwendete Umsetzung der Evolutionsstrategie erfolgt mit zwei Populationen. Die Spielstrategie wird für die Evolution somit in die zwei Rollen unterteilt. Es gibt eine Population für die Widerstandskämpfer und eine für die Spione. Die Individuen beider Populationen werden parallel in der Co-Evolution evolviert, wobei die Fitness der Widerstandskämpfer von den Spionen abhängig ist und umgekehrt.
Ein Individuum aus der Population der Widerstandskämpfer besteht aus einem Chromosom mit insgesamt 14 Genen, was einer Liste mit 14 Einträgen entspricht. Für ein Spion-Individuum gibt es zehn Parameter in der Liste. In der Umsetzung mit Python und DEAP ist ein Individuum ein Objekt einer Klasse, welches die Liste der zu evolvierenden Parameter enthält, sowie eine Fitness-Eigenschaft besitzt. (Es lassen sich auch noch weitere Objekt-Eigenschaften hinzufügen, welche für die Berechnung der Fitness nützlich sein könnten.)
Für die Selektion werden die Individuen anhand ihrer Fitness miteinander verglichen. Je höher die Fitness ist, desto besser war das Individuum in den Spielpartien.

5.2 Fitnessfunktion

Die Fitnessfunktion dient zur Bewertung der Individuen. Der Erfolg eines evolutionären Algorithmus hängt unter anderem von ihr ab. Die Evaluierung der Individuen aus der aktuellen Generation wird in diesem Fall mithilfe der sogenannten Hall-of-Fame 1997 durchgeführt. In jeder Generation wird der Widerstandskämpfer mit der höchsten Fitness zu der sogenannten Hall-of-Fame der Widerstandskämpfer hinzugefügt. Dies geschieht analog für die Spione. Die Evaluierung mit der Hall-of-Fame testet alle Individuen der aktuellen Generation in Abhängigkeit der gegnerischen besten Vorfahren. Wie die Funktion implementiert wurde, kann in Algorithmus 7 nachvollzogen werden. Für die Evaluierung der aktuellen Population der Widerstandskämpfer wird jedes dieser Individuen gegen n zufällig ausgewählte Spione aus der Hall-of-Fame getestet, indem pro Gegner eine Spielpartie durchlaufen wird. Für die Berechnung der Fitness wird gezählt, wie oft die Widerstandskämpfer eine Partie gewonnen haben. Die Fitness ergibt sich dann aus der relativen Häufigkeit der gewonnenen Partien. Analog geht man für die Evaluierung der Spione vor.
Ohne Verwendung der Hall-of-Fame für die Evaluierung würden die Individuen nur gegen die Konkurrenten aus der aktuellen Generation antreten. Welche Evaluierung bessere Ergebnisse liefert, kann nicht verallgemeinert werden. Für die Evolution der Spielstrategien hat sich jedoch die Evaluierung mit der Hall-of-Fame als besonders geeignet herausgestellt, um einen Lernprozess der Individuen zu erzielen.

Im Algorithmus 7 ist Pop_{wk} die aktuelle Population der Widerstandskämpfer und Pop_{sp} die der Spione. Mit HoF_{wk} und HoF_{sp} ist die Hall-of-Fame der Widerstandskämpfer und Spione gemeint. Ein Individuum aus der jeweiligen Population oder Hall-of-Fame wird mit wk für Widerstandskämpfer und mit sp für Spion bezeichnet.
Die in Abschnitt 4.3 eingeführte Spielumgebung benötigt fünf Spieler als Eingabe, wobei ein Spieler jeweils eine Parameterliste für den Widerstandskämpfer und eine für den Spion besitzt. Je nach zugewiesener Rolle werden die zugehörigen Parameter für die Spielpartie verwendet. Daher erfolgt die programmtechnische Umsetzung der Evaluierung im Algorithmus 7 so, dass alle fünf Spieler identische Parameter als Widerstandskämpfer und als Spion aufweisen. Es wird immer nur ein Genotyp der Widerstandskämpfer und ein Genotyp der Spione für eine Partie verwendet. Drei der Spieler erhalten die Rolle als Widerstandskämpfer und werden nach denselben Parametern als Widerstandskämpfer spielen. Die anderen zwei Spieler sind Spione und spielen ebenfalls nach demselben Spion-Genotypen. Es ist egal, welcher Spieler welche Rolle erhält, da alle die dieselben Parameter für beide Rollen besitzen. Wird ein Widerstandskämpfer evolviert, so werden alle fünf teilnehmenden Spieler aus dem zu evolvierenden Widerstandskämpfer und dem zufällig ausgewählten Spion zusammengesetzt. Für die Bewertung ist dann entscheidend, ob die Widerstandskämpfer die Partie gewinnen oder nicht.

Algorithmus 7 : Evaluierung mit der Hall-of-Fame

```
Function evaluiere(Pop_wk, Pop_sp, HoF_wk, HoF_sp, n):
    for each wk in Pop_wk                          // Widerstandskämpfer evaluieren
        Spielsiege ← 0
        for i ← 1 bis n
            if HoF_sp = leer                        // 0. Generation evaluieren
                sp ← zufälliger Gegner aus Pop_sp
            else                                    // alle anderen Generationen
                sp ← zufälliger Gegner aus HoF_sp
            end if
            Spieler ← [wk, sp]
            Spiel(Spieler, Spieler, Spieler, Spieler, Spieler)     // Spieldurchlauf
                                                   // mit fünf identischen Spielern
            if Sieg für Widerstandskämpfer
                Spielsiege ← Spielsiege + 1
            end if
        end for
        Fitness von wk ← Spielsiege/n * 100
    end for

    for each sp in Pop_sp                                  // Spione evaluieren
        Spielsiege ← 0
        for i ← 1 bis n
            if HoF_wk = leer
                wk ← zufälliger Gegner aus Pop_wk
            else
                wk ← zufälliger Gegner aus HoF_wk
            end if
            Spieler ← [wk, sp]
            Spiel(Spieler, Spieler, Spieler, Spieler, Spieler)
            if Sieg für Spione
                Spielsiege ← Spielsiege + 1
            end if
        end for
        Fitness von sp ← Spielsiege/n * 100
    end for
end
```

5.3 Evolution

Für die Anwendung der Evolutionsstrategie müssen zunächst noch einige Parameter definiert werden. In Tabelle 5.1 sind alle regulierbaren Parameter für die hier verwendete Evolutionsstrategie aufgelistet. Es wird mit einer (μ, λ)-Strategie gearbeitet.

Parameter	Erklärung
G	Anzahl der Generationen
λ	Populationsgröße für Widerstandskämpfer und Spione
s_1	Widerstandskämpfer Standardabweichung für Mutation
s_2	Spion Standardabweichung für Mutation
α	deterministische Adaption der Schrittweite für Spion mit $s_2 = \alpha * s_2$
p_{cx}	Wahrscheinlichkeit, dass die diskrete Rekombination angewandt wird (ansonsten wird Nachkomme durch Mutation erzeugt)
S	Selektionsdruck $\in (0, 1)$, d.h. $\mu = S * \lambda$
n	Anzahl der Gegner aus der Hall-of-Fame für die Evaluierung eines Individuums

Tabelle 5.1: Parameter der Evolutionsstrategie

Der Ablauf der Co-Evolution kann mit Algorithmus 8 nachvollzogen werden. Nachdem alle oben genannten Parameter eingestellt wurden, können die beiden Anfangspopulationen nach vorgegebener Größe λ erzeugt werden. Die Individuen der Widerstandskämpfer-Population werden initialisiert, indem ihre 14 Parameterwerte jeweils eine zufällige Zahl aus dem Intervall $[-1, 1]$ annehmen. Die Parameter der Spione werden zufällig aus $[0, 1]$ gewählt, da es sich um Wahrscheinlichkeiten handelt. Zur Evaluierung der Populationen wird die bereits beschriebene Fitnessfunktion verwendet. Dasjenige Individuum in der aktuellen Population der Widerstandskämpfer und Spione, welches die höchste Fitness erhält, wird in der jeweiligen Hall-of-Fame gespeichert. Es wird eine Generation nach der anderen generiert, bis das Abbruchkriterium von G-vielen Generationen erfüllt ist. Für die Selektion der Eltern werden alle Individuen der aktuellen Population anhand ihrer Fitness sortiert und μ Individuen mit den höchsten Fitnesswerten als Eltern verwendet. Die Zahl μ ergibt sich aus dem gewählten Selektionsdruck. Aus dem Eltern-Pool werden nacheinander zufällig ein oder zwei Individuen ausgewählt, welche durch die Anwendung eines genetischen Operators einen Nachkommen erzeugen, bis λ Nachkommen entstanden sind. Mit Wahrscheinlichkeit p_{cx} werden zwei zufällige Eltern ausgewählt, welche durch eine diskrete Rekombination einen Nachkommen erzeugen. Ansonsten wird immer die Mutation verwendet und die Gene eines Eltern-Individuums werden mutiert, um einen Nachkommen zu erstellen.

