Karlsruher Institut für Technologie

Effiziente Berechnung des ε-Nash-Equilibriums einer Poker-Variante

Diplomarbeit
von

Holger Kujath

an der Fakultät für Informatik

Erstgutachter: Prof. Dr. R. Studer
Zweitgutachter: Prof. Dr. Hartmut Schmeck
Betreuender Mitarbeiter: Dr. Max Völkel

Bearbeitungszeit: 27. März 2014 – 27. September 2014

„No matter what problem you encounter, whether it's a grand challenge for humanity or a personal problem of your own, there's an idea out there that can overcome it. And you can find that idea.“

— Ray Kurzweil

Danksagungen

Mein Dank gilt Herrn Prof. Dr. Rudi Studer, der mir die Möglichkeit gab, das Thema meiner Diplomarbeit selbst wählen zu dürfen und über das Vorgehen eigenständig zu entscheiden.

Ich danke meinem Betreuer Dr. Max Völkel für seine große Hilfsbereitschaft und Unterstützung. Durch seine vorausschauende Organisation konnte die Qualität der Arbeit gewährleistet werden und mithilfe seiner Anmerkungen wurde die Verständlichkeit wesentlich verbessert.

Meinem Freund Mathias danke ich für seine Bereitschaft zur tiefgehenden, inhaltlichen Auseinandersetzung mit dem Thema dieser Arbeit, dem Beitrag eigener Ideen und des Korrekturlesens der ersten Version.

Weiterhin gilt mein Dank meinem Freund Mirko, der mich bei der grafischen Gestaltung einiger Abbildungen unterstützte.

Meiner Freundin Isabel danke ich für das finale Korrekturlesen des Textes.

Mein besonderer Dank gilt meiner Familie, insbesondere meinen Eltern Regina und Günter Kujath, die mich in all meinen Entscheidungen unterstützt haben.

Holger Kujath

holger.kujath@knuddels.de

1 Einleitung

1.1 Problem

Die weltbesten Schachspieler unterlagen im letzten Jahrzehnt der Maschine, also der neuesten Schachsoftware auf einem leistungsstarken Rechner, chancenlos [1], hingegen konnten beim Poker die letzten Begegnungen zwischen Mensch und Maschine ausgeglichen gestaltet werden [2]. Und das, obwohl monetäre Anreize zur Entwicklung einer starken Pokersoftware vorhanden sind, denn im Gegensatz zu Schach wird bei Poker in der Regel um Geld gespielt. Im Jahr 2010 gab es weltweit allein sechs Millionen verschiedene Spieler, die Poker im Internet um Geld gespielt haben. Die dabei von den Betreibern der Online-Poker-Plattformen eingenommenen Gebühren (sog. *Rake*) betrugen im gleichen Jahr 3,6 Milliarden US-Dollar [3]. Unter der Annahme, dass diese Gebühren bei ca. 5% des von den Spielern eingesetzten Geldes liegen, beträgt die jährliche Geldsumme, um die im Online-Poker gespielt wird, ca. 70 Milliarden US-Dollar.

Im Gegensatz zum Schach, bei dem jeder Spieler alle Figuren sieht und damit über vollständige Informationen verfügt, kennt ein Pokerspieler die Karten seiner Gegner nicht. Darüber hinaus beinhalten die meisten Poker-Varianten Zufallselemente, durch die im späteren Spielverlauf nicht vorhersehbare, zufällige Karten ins Spiel kommen. Diese Unterschiede führen zu einer wesentlich größeren Komplexität in der Modellierung im Vergleich zu Schach. Die Komplexität, ausgedrückt in der Anzahl der Spielzustände, der in dieser Arbeit behandelten zwei Spieler Poker-Variante „*Texas No Limit Hold'em Heads up*“ (NLHE HU) entspricht 10^{71} [4]. Im Vergleich dazu hat Schach eine Komplexität von 10^{47} [5]. Für die Forschung im Bereich der künstlichen Intelligenz [6], sowie der angewandten Spieltheorie ist es deshalb eine große Herausforderung eine Pokersoftware zu entwickeln, die nicht nur stärker als ein menschlicher Profi spielt, sondern im besten Fall die spieltheoretisch optimale Strategie, das sogenannten Nash[1]-Equilibrium, verwendet. Schon damals in seinem berühmten Beweis aus dem Jahr 1951 [7] schlussfolgerte der Nobelpreisträger John Nash, dass „die Analyse eines realistischeren Pokerspiels, welches über unser vereinfachtes Model hinaus ginge, eine interessante Angelegenheit“ sein müsste.

1.2 Lösungsidee

Jedes Pokerspiel lässt sich mithilfe eines Baumes darstellen, bestehend aus Knoten in denen entweder 1) ein Spieler eine Entscheidung trifft, 2) zufällige, neue Karten ins Spiel kommen oder 3) die *Spielrunde* endet und der *Gewinn* ausgezahlt wird.

Für jeden Spieler gibt es eine große Menge an möglichen Strategien. Eine Strategie beschreibt für jeden Knoten eines konkreten Spielers, welche Handlungsalternative bzw. Aktion gewählt wird. Man spricht von einer gemischten Strategie, wenn durch die Strategie nicht nur eine bestimmte Handlungsalternative ausgewählt wird, sondern aus einer Menge mit mehreren Handlungsstrategien mit jeweils eigener Wahrscheinlichkeit ausgewählt wird.

Aufgrund der Zufallselemente und der unvollständigen Information, die für Poker charakteristisch sind, wird selbst ein Spieler mit einer optimalen Strategie nicht jede *Spielrunde* gewinnen. Entscheidend ist daher nicht der Ausgang einzelner *Spielrunde*n, sondern der Erwartungswert des Gewinns bzw. der erwartete Nutzen.

[1] Benannt nach dem Mathematiker und Nobelpreisträger John Forbes Nash Jr., der diese Eigenschaft einer Strategie als Erster entdeckte

Da bei Poker die Spieler in einer vorgegebenen Reihenfolge handeln, sind ihre Positionen innerhalb des Spiels nicht symmetrisch. Daher wird für jeden Spieler eine eigene Strategie benötigt.

Die Lösungsidee zur Berechnung optimaler Strategien für jeden Spieler besteht aus einem iterativen Algorithmus, der sich dem Nash-Equilibrium durch ein iteratives Verfahren bis auf ein sehr kleines ε annähert. Das ε gibt in diesem Fall an, wie weit der erwartete Nutzen vom spieltheoretischen Optimum, dem Nash-Equilibrium, entfernt ist. Da Poker ein Null-Summenspiel ist, entspricht ε der Summe der maximal zu erwartenden Verluste der Strategien in einer *Spielrunde*.

Um das Nash-Equilibrium zu bestimmen wird für jeden Spieler vor der ersten Iteration eine naive Strategie (z.B. immer dieselbe Aktion) angenommen und anschließend verändert der Algorithmus die Strategie eines Spielers so, dass sich der erwartete Nutzen dieses Spielers erhöht. Dazu wird für jeden Knoten, in dem der Spieler eine Entscheidung trifft, die Auswahlwahrscheinlichkeit derjenigen Handlungsalternative erhöht, die den größten, erwarteten Nutzen verspricht. Anschließend wird die Strategie des anderen Spielers verbessert, so dass die Strategien beider Spieler sich jeweils einander anpassen und mit jeder weiteren Iteration das ε reduziert wird. In dieser Arbeit wird gezeigt, dass ε durch das Verfahren mit steigender Anzahl an Iterationen gegen Null konvergiert. Die so entwickelte Strategie ist in der Hinsicht optimal, dass sie selbst für den ungünstigsten Fall, nämlich den, dass der Gegenspieler die optimale Gegenstrategie wählt, im Erwartungswert nur einen Verlust von ε erzielt. Aufgrund dieser defensiven Ausrichtung durch das Ziel der Minimierung des maximalen Verlusts wird eine gemischte, aber statische Strategie für jede konkrete Spielsituation berechnet. Da diese Strategie statisch ist, können Schwächen einen konkreten Gegners nicht gezielt exploitiert werden.

Gegen die vom Verfahren errechnete Strategie kann die optimale Gegenstrategie im Mittel höchstens einen Gewinn von ε erzielen. Für ein entsprechend kleines ε werden daher die vom Verfahren berechneten Strategien professionellen Pokerspielern überlegen sein.

1.3 Aufgabenstellung

Ziel dieser Arbeit ist die Entwicklung einer Pokersoftware für die Pokervariante *Texas No Limit Hold'em* mit zwei Spielern (*Heads up*). Dabei Handelt es sich um die weltweit beliebteste Variante. Die Software soll die Strategie iterativ dem Nash-Equilibrium annähern, wobei der Abstand ε zum Nash-Equilibrium gegen Null konvergiert.

Zur praktischen Umsetzbarkeit wird das Problem folgendermaßen eingeschränkt:

- Die Strategien werden nur für die zweite bis zur letzten *Wettrunde* (*Flop* bis *River*) berechnet.
- Die Anzahl verschiedener Einsatzhöhen ist wie auch die Anzahl des Spielgeldes (*Stacks*) sinnvoll zu beschränken.
- Die Laufzeit der Software muss so effizient sein, dass sich im Rahmen der Diplomarbeit sinnvolle Evaluierungsergebnisse erzielen lassen. Die dazu anzustrebenden, technischen Optimierungen sind zu dokumentieren.
- Es werden keine Turniere unterstützt, sondern jede Pokerrunde einzeln betrachtet (*Cashgame*).

2 Hintergrund

2.1 Spiele

Mit Entstehung der ersten Kulturen hat der Mensch auch die ersten Spiele erfunden. Das älteste, bekannte Spiel heißt „Mehen“. Es wurde in der Oberschicht des Alten Ägypten in der Zeit um 3000 v. Chr. gespielt. Ziel eines Spiels war es, mit seinen Spielsteine als Erster auf ein bestimmtes Spielfeld zu ziehen. [8]

Bei der Frage nach der Natur von Spielen hat sich in der Literatur bisher keine Definition durchsetzen können. Betrachtet man die gängigsten Definitionen, so verbindet alle miteinander, dass ein Spiel eine künstliche Welt erzeugt in der Spieler miteinander agieren, sich an Spielregeln halten und versuchen ein Spielziel zu erreichen. Diese Grundelemente werden von folgender modernen Definition abgedeckt.

"A game is a system in which players engage in an artificial conflict, defined by rules, that results in a quantifiable outcome." [9]

2.1.1 Extensivform

Die Extensivform (auch vollständiger Spielbaum genannt) ist eine spieltheoretische Darstellung eines Spiels. Im Gegensatz zur Normalform, die ein Spiel in einer Matrix darstellt bei der alle Spieler gleichzeitig eine Entscheidung treffen, ermöglicht die Extensivform die Modellierung sequenzieller Abfolgen von Entscheidungen im Spiel. Ihr wichtigstes Element ist ein gerichteter Graph, auch Spielbaum genannt. Der Spielbaum stellt visuell in einem Spiel von Anfang bis Ende alle möglichen Abläufe und Ergebnisse dar. Dies beinhaltet dabei sowohl die Entscheidungen der Spieler als auch durch Spielregeln ausgewählte, zufällige Entscheidungen.

In Abbildung 1 ist eine vereinfachte Poker Situation als Spielbaum dargestellt. Der Spieler A setzt alle seine *Chips* (*All-In*) und Spieler B kann aufgrund unvollständiger Informationen nicht unterscheiden, ob die Aktion von Spieler A ein *Bluff* ist oder nicht.

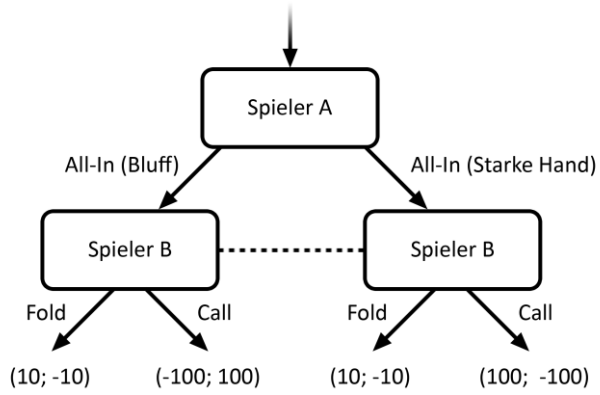


Abbildung 1: Spielbaum, bei dem Spieler B nicht unterscheiden kann, ob Spieler A *bluff*t oder eine starke Hand hält.