Algorithmus 8 : Evolutionsstrategie

Input : G, λ, s_1, s_2, α, p_{cx}, S, n
$g \leftarrow 0$ `// Generation`
$\mu \leftarrow S * \lambda$
`// Anfangspopulationen erstellen:`
$Pop_{wk} \leftarrow$ erzeuge λ zufällige Individuen der Widerstandskämpfer
$Pop_{sp} \leftarrow$ erzeuge λ zufällige Individuen der Spione
$HoF_{wk} \leftarrow [\,]$ `// leere Hall-of-Fame`
$HoF_{sp} \leftarrow [\,]$
`// Generation` 0 `evaluieren:`
evaluiere(Pop_{wk}, Pop_{sp}, HoF_{wk}, HoF_{sp}, n) `// Fitnessfunktion`
füge besten Widerstandskämpfer zu HoF_{wk} hinzu
füge besten Spion zu HoF_{sp} hinzu
while $g < G$ `// nächste Generation erstellen`
  $g \leftarrow g + 1$
  $Eltern_{wk} \leftarrow$ selektiere die μ besten Individuen aus Pop_{wk}
  $Eltern_{sp} \leftarrow$ selektiere die μ besten Individuen aus Pop_{sp}
  $Nachkommen_{wk} \leftarrow [\,]$
  $Nachkommen_{sp} \leftarrow [\,]$
  `// generiere` λ `Nachkommen für die neue Generation:`
  while *Anzahl Nachkommen der Widerstandskämpfer* $< \lambda$
    mit p_{cx}: $Ind_1, Ind_2 \leftarrow$ zwei zufällige Individuen aus $Eltern_{wk}$
    Nachkomme für $Nachkommen_{wk} \leftarrow$ diskrete Rekombination von Ind_1, Ind_2
    mit $1 - p_{cx}$: $Ind \leftarrow$ ein zufälliges Individuum aus $Eltern_{wk}$
    Nachkomme für $Nachkommen_{wk} \leftarrow$ Mutation von Ind mit s_1
  end while
  while *Anzahl Nachkommen der Spione* $< \lambda$
    mit p_{cx}: $Ind_1, Ind_2 \leftarrow$ zwei zufällige Individuen aus $Eltern_{sp}$
    Nachkomme für $Nachkommen_{sp} \leftarrow$ diskrete Rekombination von Ind_1, Ind_2
    mit $1 - p_{cx}$: $Ind \leftarrow$ ein zufälliges Individuum aus $Eltern_{sp}$
    Nachkomme für $Nachkommen_{sp} \leftarrow$ Mutation von Ind mit s_2
  end while
  $s_2 \leftarrow \alpha * s_2$
  $Pop_{wk} \leftarrow Nachkommen_{wk}$ `// ersetze alte Population durch Nachkommen`
  $Pop_{sp} \leftarrow Nachkommen_{sp}$
  `// Generation` g `evaluieren:`
  evaluiere(Pop_{wk}, Pop_{sp}, HoF_{wk}, HoF_{sp}, n)
  füge besten Widerstandskämpfer zu HoF_{wk} hinzu
  füge besten Spion zu HoF_{sp} hinzu
end while
return HoF_{wk}, HoF_{sp}

5.3.1 Mutation

Die Mutation eines Individuums aus der Population der Widerstandskämpfer erfolgt, indem alle 14 Parameterwerte durch Addition einer normalverteilten Zufallszahl verändert werden. Die zugrundeliegende Normalverteilung besitzt den Erwartungswert 0 und die Standardabweichung $s = s_1$. Bei den Parameterwerten des Widerstandskämpfers ist nicht der Wert selbst, sondern das Verhältnis zwischen den Werten entscheidend. Daher dürfen die Werte den zur Initialisierung verwendeten Bereich $[-1, 1]$ auch verlassen. Ist der Wert des Parameters r_0 zum Beispiel doppelt zu groß wie der von r_1, so wird die vorliegende Spielsituation für r_0 deutlich stärker bestraft oder belohnt, als die Situation für r_1. Wenn alle Parameter mit dem gleichen Wert vervielfacht werden, ergibt sich in der nach aufsteigenden Scores sortierten Buchführung die gleiche Reihenfolge. Weil nur das Verhältnis ausschlaggebend ist, ist die Mutationsschrittweite s_1 an sich irrelevant. Sie sollte lediglich der anfänglichen Parameterinitialisierung entsprechend passend gewählt werden. Eine Mutationsschrittweiten-Adaption ist hier also nicht nötig.
Die Mutation der Spion-Individuen erfolgt mithilfe der Normalverteilung mit Erwartungswert 0 und $s = s_2$. Die Parameter des Spions sind Wahrscheinlichkeiten und demnach auf das Intervall $[0, 1]$ begrenzt. Wird durch Addition einer normalverteilten Zufallszahl ein Parameter-Wert außerhalb der Intervallgrenzen erzeugt, so wird der Wert entsprechend auf den am nächsten liegenden Randwert gesetzt. Im Gegensatz zu den Widerstandskämpfern, ist bei den Spionen der tatsächliche Wert des Parameters von Bedeutung. Bei Vervielfachung würden ganz andere Wahrscheinlichkeiten entstehen oder die Intervallgrenzen überschritten werden. Im Fall des Spions wird daher eine deterministische Adaption der Schrittweite eingeführt. Die Schrittweite s_2 wird mit jeder Generation um den Faktor α verkleinert, um ein Konvergenzverhalten zu erzielen. Die Parameter werden bei der Mutation in frühen Generationen also stärker verändert und durch kleiner werdende Mutationsstärken nach und nach verfeinert, um den optimalen Wertebereich nicht beim nächsten Schritt wieder zu verlassen.

5.4 Ergebnisse und Auswertung

Die Parameter für die Evolutionsstrategie wurden wie in Tabelle 5.2 eingestellt und mit diesen zehn mal eine Evolution durchlaufen. Die gewählte Parameterkonstellation erzielte gute Ergebnisse, welche im Folgenden gezeigt werden. Der Selektionsdruck wurde auf $14,3\%$ gesetzt, um das Verhältnis von eins zu sieben zwischen μ und λ zu erhalten, welches laut Gerdes *u. a.* ein übliches Verhältnis ist [2].

5.4.1 Fitness

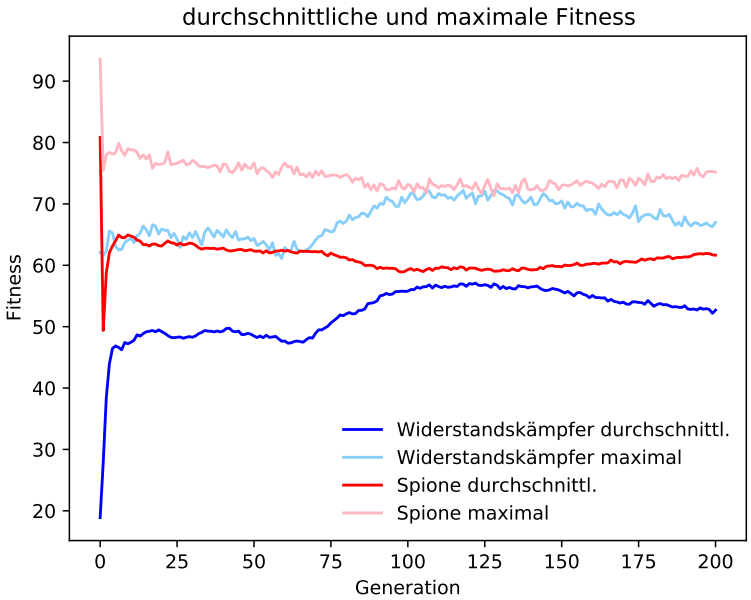
Zunächst einmal ist es interessant, die Entwicklung der Fitness während der Evolution zu betrachten. In Abbildung 5.1a wird das gemittelte Ergebnis über die zehn Simulationen