Ein endliches, strategisches Spiel in Extensivform besteht aus folgenden Elementen: [10, 11]

- Einer endlichen Anzahl an **Spielern**.
- Dem endlichen **Spielbaum**, bestehend aus einem Wurzelknoten (dem Start) und weiteren Knoten. In Blättern wird der Gewinn verteilt, in allen anderen Knoten wird durch einen Spieler bzw. per Zufall eine Aktion gewählt.
- Kanten verbinden Knoten und entsprechen **Aktionen** (von Spielern oder dem Zufall).
- Eine Menge von **Vorgeschichten**, die allen möglichen Pfaden im Spielbaums von der Wurzel zu einem Knoten entsprechen. Terminale Vorgeschichten sind Vorgeschichten, die mit einem Blatt enden. Die **sichtbaren Vorgeschichten** eines Spielers bestehen nur aus Aktionen, die vom Spieler wahrgenommen werden können. Somit sind dem Spieler unbekannte Aktionen (wie z.B. Karten des Gegners) nicht enthalten.
- Jeder Endknoten des Spielbaums besitzt ein n-Tupel, dass den **Gewinn** (oder Verlust) für alle Spieler bei Erreichen dieses Endknotens darstellt.
- Einer Teilmenge an Knoten für jeden Spieler, in denen der jeweilige Spieler an der Reihe ist, eine Aktion auszuwählen. Dies sind die **Knoten des Spielers**. Außerdem sind diese Teilmengen paarweise disjunkt.
- Alle Knoten, die nicht in einer der vorgenannten Teilmengen vorkommen, sind so genannte **Zufallsknoten**. In diesen Knoten gibt es für jede mögliche Aktion eine Wahrscheinlichkeit. Unter Berücksichtigung dieser Wahrscheinlichkeiten wählt das Spiel zufällig eine Aktion aus.
- Eine Menge an **Informationsmengen** für jeden Spieler. Jede Informationsmenge ist eine Menge von Knoten (also Spielzuständen), die aus Sicht des Spielers äquivalent sind. Er kann Knoten derselben Informationsmenge mit den ihm zur Verfügung stehenden Informationen nicht voneinander unterscheiden (wie in Abbildung 1). Zum Beispiel ist es bei Poker einem Spieler nicht möglich zu unterscheiden, welche Karten sein Gegner besitzt. In diesem Fall gibt es für jede Kombination an eigenen Karten eine Informationsmenge, die alle möglichen Kartenkombinationen des Gegners enthält. Spieler können also Informationsmengen aber nicht Spielzustände innerhalb einer Informationsmenge voneinander unterscheiden.
- Die durch die diese Elemente definierte Struktur des Spiels ist allen Spielern vollständig bekannt.

Unter Verwendung der Informationsmengen kann die Komplexität des Spielbaums reduziert werden. Für jeden Spieler gibt es einen **Informationsmengenbaum**, in dem jeder Knoten genau einer Informationsmenge entspricht. Dieser Baum hat wesentlich weniger Knoten, als der Spielbaum, da er nur dem Spieler bekannte Informationen enthält.

Diese Arbeit führt mit dem **reduzierten Baum** eine neue Darstellungsform ein. Der reduzierte Baum enthält nur noch Informationen, die beiden Spielern bekannt sind. Damit ist dessen Komplexität noch einmal geringer, als die eines Informationsmengenbaumes. Die Vorteile sind die geringere Komplexität und die Begrenzung auf einen Baum, der für beide Spieler verwendet werden kann. Die privaten Informationen jedes Spielers (z.B. seine *Handkarten*) werden nicht im Baum, sondern sind in der jeweiligen Strategie enthalten.

In dieser Arbeit sind außerdem noch folgende Eigenschaften von Spielen wichtig:

Null-Summen-Spiele Ein Null-Summen-Spiel ist ein Spiel, bei dem die Summe der Gewinne aller Spieler in jedem Endknoten genau Null beträgt.

Indeterministische Spiele Ein deterministisches Spiel, wie z.B. Schach, ist ein Spiel ohne Zufallsknoten. Dementsprechend besitzt ein indeterministisches Spiel mindestens einen Zufallsknoten.

Imperfekte Informationen Bei einem Spiel mit perfekten Informationen bestehen die Informationsmengen aller Spieler genau aus einem Knoten bzw. Spielzustand. Jeder Spieler verfügt über genügend Informationen, um alle Spielzustände voneinander zu unterscheiden. Im Falle von imperfekten Informationen existiert mindestens eine Informationsmenge mit mehreren Knoten. In welchem dieser Knoten sich das Spiel gerade tatsächlich befindet, ist vom Spieler nicht eindeutig unterscheidbar.

2.1.2 Strategien

Die **gemischte Strategie** eines Spielers legt für jede seiner Informationsmengen eine Wahrscheinlichkeitsverteilung fest. Eine solche Wahrscheinlichkeitsverteilung gibt für jede mögliche Aktion des Spielers an, mit welcher Wahrscheinlichkeit diese von ihm ausgewählt wird. Im Unterschied dazu wählt ein Spieler mit einer **reinen Strategie** für jede Informationsmenge immer nur eine bestimmte Aktion mit Wahrscheinlichkeit eins aus.

Ein **Strategieprofil** enthält für jeden Spieler genau eine Strategie.

Der erwartete Nutzen einer Strategie A gegen eine bestimmte andere Strategie entspricht dem im Mittel zu erwartenden Gewinn von A. Bei gegebenem Strategieprofil, also der Kenntnis der Strategien aller Spieler, kann der erwartete Nutzen jeder Strategie des *Strategieprofil*s berechnet werden [12]. Dazu wird eine Tiefensuche durch den Spielbaum durchgeführt, bei der die Gewinne jedes Spielers in den Endknoten mit den Wahrscheinlichkeitsverteilungen der Strategien in den Spielerknoten und den Zufallsknoten gewichtet und aufsummiert werden.

Ein Programm welches eine **statische Strategie** verwendet, hat für jede Aktion eines Spielers eine vorher festgelegte Wahrscheinlichkeit, mit der diese Aktion gewählt wird. Im Gegensatz dazu würde ein Programm mit adaptiver Strategie sein Verhalten dem Gegner anpassen.

2.1.3 Nash-Equilibrium

Der Begriff Equilibrium ist in vielen Wissenschaften wie der Physik, Chemie, Biologie und der Wirtschaft von zentraler Bedeutung. Ganz allgemein ist ein Equilibrium der Zustand eines Systems, in dem sich alle existierenden Kräfte ausgleichen.

In der Spieltheorie ist ein Strategieprofil ein **Nash-Equilibrium**, wenn es für keinen Spieler möglich ist, einen höheren erwarteten Nutzen durch Änderung seiner Strategie zu erzielen [13]. Dieses umfassende Konzept hatte sowohl Einfluss auf die Wirtschaftswissenschaften und die Soziologie als auch auf die Naturwissenschaften [14].

Ein Strategieprofil ist ein **ε-Nash-Equilibrium**, wenn es für keinen Spieler möglich ist, durch eine Änderung seiner Strategie seinen erwarteten Nutzen um mehr als ε zu erhöhen.

Beim Nash-Equilibrium handelt es sich um eine Strategie, die darauf ausgelegt ist den Gewinn des Spielers selbst bei einem optimal spielenden Gegner zu maximieren. Bei einem Spiel mit zwei Spielern A und B tritt aus Sicht von Spieler A der ungünstigste Fall dann ein, wenn Spieler B die Strategie von Spieler A bekannt ist und B die optimale Gegenstrategie wählt. Diese optimale Gegenstrategie ist die Strategie, die den erwarteten Nutzen von B maximiert. Unter dieser Voraussetzung maximiert Spieler A seinen Nutzen, in dem er das Nash-Equilibrium als seine Strategie wählt. Hier zeigt sich auch die besondere Qualität des Nash-Equilibriums, das den höchsten, garantierten, erwarteten Nutzen realisiert.

Für den Fall, dass der Gegner nicht die optimale Gegenstrategie wählt, ist das Nash-Equilibrium in aller Regel nicht die Strategie, die den höchsten erwarteten Nutzen besitzt, da es die Fehler des Gegners nicht gezielt exploitiert [15].

Im Falle von nicht-kooperativen Null-Summen-Spielen ist das Nash-Equilibrium in der Praxis eine sehr wertvolle Strategie, weil der erwartete Nutzen größer gleich Null beträgt, sofern die Reihenfolge der Spieler zufällig variiert oder die konkrete Reihenfolge der Spieler keine Vor- oder Nachteile mit sich bringt.

2.2 Poker

Poker bezeichnet eine Familie an verschiedenen Kartenspielen, deren folgende Gemeinsamkeiten zu Grunde liegen:

- Es wird mit französischen Karten zu 52 Blatt gespielt.
  - Jede Karte hat zwei Dimensionen: Farbe und Wert.
  - Es gibt vier verschiedene Farben, deren Wertigkeit sich nicht unterscheidet (Symbole der vier Farben ♠, ♣, ♥, ♦).
  - Außerdem existieren dreizehn verschiedene, aufsteigend sortierte Werte (2, 3, 4, 5, 6, 7, 8, 9, T(en), J(ack), Q(ueen), K(ing), A(s)).
- Eine *Hand* besteht aus fünf Karten.
  - Jede *Hand* hat einen *Pokerrang*, welcher von der Seltenheit der Karten-Kombination determiniert wird.
  - Die Hand mit dem höchsten *Pokerrang* gewinnt.
  - Der Pokerrang der *Hände* aufsteigend aufgelistet:
    Karte mit höchstem Wert, ein *Paar* (zwei Karten mit gleichem Wert), zwei *Paare*, *Drilling*, *Straight* (5 Karten mit aufsteigender Wertigkeit), *Flush* (5 Karten mit derselben Farbe), *Full House* (*Paar* + *Drilling*), *Vierling*, *Straight Flush* (Straße + *Flush*)
  - Innerhalb desselben *Pokerrang*s wird nach Wert der übrigen Karten entschieden. Haben alle Karten denselben Wert liegt ein Unentschieden vor.
  - Stehen für einen Spieler mehr als fünf Karten zur Auswahl, dann gilt für den Spieler die Teilmenge von genau fünf Karten als Hand, die seinen *Pokerrang* maximiert.
- Poker ist ein sequenzielles Spiel, es gibt keine Aktionen, die gleichzeitig passieren.
- Poker ist ein Spiel mit unvollständigen Informationen.
  - Die *Handkarten* jedes Spielers sind geheim.
- Jeder Spieler verfügt über eine bestimmte Menge an Spielgeld (sog. *Stack*).
  - Das Spielgeld wird durch verschiedenfarbige *Chips* repräsentiert.
  - Beim *Cashgame* wird in Casinos vor dem Spiel echtes Geld in Spielgeld umgetauscht, das Spielgeld kann wieder zurück in echtes Geld getauscht werden.

    - Innerhalb einer *Spielrunde* gibt es mehrere *Wettrunden* auf denen alle Spieler *Chips* setzen können. NLHE besteht aus vier *Wettrunden*.
    - In der ersten Runde gibt es bei modernen Poker-Varianten erzwungene Einsätze (sog. *Blind*) für zwei Spieler. Nur durch die *Blind*s ist es für rational spielende Teilnehmer sinnvoll, auch ohne perfekte *Handkarten Chips* zu setzen.
    - Hat ein Spieler *Chips* gesetzt, kann der darauf folgende Spieler dieselbe Anzahl an *Chips* setzen (*Call*), schieben (*Check*) wenn keine *Chips* gesetzt wurden, mehr *Chips* setzen (*Raise*) oder aufgeben (*Fold*).
    - Alle gesetzten *Chips* landen im sogenannten *Pot*.
    - Wenn alle außer einem Spieler per *Fold* ausscheiden, gewinnt der verbleibende Spieler den *Pot*.
    - Ansonsten gewinnt der Spieler den *Pot*, der am Ende der *Wettrunde* (*Showdown*) die Hand mit dem höchsten *Pokerrang* besitzt. Besitzen mehrere Spieler gleichzeitig eine *Hand* mit dem höchsten *Pokerrang* wird der *Pot* zwischen diesen aufgeteilt (*Split*).
- *Bluff*en, also das Setzen von *Chips* trotz sehr geringem Pokerrang, ist ein wichtiges Element.
    - *Bluff*en wird möglich, weil ein Spieler durch Setzen von *Chips* alle anderen Spieler zur Aufgabe zwingen kann, ohne dass es zu einer Bewertung seines *Pokerrangs* kommt.
- Zu Beginn jeder Runde erhält ein Spieler den *Dealer-Button*.
    - Der Spieler mit dem *Button* kann in jeder *Wettrunde* als Letzter entscheiden, ob er *Chips* in den *Pot* setzt.
    - Vor Beginn der nächsten *Spielrunde* wandert der *Button* im Uhrzeigersinn zum nächsten Spieler.
- Ziel des Spiels ist es, die Anzahl der *Chips* zu maximieren.