Parameter	eingestellter Wert
G	200
λ	150
s_1	0.1
s_2	0.1
α	0.995
p_{cx}	0.1
S	$0.143 \Rightarrow \mu = 21$
n	100

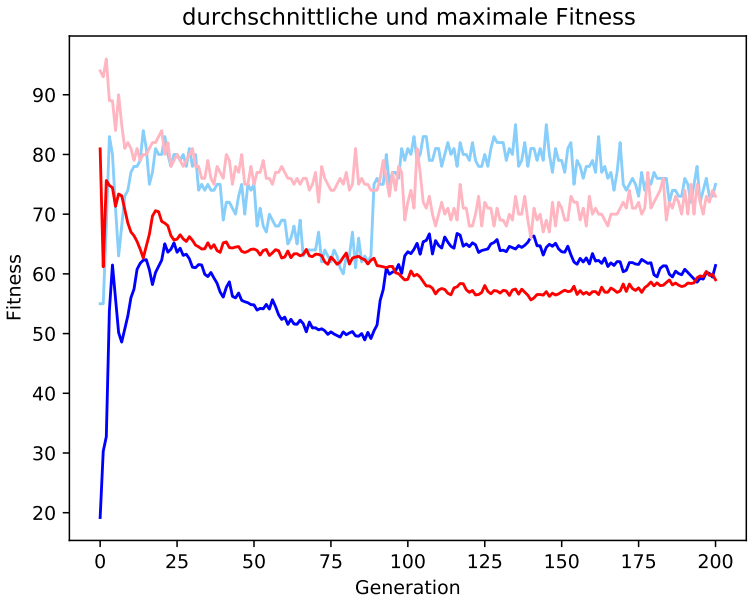
Tabelle 5.2: Einstellung der Parameter für die Evolutionsstrategie

dargestellt und in Abbildung 5.1b das Ergebnis einer der zehn Simulationen, welche im Folgenden als Simulation 1 bezeichnet wird.
Die beiden Abbildungen zeigen die Fitness über den Verlauf der Generationen. Die beiden blauen Kurven gehören zu den Widerstandskämpfern und die beiden roten zu den Spionen. Die farblich jeweils dunklere Kurve zeigt die durchschnittliche Fitness der Population in der entsprechenden Generation. Die jeweils hellere Kurve zeigt die maximal erreichte Fitness in der Generation. Wie man sehen kann, ist die Fitness der Widerstandskämpfer zu Beginn der Evolution deutlich schlechter als die der Spione. Das bedeutet, dass die zufällig generierten Spione viel häufiger gewinnen als die zufällig initialisierten Widerstandskämpfer. Die Fitness der Anfangspopulationen liegt im Durchschnitt etwa bei 80% für die Spione und nur bei 20% für die Widerstandskämpfer. Dies liegt vermutlich daran, dass der Suchraum für die Parameter des Widerstandskämpfers bedeutend größer ist und weniger gute Parameterkonstellationen bei der Initialisierung getroffen werden. Innerhalb der nächsten Generationen steigt die Kurve für den Widerstandskämpfer stark an, was darauf hindeutet, dass bereits bessere Parameter gefunden wurden. Der Spion erfährt eine fallende Fitness, was jedoch nicht bedeuten muss, dass sich die Parameter verschlechtert haben. In Abbildung 5.1a kann man sehen, dass die Fitness im Durchschnitt über die zehn Simulationen größtenteils sehr konstant bleibt. Dies bedeutet nicht automatisch, dass keine weitere Verbesserung der Parameter eintrat, sondern es muss untersucht werden, ob dieses Verhalten auf den Red-Queen-Effekt zurückzuführen ist. Mithilfe von geeigneten Messmethoden kann gezeigt werden, dass trotz konstant bleibender und auch fallender Fitness bessere Parameter evolviert wurden. Weil eine Co-Evolution stattfindet, ist durch die Fitness allein der Lernfortschritt nicht feststellbar.

(Hinweis: Nur bei Bewertung der Anfangspopulationen werden die gegnerischen Individuen aus derselben Generation entnommen, da noch keine Individuen in der Hall-of-Fame vorliegen. Für die Evaluierung nachfolgender Generationen werden die Vorfahren aus der Hall-of-Fame als Gegner ausgewählt. Weil die Individuen also nicht in Abhängigkeit der gegnerischen Individuen aus der aktuellen Generation bewertet werden, ergibt die aufsummierte



(a) Fitness im Mittel über zehn Evolutionsdurchläufe



(b) Fitness von Simulation 1

Abbildung 5.1: Fitness im Verlauf der Generationen
rot = Spion, blau = Widerstandskämpfer; farblich dunklere Kurve zeigt die durchschnittliche Fitness und die hellere Kurve die maximale Fitness

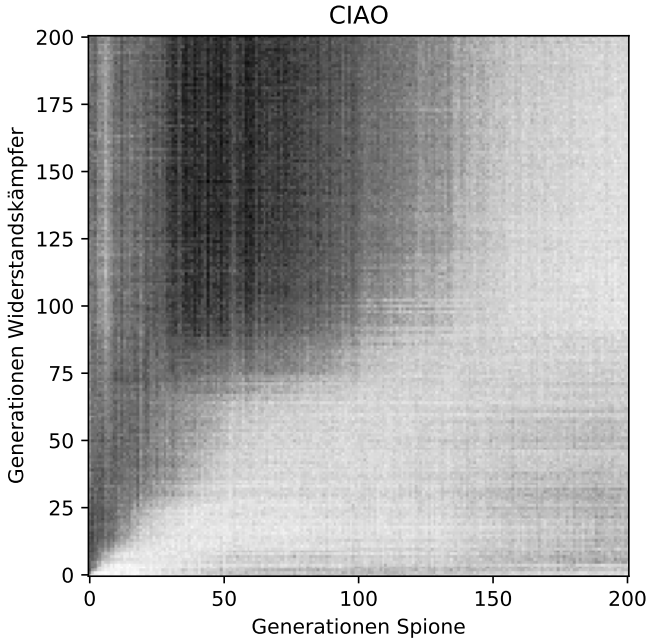
durchschnittliche Fitness von Widerstandskämpfer und Spion nicht 100%, so wie es in der nullten Generation der Fall ist.)

5.4.2 Angewandte Messmethoden

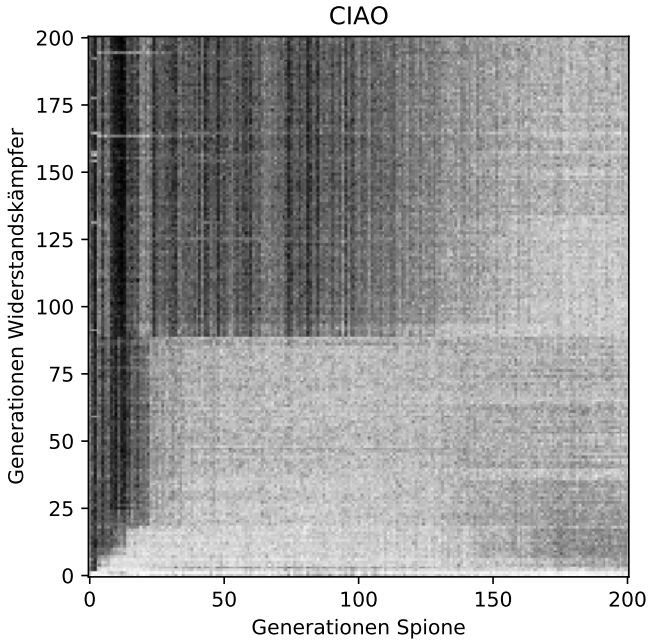
In jeder Generation wird der beste Widerstandskämpfer und der beste Spion in der Hall-of-Fame gespeichert. Dies ist nicht nur relevant für die Evaluierung mit der Hall-of-Fame-Methode, sondern auch, um nach beendeter Evolution verschiedene Tests durchführen zu können. Die Tests dienen dazu, herauszufinden, ob tatsächlich ein Lernprozess bei der Co-Evolution stattgefunden hat. In Kapitel 3.4 wurde bereits auf verschiedene Messmethoden eingegangen, welche hier nun verwendet werden, um die Evolution auszuwerten.
Um einen Lernfortschritt festzustellen, müssten die Individuen aus späteren Generationen besser sein, als die Konkurrenten aus früheren Generationen [10]. Die CIAO-Methode von Cliff & Miller und das Master-Tournament von Floreano & Nolfi basieren auf dieser Idee und verwenden jeweils das beste Individuum aus jeder Generation als Repräsentant, um den Lernprozess zu überprüfen.