Die einzelnen Poker-Varianten unterscheiden sich vor allem im konkreten Ablauf der *Wettrunden* und in der Relation von bekannten zu unbekannten Informationen. Während in den früheren Jahren mit Poker-Varianten wie „Five-card draw“[2] kaum Informationen über die Handstärke der Gegner bekannt waren, sind seit Anfang des Jahrtausends neue Varianten wie „*Texas Hold'em*“ populär geworden, bei denen die Spieler den *Pokerrang* des Gegners viel besser eingrenzen können und damit mehr Informationen über ihre Gegner besitzen. In den älteren Poker-Varianten bestand die Kunst mehr darin, die Stärke der eigenen Hand richtig einzuschätzen. Bei den neueren Poker-Varianten hingegen gewinnt der Spieler, der die Stärke der gegnerischen Hand richtig einzugrenzen vermag [16].

***Texas Hold'em* (HE)** *Texas Hold'em* ist mittlerweile die populärste Poker-Variante. Eine *Spielrunde* besteht dabei aus den folgenden vier, aufeinander folgenden, *Wettrunden*:

1. *Preflop*: Jeder Spieler erhält zwei Karten, die sog. *Handkarten*. Die *Handkarten* sind verdeckt und können nur von dem Spieler eingesehen werden, zu dem sie gehören. Bevor die eigentliche *Wettrunde* beginnt, müssen zwei Spieler (*Small Blind* und *Big Blind*) einen erzwungen Einsatz tätigen. Der *Small Blind* wird vom Spieler mit dem *Button* gestellt.
2. *Flop*: Es werden drei *Gemeinschaftskarten* öffentlich aufgedeckt.
3. *Turn*: Es wird eine vierte *Gemeinschaftskarte* öffentlich aufgedeckt.
4. *River*: Es wird eine fünfte *Gemeinschaftskarte* öffentlich aufgedeckt.

[2] Bei dieser Variante hält jeder Spieler fünf *Handkarten*, es gibt keine öffentlich sichtbaren *Gemeinschaftskarten*

Gemeinschaftskarten können von allen Spielern gleichzeitig für ihre *Hand* verwendet werden. Findet am Ende der vierten *Wettrunde* ein *Showdown* statt, dann bildet jeder Spieler aus insgesamt sieben Karten, also seinen zwei *Handkarten* und den fünf *Gemeinschaftskarten* eine *Hand* [17]. Die bereits aufgedeckten *Gemeinschaftskarten* werden auch als das *Board* bezeichnet.

***No Limit* (NL)** Der Zusatz *No Limit* zeigt an, dass ein Spieler, der innerhalb einer *Wettrunde* an der Reihe ist, eine beliebige Anzahl seiner *Chips* setzen darf. Er darf also auch alle restlichen *Chips* (sog. *All-In*) einsetzen. Ein *All-In* ist ein häufig eingesetztes Element innerhalb eines *Bluffs*, denn der Gegner muss sich gut überlegen ob er das Risiko eingeht, eine große Menge an *Chips* zu verlieren.

Im Gegensatz dazu ist bei der *Limit* Variante nur eine festgelegte Einsatzhöhe (ein bis zwei *Big Blind*s) erlaubt, deren Höhe in Abhängigkeit von der aktuellen *Wettrunde* variieren kann. Der Unterschied von *No Limit* zu *Limit* spiegelt sich deutlich in der Komplexität nieder. Bei ansonsten gleichen Regeln hätte die in dieser Arbeit verwendete Pokervariante mit einer Limit-Regel nur eine Komplexität von ungefähr 10^{18} [18] anstatt 10^{71} bei *No Limit* (siehe auch Abschnitt 4.4).

***Heads-up* (HU)** bezeichnet ein Poker-Spiel mit genau zwei Spielern.

Position* und *Button Bei *Heads-up* gibt es zwei verschiedene *Positionen*. Der Spieler, der nach dem *Flop* als Erster an der Reihe ist, besitzt „keine *Position*“ (*OUT*). Der andere Spieler hat den *Button* und ist deshalb „*in Position*“ (*IN*). Dadurch hat der *IN*-Spieler einen Informationsvorteil, der in aller Regel auch seinen erwarteten Nutzen dadurch verbessert, dass er seine Entscheidungen später trifft.

2.2.1 *No Limit Texas Hold'em* und Spieltheorie

Bei *No Limit Texas Hold'em* (NLHE) handelt es sich aus spieltheoretischer Sicht um ein indeterministisches Null-Summen Spiel mit imperfekten Informationen.

Zufallsknoten bestimmen zu Beginn die *Handkarten* der Spieler und später die *Gemeinschaftskarten*. In den Knoten eines Spielers entscheiden diese über den Einsatz von *Chips*. Der Gewinn des *Pot*s wird in den Endknoten festgelegt, entweder durch *Showdown* oder dadurch, dass nur noch ein Spieler verbleibt. Abbildung 2 zeigt den Ausschnitt des reduzierten Spielbaums (Abschnitt 2.1.1) für NLHE.

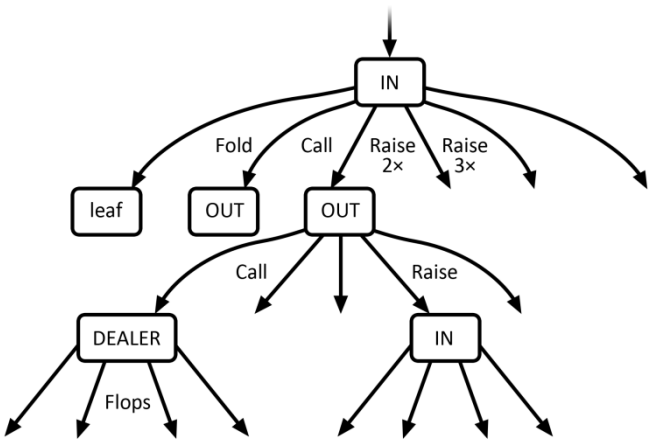


Abbildung 2: Ausschnitt eines reduzierten Baumes für NLHE mit Knoten beider Spieler sowie einem Dealer Knoten.

2.2.2 Grundlegende Ansätze für Pokersoftware

Da Poker im Internet regelmäßig um Geld gespielt wird, gibt es eine große Anzahl an Poker spielenden Programmen. Im Folgenden werden die wichtigsten Ansätze zur Berechnung einer spielstarken Strategie diskutiert.

Ansatz: Regel-, Formelbasiert

Der erste Ansatz für Pokersoftware war Regel basiert. Die Software entschied nach if-then-else und konnte aufgrund des Einsatzes, Pokerrangs und der Pot-Größe Entscheidungen fällen. Mit einer einfachen Menge an Regeln konnten simple, gemischte Strategien definiert werden.

Darauf aufbauend wurden formelbasierte Ansätze verwendet, die eine bessere Bewertung der aktuellen Situation zuließen. Mithilfe der Formeln für die relative Handstärke (siehe Abschnitt 4.4.1.4) oder auch des *Pot* Odds Konzepts (siehe 4.4.1.3) traf die Software Entscheidungen [18]. In den heutigen Poker-Programmen werden diese Formeln ausschließlich zur Berechnung einer Abstraktion des Spiels verwendet, durch die eine Komplexitätsreduktion erzielt wird.

Eines der ersten Poker-Programme, das auf diesen Ansätzen basierte, hieß „Poki“ und wurde 2002 von einem Team der Universität von Alberta entwickelt [19].

Ansatz: Big Data Im Online-Poker werden die Daten der gespielten Poker-Runden durch die Spieler und die Betreiber gespeichert. Dadurch sind Datenbanken mit mehr als einer Milliarde gespielter *Spielrunde*n *Cashgame* [20] entstanden. Mithilfe solcher Datenbanken ist es möglich, in Abhängigkeit der aktuellen Spielsituation (*Handkarten, Gemeinschaftskarten* und dem Einsatz des Gegners) Korrelationen in den historischen Daten zwischen den zu wählenden Aktionen und dem durchschnittlichen Gewinn zu bestimmen. Die Pokersoftware wählt dann diejenige Aktion aus für die, anhand der historischen Daten, der höchste, zu erwartende Gewinn ermittelt wurde. Um statt einer deterministischen eine gemischte Strategie zu erzeugen, wählt man die beste Aktion nur mit der höchsten Wahrscheinlichkeit aus und verteilt das restliche Wahrscheinlichkeitsgewicht auf die nächstbesten Aktionen.

Die Qualität dieser Strategie, die auf den Ergebnissen vergangener Poker-Runden basiert, hängt von zwei Faktoren ab. Einerseits ist die Anzahl der in der Datenbank gespeicherten *Spielrunden* entscheidend, um statistische Ausreißer zu minimieren. Außerdem hängt die Qualität der Strategie auch von der Qualität der Spieler ab, deren Spiele in der Datenbank enthalten sind. Handelt es sich um Daten von Anfängern und Fortgeschrittenen, wird die Strategie gegen Profis verlieren.

In der Evaluierung (siehe Abschnitt 7.1) zeigt sich aber, dass selbst in einer stark abstrahierten Form von NLHE HU allein von der 2. *Wettrunde* mehr als eine Billionen verschiedene Spielzustände existieren. Damit wird klar, dass selbst eine Datenbank mit einer Milliarden *Spielrunde*n nur einen winzigen Teil des gesamten Zustandsraumes abdecken kann.

Ansatz: Exploitierend mithilfe eines Gegner-Modells Ziel von allen exploitierenden, adaptiven Ansätzen ist die Anpassung der eigenen Strategie an die Strategie des Gegners. Insbesondere geht es darum, die Fehler des Gegners auszunutzen und dadurch den eigenen, erwarteten Nutzen zu erhöhen. Dazu wird ein Modell der Strategie des Gegners anhand der bisherigen Spielzüge

erzeugt. Mithilfe dieses Modells lassen sich dann die gegnerischen Wahrscheinlichkeiten für die Auswahl seiner Aktionen im Spielbaum schätzen. Gegen die auf diese Weise geschätzte Strategie des Gegners, berechnet die Software eine optimale Gegenstrategie (siehe Abschnitt 4.1). Je genauer die gegnerische Strategie geschätzt wird, umso höher der eigene, erwartete Nutzen.

Ein bekanntes, exploitierendes Pokerprogramm heißt „Brennus“ [21]. Es verfügt über ein Gegner-Modell, dass in jedem Spielzustand die Wahrscheinlichkeit aller *Handkarten* des Gegners (sog. *Range*) auf Grundlage der vergangenen Spielzüge des Gegners schätzt und mithilfe von Tiefensuche die Aktion mit dem höchsten Nutzen auswählt.

Grundsätzlich basieren adaptive Ansätze, wie Big Data Ansätze, auf historischen Daten. Allerdings sind für adaptive Ansätze nur die historischen Daten des derzeitigen Gegners interessant. Der Nachteil ist, dass ein rein adaptiver Ansatz am Anfang ohne Daten sehr schlecht spielt und erst mit steigender Anzahl gespielter Runden ein besseres Modell des Gegners aufbaut. Ist dieses Modell erst einmal robust genug, wird der adaptive Ansatz gute Ergebnisse erzeugen, weil er jede Schwäche der Strategie des Gegners ausnutzt. Eine Schwäche adaptiver Ansätze zeigt sich, wenn der Gegner seine Strategie variiert und damit das gelernte Modell nutzlos macht.

Der erwartete Nutzen einer optimalen, adaptiven Strategie *A* im Spiel gegen *B* konvergiert gegen das ε der optimalen Gegenstrategie G_B, also der Distanz der gegnerischen Strategie zum Nash-Equilibrium. Denn ε entspricht immer genau dem erwarteten Nutzen einer optimalen Gegenstrategie.

Ansatz: Spieltheorie und Berechnung des Nash-Equilibriums Im Gegensatz zu adaptiven Ansätzen wird für die Berechnung des Nash-Equilibriums die Strategie des aktuellen Gegners nicht berücksichtigt. Es wird der „Worst Case“ angenommen und davon ausgegangen, dass die eigene Strategie in allen Details bekannt ist und ein Gegner dieses Wissen nutzt, um die optimale Gegenstrategie zu wählen. Für diesen Fall maximiert das Nash-Equilibrium den zu erwartenden Nutzen.