CIAO

In Abbildung 5.2 sind zwei CIAO-Datensätze dargestellt. Das obere Bild (5.2a) zeigt dabei wieder das durchschnittliche Ergebnis über die zehn Evolutionsdurchläufe. Das untere Bild (5.2b) visualisiert den CIAO-Datensatz von Simulation 1.
Nachdem die Evolution durchlaufen wurde, kann der zugehörige CIAO-Datensatz ermittelt werden, indem jedes Individuum aus der Hall-of-Fame der Widerstandskämpfer jeweils m-mal gegen jedes Individuum aus der Hall-of-Fame der Spione antritt. Ein Datenpunkt entspricht hier immer der relativen Häufigkeit gewonnener Spiele für den Widerstandskämpfer. Zum genaueren Verständnis kann der Berechnungsvorgang in Algorithmus 9 nachvollzogen werden. Der Parameter m wurde auf den Wert 100 gesetzt. Bei 201 durchlaufenen Generationen (Generationen 0 bis 200) ergeben sich $201 * 201$ Werte im Datensatz. Jeder dieser Werte wird als ein Pixel im Bild aufgetragen.
Ein Pixel an der Stelle (g_x, g_y) visualisiert, wie gut der beste Widerstandskämpfer aus Generation g_y gegen den besten Spion aus Generation g_x in m Spielen abgeschnitten hat. In einer Reihe im CIAO-Bild werden die Ergebnisse eines einzigen Widerstandskämpfers gegen die verschiedenen Spione aus der Hall-of-Fame abgetragen. Je dunkler der Pixel ist, desto häufiger hat der Widerstandskämpfer gewonnen und je heller der Pixel, desto häufiger gewann der Spion. Ein komplett schwarzer Pixel würde einer relativen Gewinnhäufigkeit von 100% für den Widerstandskämpfer bedeuten. Ist er komplett weiß, heißt dies, dass der Widerstandskämpfer 0% der Spiele gewonnen hat und somit der Spion 100% der Spiele gewann. Wenn die Widerstandskämpfer gegen viele Spion-Vorfahren gewinnen und gegen die Spione aus nachfolgenden Generationen häufiger verlieren, dann kann man eine diagonale Aufteilung in



(a) CIAO-Datensatz im Mittel über zehn Evolutionsdurchläufe



(b) CIAO-Datensatz von Simulation 1

Abbildung 5.2: Visualisierung von CIAO-Datensätzen

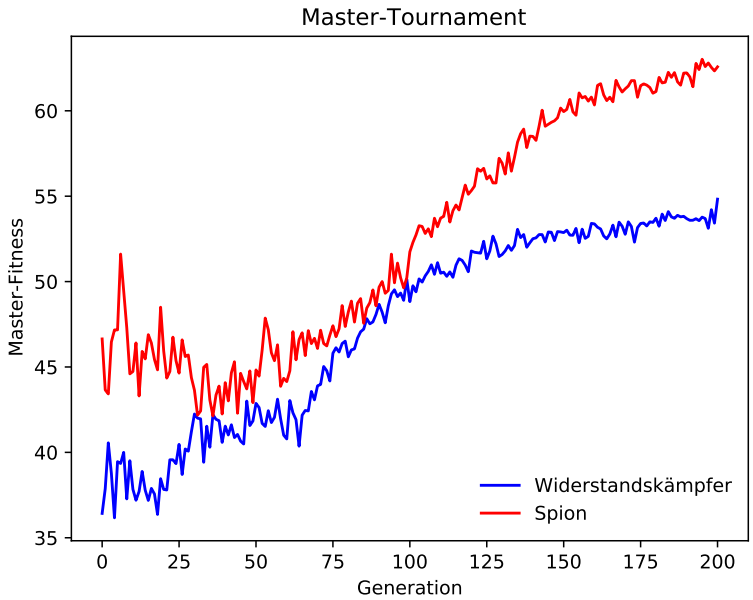
dem Bild feststellen, wobei oben eine dunklere Fläche und unten eine hellere entsteht. Genau solch eine diagonale Trennung ist ein Indikator für einen kontinuierlichen Lernfortschritt [12]. In Abbildung 5.2a ist die diagonale Aufteilung deutlich zu erkennen. Im Durchschnitt über die zehn Evolutionsdurchläufe kann man mit der hier angewandten Evolutionsstrategie also einen Lernfortschritt erzeugen. Die CIAO-Daten zeigen, dass die Individuen gegen die Konkurrenten aus vorherigen Generationen besser abschneiden, als gegen die Konkurrenten aus späteren Generationen. Dies bedeutet, dass die Parameter der jeweils gegnerischen Individuen in der Evolution noch weiter verbessert wurden. Auch in Abbildung 5.2b, welche zu Simulation 1 gehört, ist dies zu erkennen. Zu erwähnen ist allerdings, dass nicht alle zehn Simulationen solch ein gutes Ergebnis gezeigt haben. Bei manchen Evolutionsdurchläufen war keine diagonale Trennung zu sehen. Im Durchschnitt ist der Lernfortschritt jedoch deutlich erkennbar.

Algorithmus 9 : Berechnung der Gewinnhäufigkeiten für den CIAO-Datensatz

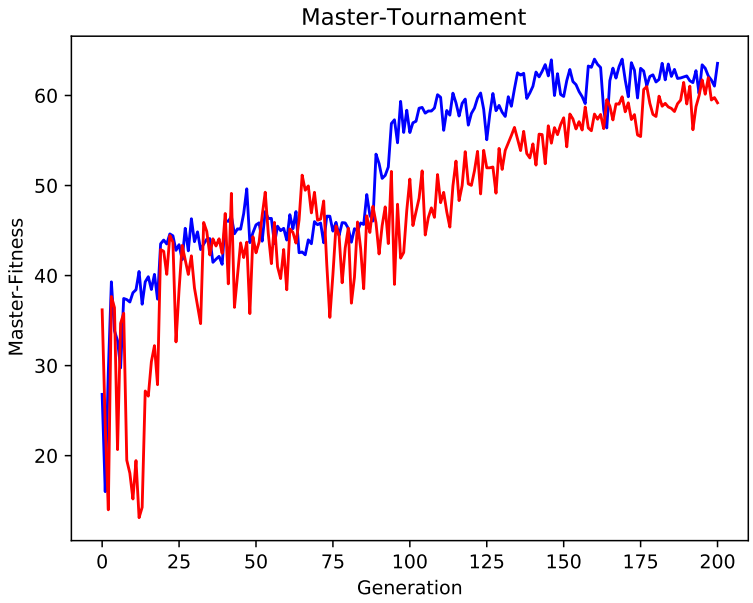
```
g_y ← 0                          // Generation des betrachteten Widerstandskämpfers
for each wk in HoF_wk
    g_x ← 0                                  // Generation des betrachteten Spions
    for each sp in HoF_sp
        Spieler ← [wk, sp]
        for i ← 1 bis m
            Spiel(Spieler, Spieler, Spieler, Spieler, Spieler)  // m Spieldurchläufe
                                                                // mit fünf identischen Spielern
        end for
        Wert bei (g_x, g_y) ← Spielsiege / m          // Widerstandskämpfer Spielsiege
        g_x ← g_x + 1
    end for
    g_y ← g_y + 1
end for
```

Master-Tournament

Für die Erstellung der Master-Tournament-Daten für Abbildung 5.3 wurde sehr ähnlich vorgegangen wie für den CIAO-Datensatz. Es handelt sich lediglich um eine andere Darstellungsvariante. Abbildung 5.3a zeigt dabei das durchschnittliche Ergebnis des Master-Tournaments über die zehn Evolutionsdurchläufe. Abbildung 5.3b gehört zu Simulation 1.
Die blaue Kurve zeigt die Master-Fitness der Widerstandskämpfer und die rote die der Spione. Für die Erstellung der blauen Kurve wurde für jede Generation der entsprechende Widerstandskämpfer aus der Hall-of-Fame entnommen und dieser trat nacheinander m-mal gegen jeden Spion aus der Hall-of-Fame der Spione an. Dies ist identisch mit dem Vorgehen



(a) Master-Tournament im Mittel über zehn Evolutionsdurchläufe



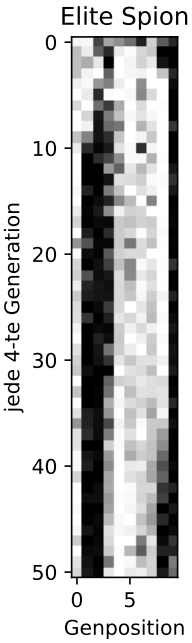
(b) Master-Tournament von Simulation 1

Abbildung 5.3: Master-Fitness für Widerstandskämpfer (= blau) und Spione (= rot)

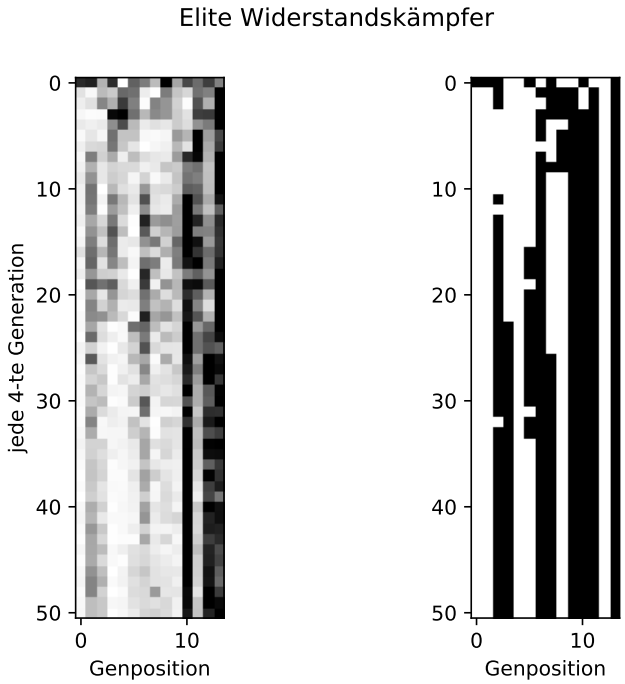
bei Erstellung der CIAO-Daten und m wurde auch hier auf den Wert 100 gesetzt. Insgesamt werden für einen Widerstandskämpfer $m * 201$ (für Generationen 0 bis 200) Partien gespielt. Für den CIAO-Datensatz wird für jeden Gegner die relative Gewinnhäufigkeit berechnet. Beim Master-Tournament ist es aber so, dass der Widerstandskämpfer einen einzigen Wert erhält, welcher die relative Gewinnhäufigkeit über alle $m * 201$ Spiele angibt. Dies ist die sogenannte Master-Fitness, welche in dem Graph eingetragen wird. Für die Spione wird analog vorgegangen.
In Abbildung 5.3a kann man sehr gut sehen, dass die Kurven beider Populationen im Durchschnitt fast kontinuierlich wachsen, was laut Floreano & Nolfi einen Lernfortschritt bedeutet. Auch für Simulation 1 kann der Lernfortschritt in Abbildung 5.3b deutlich nachvollzogen werden. Man kann außerdem feststellen, dass die Widerstandskämpfer und Spione aus den letzten Generationen die höchste Master-Fitness besitzen. Dies sind also die besten Individuen der gesamten Evolution.

Elite Bitmap

Um zu sehen, was auf Ebene der Genotypen passiert, eignet sich die Methode der Elite Bitmap nach Cliff & Miller. Die Elite entspricht den besten Individuen aus allen Generationen, was identisch mit der Hall-of-Fame ist. Diese Visualisierung wurde ursprünglich für genetische Algorithmen entwickelt, bei welchen die Individuen durch Nullen und Einsen kodiert werden und die Farbpixel daher nur schwarz oder weiß sind. Für die Evolutionsstrategien wird die Elite Bitmap hier etwas abgewandelt, indem die reellen Parameter in Graustufen visualisiert werden.
Die Elite Bitmaps in Abbildung 5.4 beziehen sich nur auf die Evolution von Simulation 1. Dabei wird nur jede vierte Generation betrachtet, um ein übersichtlicheres Bild zu erhalten. Die Zahl 50 an der Y-Achse steht daher für die Generation $50 * 4 = 200$. In der oberen Elite Bitmap (5.4a) werden die Chromosomen der Elite-Spione in Graustufen dargestellt. Da die Parameter des Spions in dem Intervall $[0, 1]$ liegen, entspricht ein schwarzer Pixel dem Parameterwert 1 und ein weißer dem Wert 0. Sämtliche Werte dazwischen werden durch die verschiedenen Grautöne repräsentiert. An Genposition i befindet sich der Wert des Parameters s_i für $i = 0, \ldots, 9$. Wie man erkennen kann, wird ab Generation 40 (entspricht der Zahl 10 auf der Y-Achse) bereits eine gewisse Strategie beibehalten, welche nur noch allmählich modifiziert wird und ab Generation 160 (40 auf der Y-Achse) bereits dem Endergebnis aus Generation 200 ähnelt.
Für die Widerstandskämpfer funktioniert die Visualisierung in einer Elite Bitmap nicht wie bei den Spionen, da der Suchraum für die Parameter nicht auf ein bestimmtes Intervall begrenzt ist. Da nur die Verhältnisse zwischen den Parametern r_i $(i = 0, \ldots, 13)$ für die Buchführung relevant sind, werden die Parameter normiert, um die verschiedenen Individuen miteinander vergleichen zu können. Für jedes Individuum aus der Hall-of-Fame muss demnach der betragsmäßig größte Parameterwert ermittelt werden, um anschließend alle



(a) Genotyp Spion



(b) Genotyp Widerstandskämpfer, links: normierte Parameter im Betrag, rechts: Vorzeichen der Parameter (schwarz = positiv, weiß = negativ)

Abbildung 5.4: Genotypen der besten Individuen aus jeder vierten Generation von Simulation 1

Parameterwerte des Individuums durch diesen Wert zu dividieren. Da die Parameter des Widerstandskämpfers auch negative Werte annehmen dürfen, wurden zur besseren Übersicht zwei Bilder erstellt. Das linke Bild in Abbildung 5.4b zeigt die normierten Parameter der Widerstandskämpfer im Betrag. Das heißt, dass nur Werte aus dem Intervall $[0, 1]$ visualisiert werden. Genauso wie bei den Spionen entspricht ein schwarzer Pixel dem Wert 1, ein weißer dem Wert 0 und die Grautöne repräsentieren die Werte dazwischen. Das rechte Bild zeigt die zugehörigen Vorzeichen der Parameterwerte. Ein schwarzer Pixel steht für ein positives Vorzeichen und ein weißer für ein negatives. Man kann sehen, dass sich die Vorzeichen der besten Widerstandskämpfer ab Generation 136 (34 auf der Y-Achse) nicht mehr ändern. Im linken Bild sieht man, dass bereits ab Generation 104 (26 auf der Y-Achse) nur noch kleine Änderungen der Parameterwerte (im Verhältnis betrachtet) stattfinden. Man kann außerdem feststellen, dass die Elite-Individuen in ihren Vorzeichen für die Parameter r_0, r_1, r_4, r_6, r_9, r_{10}, r_{11}, r_{12} und r_{13} (Index des Parameters entspricht der Genposition) über die gesamte Evolution sehr konstant geblieben sind.