Ein Nachteil des Nash-Equilibriums ist die hohe Komplexität der Berechnung. In der Implementierung und Evaluierung zeigt sich, wie umfangreich sich das Problem der Berechnung eines ε-Nash-Equilibriums mit kleinem ε in relevanter Zeit tatsächlich darstellt. Außerdem ist das Nash-Equilibrium nur dann die optimale Strategie, wenn der Gegner auch die optimale Gegenstrategie wählt. Die Schwächen eines Gegners vermag ein Nash-Equilibrium nicht gezielt zu exploitieren und erzielt gegen schwache Gegner deshalb einen geringeren Gewinn, als ein adaptiver Ansatz. Es zeigt sich aber, dass es gegen starke Gegner in aller Regel die beste Strategie ist (siehe Abschnitt 3.1).

In Abschnitt 4.3.1 sowie Kapitel 7 wird sich herausstellen, dass der in dieser Arbeit gewählte, iterative Ansatz gegen das Nash-Equilibrium konvergiert.

3 Verwandte Arbeiten

Mit der Entstehung von Pokersoftware entstand auch die Suche nach allgemeinen Bewertungskriterien, nach der Vergleichbarkeit der Spielstärke unterschiedlicher Software. Wie im letzten Kapitel dargestellt, gibt es statische und adaptive Ansätze. Bei der Suche nach einem allgemeinen Bewertungskriterium gibt es für diese beiden Ansätze zwei grundsätzlich unterschiedliche Herangehensweisen.

Statische Strategien werden einmal erstellt und anschließend nicht weiter an den Gegner angepasst. Daher eignet sich als Bewertungsmethode der Abstand ε zum Nash-Equilibrium. Es ist ein Maß für die Robustheit der Strategie, das eine Aussage über den im „Worst Case" zu erwartenden Verlust macht.

Für adaptive Strategien, die mithilfe eines Gegnermodells versuchen die Strategie des Gegners mit steigender Zahl an Spielen immer besser zu exploitieren, ist die Bewertung mithilfe des „Worst Case" ungeeignet. Anstatt dessen testet man die adaptive Strategie in einer großen Zahl an Spielen gegen mehrere, konkrete Strategien *B* und vergleicht den erwarteten Nutzen der adaptiven Strategie mit der optimalen Gegenstrategie G_B. Je geringer der Abstand ist, umso besser die adaptive Strategie. Außerdem ist die Lerngeschwindigkeit, also die Konvergenzgeschwindigkeit der adaptiven Strategie gegen G_B, ein weiteres Bewertungskriterium.

Die beiden dargestellten Qualitätsmaße sind aber nur ermittelbar, wenn die Strategien der Kombattanten vollständig bekannt sind. Dies ist in der Realität häufig nicht der Fall. Stattdessen lässt man die Pokersoftware in einer Simulation viele Male gegeneinander antreten. Dabei zeigt sich allerdings schnell, dass Strategien bei Poker häufig intransitiv sind [22] und es nicht immer möglich ist eine allgemein gültige Rangfolge von mehreren Poker Bots durch *Heads up* Spiele herzustellen. Es kann vorkommen, dass A gegen B gewinnt, B gegen C und C wiederrum gegen A.

Die erste Plattform, auf der Poker-Programme regelmäßig gegeneinander antraten, war der Internet-Relay-Chat, kurz IRC. Auf dem IRC Poker Server konnten Programme gegeneinander mit Spielgeld antreten. Die Programme starteten auf dem ersten Level und konnten sich für die höheren Level qualifizieren, sofern sie auf dem ihrem aktuellen Level im Schnitt mehr Spielgeld gewannen als verloren. [18]

3.1 Weltmeisterschaft der Pokersoftware

Seit 2006 wird die Annual Computer-Poker Competition[3] (ACPC) jedes Jahr entweder auf den Konferenzen AAAI [4] oder IJAI[5] ausgetragen. Es Handelt sich dabei um einen offenen Wettbewerb von Pokersoftware in der Poker-Variante *Texas Hold'em*. Im Jahr 2014 traten die Bots in folgenden fünf verschiedenen Disziplinen gegeneinander an:

- 2x Limit *Hold'em*: *Heads up* und 3-Spieler
- 2x *No Limit Hold'em Heads up*: „Bankroll" und „Run-off"
- 1x Kuhn Poker[6]: 3-Spieler

[3] http://www.computerpokercompetition.org/
[4] Association for the Advancement of Artificial Intelligence
[5] International Journal of Artificial Intelligence
[6] Kuhn Poker ist eine Variante mit stark reduzierter Komplexität, bei der es nur verschiedene drei Karten gibt

Innerhalb eines Matches einer Disziplin treten zwei bzw. drei Poker Programme gegeneinander an. Ein Match besteht aus zweimal 3.000 gespielten Händen. Nach den ersten 3.000 Händen wird der Speicher der Poker Software gelöscht. In den nächsten 3.000 Händen werden exakt dieselben Karten verteilt wie bei den ersten 3.000 Händen, nur mit dem Unterschied, dass die Positionen der Programme getauscht werden. Jedes Pokerprogramm spielt jede Situation also aus allen Positionen. Damit werden einerseits die unterschiedlichen Ausgangslagen durch die Positionen (vgl. *Button* in Abschnitt 2.2) und gleichzeitig die Varianz durch die unterschiedlichen *Handkarten* jedes Spielers ausgeglichen. Mit dieser Technik des „Duplicate Poker" gelingt es, nach wenigen tausend Händen ein aussagekräftiges Ergebnis zu erhalten.

Bei der Disziplin „Bankroll" tritt jedes Poker-Programm einmal gegen jedes andere Poker-Programm an. Die Anzahl der dabei gewonnenen *Chips* wird für jedes Programm aufsummiert. Anschließend werden die Programme nach der Reihenfolge der gewonnen Chips bewertet. In dieser Disziplin haben adaptive Programme Vorteile, die Gegner mit einer schwachen Strategie gezielt exploitieren können und deshalb mehr *Chips* gewinnen, als zum Beispiel eine statische Strategie nahe dem Nash-Equilibrium. Es stellt sich aber auch die Frage, ob die Anzahl von 3.000 Spielen groß genug ist, um ein robustes Gegner-Modell zu lernen. Diese Bedenken gegen adaptive Ansätze spiegeln sich auch in den Ergebnissen dieses Wettbewerbs wieder, bei dem in den letzten Jahren Programme mit statischen Strategien den Wettbewerb dominiert haben.