5.4.3 Evolvierte Parameter

Die evolvierten Parameterwerte für die in Abbildung 5.4 visualisierten Genotypen werden in Tabelle 5.3 für den Widerstandskämpfer und Spion aus Generation 200 angegeben. Für den Widerstandskämpfer handelt es sich dabei wieder um die normierten Werte. Die besten Individuen aus der letzten Generation gehören zu den stärksten aus der gesamten Evolution, wie man im Master-Tournament in Abbildung 5.3b sehen kann. Die gezeigten Parameterwerte sind das Ergebnis aus einem der insgesamt zehn Evolutionsdurchläufe. Es gab zum Teil Abweichungen in den anderen durchlaufenen Simulationen. Daher ist in Tabelle 5.3 ebenfalls der Mittelwert und die Standardabweichung für die einzelnen Parameter der zehn Simulationen angegeben. Die Ergebnisse beziehen nur auf die Parameter des besten Spions und besten normierten Widerstandskämpfers aus Generation 200. Für den Widerstandskämpfer gibt es in der Tabelle außerdem die Spalte „u", welche angibt, wie viele der in zehn Simulationen evolvierten Parameter im Vorzeichen mit dem des Mittelwerts übereinstimmen.

Für den Widerstandskämpfer gilt, dass Parameter, welche einen Wert sehr nah an 0 annehmen, niedrig gewichtet sind und keinen so großen Einfluss auf die Buchführung und damit den Spielsieg haben. Betragsmäßig hohe Parameterwerte werden dagegen mit einer hohen Gewichtung versehen und verändern die Scores der Spieler in der Buchführung stärker. Positive Werte „bestrafen" die Scores der Spieler und negative „belohnen" sie, denn es gilt, dass ein Spieler mit einem hohen Score vielleicht ein Spion ist und daher für das Missionsteam nicht in Frage kommt.
Betrachtet man die evolvierten Parameter des Widerstandskämpfers in Tabelle 5.3, stellt man fest, dass r_0 sowohl in Simulation 1 als auch im Durchschnitt über die zehn Simulationen sehr gering gewichtet wird, sodass das Vorzeichen kaum eine Relevanz hat. Der

Wk	Simulation 1	M	SD	u		Spion	Simulation 1	M	SD
r_0	-0.06	0.06	0.176	4		s_0	0.265	0.306	0.336
r_1	-0.368	-0.315	0.318	7		s_1	0.917	0.454	0.367
r_2	0.322	0.092	0.201	7		s_2	0.98	0.973	0.036
r_3	0.017	-0.188	0.299	6		s_3	0.163	0.285	0.319
r_4	-0.068	-0.274	0.272	10		s_4	0.0	0.164	0.294
r_5	-0.152	-0.038	0.229	7		s_5	0.151	0.038	0.045
r_6	0.272	0.252	0.273	9		s_6	0.131	0.148	0.197
r_7	0.287	-0.038	0.351	6		s_7	0.275	0.274	0.122
r_8	-0.241	-0.146	0.254	6		s_8	0.91	0.56	0.437
r_9	0.105	-0.013	0.313	3		s_9	0.659	0.714	0.332
r_{10}	0.919	0.473	0.217	10					
r_{11}	0.343	0.34	0.247	9					
r_{12}	-0.875	-0.706	0.414	9					
r_{13}	1.0	0.662	0.297	10					

Tabelle 5.3: Parameter des besten Individuums aus Generation 200 für Simulation 1 und Mittelwert und Standardabweichung über alle zehn Simulationen.
(alle Werte auf dritte Nachkommastelle gerundet)
Wk = Widerstandskämpfer, M = Mittelwert, SD = Standardabweichung
u = Anzahl der mit dem Mittelwert übereinstimmenden Vorzeichen

Parameter r_0 ist ebenfalls derjenige Parameter, bei welchem die geringste Standardabweichung über die zehn Evolutionsdurchläufe zu beobachten ist. Der am höchsten gewichtete positive Parameter aus Simulation 1 ist r_{13}, wobei dieser auch der im Durchschnitt am stärksten gewichtete positive Parameter ist. Dabei wurde bei allen zehn Simulationen das positive Vorzeichen evolviert. Spieler werden durch den Parameter sehr stark als Spion verdächtigt, weil sich ihr Score in der Buchführung um den Parameterwert r_{13} erhöht. Die evolvierte Gewichtung und das Vorzeichen ist sehr nachvollziehbar, da dieser Parameter eingesetzt wird, sobald ein anderer Spieler für ein Team der Größe drei stimmt, für welches er jedoch nicht als Teammitglied vorgeschlagen wurde. In der Implementierung ist bereits vorgegeben, dass ein Widerstandskämpfer in solch einer Situation mit „Nein" stimmen würde. Demnach ist ein Spieler, der stattdessen mit „Ja" gestimmt hat, tatsächlich ein Spion und wird mit dem Parameterwert r_{13} belangt. Der zweithöchste positive Parameter in Simulation 1 und im Mittelwert ist r_{10}. Auch dies erscheint sinnvoll, da in der Implementierung vorgegeben ist, dass ein Widerstandskämpfer bei einem fünften Wahlversuch mit „Ja" stimmt. Ein Spieler, der mit „Nein" gestimmt hat, muss demnach ein Spion sein und wird für sein Verhalten mit Parameterwert r_{10} bestraft. Auch hier wurde in allen zehn Simulationen ein positives Vorzeichen evolviert.
Die stärkste Belohnung erhalten Spieler durch r_{12}. Ein Spieler, der ein Team bei der Wahl

ablehnte, welches eine sabotierte Mission hervorbrachte, wird in den Augen des Widerstandskämpfers demnach als gut bewertet. In neun von zehn Evolutionsdurchläufen wurde hierfür ein negatives Vorzeichen evolviert. Die Standardabweichung ist bei diesem Parameter jedoch mit 0.414 am höchsten, was daran liegt, dass ein positiver Wert in einer der Simulationen evolviert wurde. Ohne den Ausreißer würde der Mittelwert bei -0.819 und die Standardabweichung bei 0.248 liegen.
Interessant ist es außerdem, die Ergebnisse für Parameter r_6 und r_7 zu betrachten. Der Parameter r_6 wurde für die Teammitglieder eines Teams der Größe drei verwendet, welches keine Sabotage lieferte und der Buchführungs-Spieler selbst aber nicht Mitglied dieses Teams war. Wie man erwarten würde, wurden die Teammitglieder bestraft, weil unter diesen auf jeden Fall mindestens ein Spion sein muss. Im Durchschnitt erhöht sich ihr Score jeweils um 0.252 und auch in neun von zehn Evolutionen wurde ein positiver Parameterwert evolviert. Der Teamleader, der das eben beschriebene Team erstellt hat, wird zusätzlich mit Parameter r_7 bewertet. Dieser wird jedoch im Durchschnitt mit einem negativen Parameterwert belohnt. Die evolvierten Vorzeichen waren bei den zehn Simulationen aber weniger oft übereinstimmend. Parameter, bei welchen die Vorzeichen in allen oder in fast allen Evolutionsdurchläufen übereinstimmen, geben mehr Sicherheit, ob der Parameter zur belohnenden oder bestrafenden Bewertung eines Spielers eingesetzt werden sollte. Unabhängig davon, was die Strategie des Spions ist, sind die entsprechenden Belohnungen oder Bestrafungen immer sinnvoll, was vor allem bei r_{10} und r_{13} logisch erscheint. Bei Parametern, für welche kein einheitliches Vorzeichen evolviert wurde, wird die Evolution dieser Parameter vermutlich stärker von dem Verhalten der Spione beeinflusst und je nach evolvierter gegnerischer Strategie daher auch ein unterschiedliches Ergebnis für den Parameter erzielt. Die Strategie wird an die Gegner-Strategie angepasst.

Für den Spion wurden Wahrscheinlichkeiten evolviert. Die Standardabweichungen sind für die Parameter s_2 und s_5 am geringsten. Der Parameter s_2 gibt an, mit welcher Wahrscheinlichkeit der Spion bei einem fünften Wahlversuch mit „Ja" stimmt. Da der Widerstandskämpfer lernt, ein Wählen mit „Nein" zu bestrafen, ist es nicht überraschend, dass der Spion lernt, mit einer durchschnittlichen Wahrscheinlichkeit von 97.3% mit „Ja" zu stimmen. In Simulation 1 liegt der Wert bei 98%. Der Parameter s_5 steht für die Wahrscheinlichkeit, mit welcher der Spion für ein Team stimmt, in welchem keine Spione sind. Die evolvierte durchschnittliche Wahrscheinlichkeit liegt bei 3.8%. Das heißt also, dass ein Spion fast immer gegen ein Team ohne Spione wählt. Für die Parameter s_6 und s_7 ist die Standardabweichung auch noch verhältnismäßig gering. Mit Wahrscheinlichkeit s_6 sabotiert ein Spion, wenn zwei Spione im Team sind und ein Team der Größe zwei vorliegt. s_7 wurde analog für ein Team der Größe drei verwendet. Die Werte liegen im Durchschnitt bei 14.8% für s_6 und 27.4% für s_7. Die evolvierten Werte aus Simulation 1 unterscheiden sich davon kaum. Der Spion sabotiert in solch einer Situation nicht so häufig, was sinnvoll erscheint. Es ist logisch nach-

vollziehbar, dass der Spion lernt, bei einem Team der Größe zwei noch vorsichtiger bei der Sabotage zu sein als bei einem Team der Größe drei, da im ersten Fall die Identität beider Spione enthüllt werden könnte, wenn beide Spione sabotieren.
Der Parameter s_8 unterliegt einer sehr hohen Abweichung zwischen den Simulationen. Er gibt an, mit welcher Wahrscheinlichkeit der Spion als einziger Spion im Team in der ersten Mission sabotieren möchte. Im Durchschnitt geschieht dies in 56% der Fälle. Die hohe Standardabweichung lässt darauf schließen, dass es verschiedene gute Strategien gibt. Bei der einen wird in der ersten Mission mit hoher Wahrscheinlichkeit sabotiert, um bereits einen Punkt zu erhalten. Bei einer anderen Strategie wird nicht sabotiert, um in Augen der Widerstandskämpfer mit dem Parameter r_3 oder r_4 bewertet und damit vielleicht belohnt zu werden. Dass verschiedene Strategien zugrunde liegen können, gilt auch für die anderen Parameter mit einer hohen Standardabweichung.
Mit einer im Durchschnitt hohen Wahrscheinlichkeit s_9 wird der Spion als einziger Spion im Team in allen anderen Missionen sabotieren, wie man es auch erwarten würde. Bei den Parametern s_0 und s_1, welche für die Erstellung des Teams verwendet werden, ist es durchaus sinnvoll, dass s_0 einen niedrigeren Wert als s_1 annimmt, damit bei einem Team der Größe zwei möglichst kein weiterer Spion mit ausgewählt wird und bei einem Team der Größe drei dies öfter geschehen darf. Die Wahrscheinlichkeit, dass der Spion für ein Team stimmt, in dem sich ein Spion befindet, ist bei dem evolvierten Parameter s_3 überraschend gering. Das Ergebnis könnte dadurch erklärt werden, dass die Widerstandskämpfer Spieler mit dem Parameter r_{12} belohnen, wenn der Spieler gegen ein Team gestimmt hat, durch das die Mission sabotiert wurde. Noch kleiner ist die Wahrscheinlichkeit s_4, für ein Team mit zwei Spionen zu stimmen. Bei Simulation 1 wird nie für ein Team gestimmt, in dem sich zwei Spione befinden.

Dass insgesamt sehr hohe Abweichungen zwischen Parameterwerten aus allen zehn Simulationen bei den Spionen und Widerstandskämpfern vorhanden sind, deutet darauf hin, dass es keine optimale Strategie für das Modell gibt. Es werden zum Teil verschiedene Strategien evolviert und es gibt keine allgemeine Strategie, welche besser als alle anderen Strategien ist. Man kann im Durchschnitt bei allen Simulationen einen deutlichen Lernfortschritt feststellen, der aber zum Teil zu anderen Ergebnissen führt. Die Strategien werden passend zur gegnerischen Strategie adaptiert. Die Belegung derjenigen Parameter, bei welchen sehr geringe Standardabweichungen vorliegen und welche demnach in allen Simulationen sehr ähnlich evolviert wurden, entspricht einer allgemein guten oder annähernd optimalen Parameterwahl für die verschiedenen Strategien.

5.4.4 Evolvierter Widerstandskämpfer und Spion im Spiel

Der beste Widerstandskämpfer und der beste Spion aus Generation 200 würden bei optimaler Anpassung an die gegnerische Strategie jeweils 50% der Spiele gewinnen, wenn sie gegeneinander antreten. Dann ist das Spiel fair. Wenn der evolvierte Widerstandskämpfer und Spion

aus Simulation 1, für welche die Parameterwerte aufgelistet wurden, 10.000 Partien spielen, kann man feststellen, dass die Widerstandskämpfer etwa 44% und die Spione 56% der Partien gewinnen. In Abbildung 5.2b wird dieser Test mit 100 durchgeführten Spielen durch den Pixel an der Stelle $(200, 200)$ visualisiert. Der zugrundeliegende Wert im CIAO-Datensatz beträgt dort 43% Gewinnhäufigkeit für den Widerstandskämpfer. Führt man diesen Test auch bei den evolvierten Individuen der anderen Simulationen durch, so gibt es Ergebnisse, bei welchen die Spione sogar 60% oder 70% der Spiele gewinnen. Dies deutet darauf hin, dass der Spion in der Evolution einen Vorteil hat. Der Suchraum für die Parameter des Spions ist begrenzt auf das Intervall $[0, 1]$, wohingegen der des Widerstandskämpfers unbegrenzt ist, da hier das Verhältnis zwischen den Parametern entscheidend ist. Das kann dazu führen, dass der Spion schneller in der Lage ist, geeignete Strategien zu evolvieren, an die sich der Widerstandskämpfer erst anpassen muss. Dass es für den Widerstandskämpfer in diesem Modell aber grundsätzlich möglich ist 50% der Spiele oder mehr zu gewinnen, zeigen viele der dunklen Datenpunkte in Abbildung 5.2. Was man in Abbildung 5.2b außerdem sieht, ist dass der Spion etwa ab Generation 150 einen Vorteil gegenüber den Widerstandskämpfern hat, wodurch ein etwas hellerer Block oben rechts in der Abbildung entsteht. Genau das lässt sich durch die Elite Bitmap in Abbildung 5.4 erklären, in der man sieht, dass der Genotyp des besten Spions etwa bei Generation 150 (entspricht 38 auf der Y-Achse) eine offensichtlich vorteilhafte Veränderung erhält. Der Genotyp des Widerstandskämpfers verändert sich von diesem Zeitpunkt an jedoch nicht markant bis zum Ende der Evolution, was bedeutet, dass eine Anpassung an die veränderte Spion-Strategie nicht mehr stattgefunden hat. Eventuell fehlen dem Widerstandskämpfer hier auch noch weitere Parameter, die eine bessere Adaption an die gegnerische Strategie ermöglichen.

5.5 Fazit und Ausblick

Durch die Messmethoden konnte gezeigt werden, dass ein deutlicher Lernfortschritt der Widerstandskämpfer und Spione im Mittel über die zehn Evolutionsdurchläufe erreicht wurde. Bei einer Co-Evolution ist die Fitness eines Individuums von den anderen Individuen abhängig. In den Ergebnissen ist ganz deutlich zu sehen, dass eine steigende Fitness für den Widerstandskämpfer eine fallende für den Spion bedeutet und umgekehrt. Weiterhin gibt es lange Phasen, in denen die Fitness beider Populationen weitaus konstant bleibt. Wenn man nur die Fitness betrachten würde, könnte man meinen, dass sich die Strategien bereits nach sehr wenigen Generationen kaum noch verbessern. Dass dies nicht der Fall ist, wurde durch das Master-Tournament und die CIAO-Daten gezeigt. Die überwiegend konstant bleibende Fitness der beiden Populationen ist damit auf den Red-Queen-Effekt zurückzuführen. Es findet über viele Generationen hinweg ein Lernprozess statt, obwohl sich dieser nicht in der Fitness widerspiegelt.
Eine optimale Strategie konnte für dieses Modell jedoch nicht gefunden werden, was man

anhand der abweichenden Ergebnisse in den zehn Simulationen feststellen kann. Nur ein paar wenige Parameter zeigten ein sehr ähnliches Ergebnis, was wahrscheinlich einer optimalen Parameterbelegung entspricht. Die anderen Parameter werden mehr von der gegnerischen Strategie beeinflusst und im Laufe der Evolution an diese angepasst. Es besteht die Vermutung, dass es für dieses Modell keine optimale Strategie gibt. Wenn eine allgemein beste Strategie nicht existiert, dann ist es das beste, die passende Strategie zur gegnerischen Strategie auszuwählen [7]. Dies geschieht in den durchlaufenen Simulationen. Dabei ist der beste evolvierte Widerstandskämpfer am Ende der Evolution nicht immer so gut wie der Spion, was vermutlich daran liegt, dass die Widerstandskämpfer einen größeren Suchraum als die Spione besitzen und die Evolution ein schnelleres sinnvolles Evolvieren der Spione ermöglicht. Vielleicht aber fehlen dem Widerstandskämpfer noch geeignete Parameter, die eine flexiblere Adaption an die gegnerische Strategie erst ermöglichen. Man könnte versuchen, die Evolution oder das Modell auf dieses Problem hin besser anzupassen und den genauen Grund zu ermitteln. Vielleicht sind auch andere Einstellungen der Parameter für die Evolutionsstrategie noch besser geeignet.

Die in dieser Arbeit verwendete Fitnessfunktion arbeitet mit der Hall-of-Fame, wobei immer nur ein Widerstandskämpfer-Genotyp gegen einen Spion-Genotyp im Spiel antritt. In der Co-Evolution wurden die Individuen also danach bewertet, wie gut sie spielen, wenn alle Mitspieler, die zur eigenen Gruppe gehören, nach derselben Strategie spielen. Man könnte dies als eine „Team-Spieler-Evaluierung" bezeichnen. Ebenfalls denkbar ist eine Evaluierung mit der Hall-of-Fame, bei der verschiedene Genotypen verwendet werden. Um die Individuen aus der aktuellen Population der Widerstandskämpfer zu evaluieren, würde man für ein Spiel immer drei verschiedene Widerstandskämpfer aus der Population auswählen und zwei zufällige Spione aus der Hall-of-Fame. Analog würden für die Fitness-Berechnung der Spione zwei verschiedene Spione ausgewählt werden und drei zufällige Widerstandskämpfer aus der Hall-of-Fame. Diese Variante wurde ebenfalls ausprobiert, die Ergebnisse im Rahmen dieser Bachelorarbeit jedoch nicht weiter ausgewertet. Daher soll an dieser Stelle nur kurz darauf eingegangen werden, dass auch bei dieser Evaluierungsmethode ein Lernfortschritt erreicht werden kann. Dies zeigt Abbildung 5.5, welche den gemittelten CIAO-Datensatz über zehn Evolutionsdurchläufe visualisiert. Die Parameter für die Evolutionsstrategie wurden hierbei genauso wie in Tabelle 5.2 gewählt.
Eine weitere Möglichkeit, welche nicht ausprobiert wurde, ist die Verwendung nur einer Population, wobei ein Individuum aus den beiden Parametersätzen für Spion und Widerstandskämpfer besteht und durch die zufällige Rollenzuweisung nach den jeweiligen Parametern handelt. Die Fitness bewertet, wie gut das Individuum insgesamt mit seinen beiden Rollen-Strategien ist. Auch wenn nur eine Population vorliegt, handelt es sich dabei nach wie vor um eine Co-Evolution, weil die Fitness immer von anderen Individuen abhängig ist (und diese dürfen auch in derselben Population sein).

Des Weiteren könnte man ein noch besseres Modell entwickeln. Die Wahlen, welche zu keiner Wahlmehrheit geführt haben, geben viele wertvolle Informationen über die Spieler preis. Diese wurden für das in dieser Arbeit verwendete Modell jedoch nicht berücksichtigt. Interessant wäre es auch, zu untersuchen, wie sich die Strategien in der Evolution entwickeln, wenn eine andere Spieleranzahl verwendet wird. In dem hier verwendeten Modell wurden die Spielzüge speziell auf fünf Spieler ausgerichtet. Man könnte versuchen, ein allgemeineres Modell zu entwickeln. Die Kommunikationsebene mit einzubeziehen ist eine weitere Herausforderung. Es besteht jedoch die Vermutung, dass je komplexer die Modellierung ist, es umso schwieriger wird, eine optimale Strategie für das Modell zu finden.
Zu beachten ist außerdem, dass die auf Basis des Modells evolvierten Strategien unter Umständen nur für dieses Modell gut geeignet sind. Wie tauglich die evolvierten Strategien sind, wenn sie gegen andere Modelle oder menschliche Spieler antreten, könnte weiterführend auch noch untersucht werden.

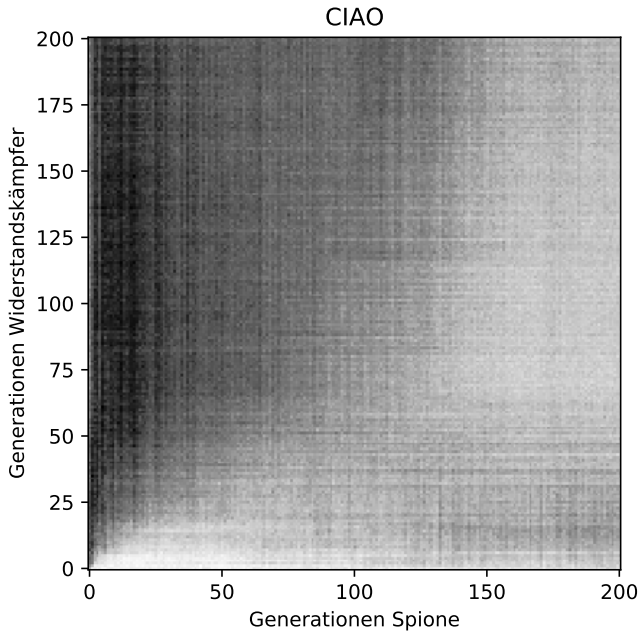


Abbildung 5.5: Gemittelter CIAO-Datensatz über zehn Simulationen mit einer anderen Hall-of-Fame-Evaluierung

Literatur

1. Eskridge, D. *The Resistance* (Indie Boards und Cards, 2010).
2. Gerdes, I., Klawonn, F. & Kruse, R. *Evolutionäre Algorithmen* 1. Aufl. ISBN: 978-3-528-05570-7 (Vieweg+Teubner Verlag, Juli 2004).
3. Ebner, M. *Evolutionäre Algorithmen* Vorlesungsskript. Universität Greifswald, Institut für Mathematik und Informatik. 2011.
4. Kruse, R. *Evolutionäre Algorithmen* Vorlesungsskript. Otto-von-Guericke-Universität Magdeburg, Fakultät für Informatik. Apr. 2010.
5. Kaderali, L. *Modul Biometrie, Mathematische Modelle in der Biometrie und der biomedizinischen Forschung, 13. Vorlesung: Genetische Algorithmen* Vorlesungsskript. Universität Greifswald, Institut für Bioinformatik.
6. Ebner, M. Coevolution and the Red Queen effect shape virtual plants. *Genetic Programming and Evolvable Machines* **7,** 103–123 (März 2006).
7. Nolfi, S. & Floreano, D. Coevolving Predator and Prey Robots: Do "Arms Races" Arise in Artificial Evolution? *Artificial Life* **4,** 311–335 (Okt. 1998).
8. Ridley, M. *The Red Queen: Sex and the Evolution of Human Nature* ISBN: 978-0140167726 (Penguin Books, 1994).
9. Van Valen, L. A new evolutionary law. *Evolutionary Theory* **1,** 1–30 (Juli 1973).
10. Cliff, D. & Miller, G. Tracking The Red Queen: Measurements of adaptive progress in co-evolutionary simulations. *Advances in Artificial Life,* 200–218 (1995).
11. Floreano, D. & Nolfi, S. *God Save the Red Queen! Competition in Co-Evolutionary Robotics* in *GENETIC PROGRAMMING 1997: PROCEEDINGS OF THE SECOND ANNUAL CONFERENCE* (Morgan Kaufmann, 1997), 398–406.
12. Floreano, D. & Mattiussi, C. *Bio-Inspired Artificial Intelligence: Theories, Methods, and Technologies* ISBN: 978-0-262-06271-8 (MIT Press, 2008).
13. Hillis, W. D. Co-evolving parasites improve simulated evolution as an optimization procedure. *Physica D: Nonlinear Phenomena* **42,** 228–234 (Juni 1990).
14. Rosin, C. D. & Belew, R. K. New Methods for Competitive Coevolution. *Evolutionary Computation* **5,** 1–29 (März 1997).

15. *DEAP documentation* https://deap.readthedocs.io/en/master/.