Die Disziplin „Run-off" besteht aus einer Art Eliminationsturnier mit mehreren Turnierrunden. Die Anzahl der Turnierrunden entspricht der Anzahl der Programme minus eins. In jeder Turnierrunde scheidet das Programm aus, das die wenigsten *Chips* gewonnen hat. Die Rangfolge am Ende entspricht der umgekehrten Reihenfolge der ausgeschiedenen Pokerprogramme. Im nächsten Abschnitt zeigt sich, dass auch in dieser Disziplin Pokerprogramme mit einem Ansatz zur Annäherung ans Nash-Equilibrium jeweils immer die obersten Plätze erreichen.

Sieger der ACPC

Auf der ersten Austragung der ACPC im Jahr 2006 siegte die Forschungsgruppe der Universität von Alberta aus Canada mit Hyperborean[7], einem Poker-Programm der Computer-Poker Research Group[8] (CPRG). Vorbild der CPRG war IBMs Deep Blue, eine Software die bereits im Jahr 1997 den damaligen Schachweltmeister Gary Kasparov unter Turnierbedingungen schlug. Die CPRG wurde mit dem Ziel gegründet, gleiches für Poker zu erreichen. Die seitdem weiterentwickelte Software Hyperborean entschied die Wettkämpfe der ACPC von Beginn an zu ihren Gunsten. Eine Ausnahme stellt aber das aktuelle Jahr 2014 dar, bei dem die Hyperborean nur den dritten Platz belegte.

In Tabelle 1 ist zu sehen, dass die Pokersoftware Hyperborean [12] [23], Tartanian [24], Neo Poker Bot [25], Slumbot NL [26], Nyx [27] und Prelude [25] jeweils die besten Plätze innerhalb der letzten vier Jahre belegen. Alle diese Programme basieren auf der Berechnung eines ε-Nash-Equilibriums. Damit dominiert diese Methode die zentrale Wettbewerbsveranstaltung für Pokersoftware.

[7] Hyperborea (griechisch „jenseits des Nördlichen"; Boreas war der Gott des Nordwinds) ist ein Sagen umworbenes, von den antiken griechischen Geographen weit im Norden lokalisiertes, Land
[8] http://poker.cs.ualberta.ca/

Tabelle 1: Bestplatzierte bei der ACPC im "Run-off" Wettbewerb seit 2011.

Sieger *Heads up No Limit Texas Hold'em*, Run-off [25]	
2011	1. Hyperborean-2011-2p-nolimit-iro (University of Alberta, Canada) 2. SartreNL[9] (University of Auckland, New Zealand) 3. Hugh (USA & Canada)
2012	1. Hyperborean (University of Alberta, Canada) 2. Tartanian5 (Carnegie Mellon University, USA) 3. Neo Poker Bot[10] (Alexander Lee, Spain)
2013	1. Hyperborean (University of Alberta, Canada) 2. Slumbot NL (Eric Jackson, USA) 3. Tartanian6 (Carnegie Mellon University, USA) 3. Nyx (Charles University, Czech Republic)
2014	1. Tartanian7 (Carnegie Mellon University, USA) 2. Prelude[11] (Tim Reiff, USA) 3. Hyperborean (University of Alberta, Canada)

Ein Auszug der genauen Ergebnisse einzelner Begegnungen des Run-off Wettbewerbs von August 2014 sind in Abbildung 3 dargestellt. Gut zu erkennen ist die klare Überlegenheit von Tartanian7, das gegen alle anderen Programme einen signifikant höheren, erwarteten Nutzen erreicht.

Dabei ist zu berücksichtigen, dass die Software Tartanian7 der Carnegie Mellon University mit Blacklight, dem Pittsburgh Supercomputer, berechnet wurde. Blacklight verfügt über 4096 Prozessoren, einem Arbeitsspeicher von insgesamt 32 TeraByte und konnte so eine wesentlich komplexere Abstraktion von NLHE HU als die Konkurrenten verwenden [25].

Außerdem zeigt sich in den Ergebnissen die intransitive Natur von Poker. Die viertplatzierte Software Slumbot erreicht einen positiven Nutzen gegen die zweitplatzierte Software Prelude.

	SartreNLExp	Nyx	Hyperborean_iro	Tartanian7	Slumbot	Prelude
SartreNLExp		-145.94 ± 58.93	-109.86 ± 40.91	-260.96 ± 46.71	-199.00 ± 31.22	-115.80 ± 44.68
Nyx	145.94 ± 58.93		-53.83 ± 43.80	-120.96 ± 38.07	-63.51 ± 42.64	-58.88 ± 52.39
Hyperborean_iro	109.86 ± 40.91	53.83 ± 43.80		-20.76 ± 16.22	16.24 ± 14.30	-9.71 ± 15.43
Tartanian7	260.96 ± 46.71	120.96 ± 38.07	20.76 ± 16.22		32.73 ± 15.63	19.76 ± 15.78
Slumbot	199.00 ± 31.22	63.51 ± 42.64	-16.24 ± 14.30	-32.73 ± 15.63		6.32 ± 14.70
Prelude	115.80 ± 44.68	58.88 ± 52.39	9.71 ± 15.43	-19.76 ± 15.78	-6.32 ± 14.70	

Abbildung 3: Ausschnitt (6 von 15 angetretenen Programmen) der "Run-off" Ergebnisse von der ACPC 2014.

Insgesamt lässt sich feststellen, dass der öffentliche Wettbewerb ACPC die momentan geeignetste Methode zur Evaluierung von Pokersoftware darstellt. Die Autoren der wichtigsten, wissenschaftlichen Arbeiten im Bereich Computer Software haben auch jeweils ein eigenes Programm für den Wettbewerb gestellt, um die neuesten Konzepte zu evaluieren.

[9] Gegen Sartre kann man online antreten: https://www.cs.auckland.ac.nz/poker/
[10] Auf Basis von Neo Poker Bot wurde eine Poker-Lernsoftware erstellt: http://www.neopokerbot.com/
[11] Auch auf Basis von Prelude ist eine neue Poker-Lernsoftware verfügbar: http://www.unfoldpoker.com/

4 Analyse: Berechnung des Nash-Equilibriums

In diesem Kapitel werden die theoretischen Grundlagen dargestellt, analysiert und diskutiert, die für eine effektive Berechnung eines Nash-Equilibriums benötigt werden. Die vorgestellten Konzepte werden von den erfolgreichsten Pokerprogrammen der ACPC und von der begleitend zu dieser Arbeit erstellten Software genutzt.

Wie in Abschnitt 2.1.3 dargestellt, ist ein Nash-Equilibrium ein Strategieprofil mit besonderer Qualität. Die besondere Qualität drückt sich darin aus, dass kein Spieler durch Abweichung von seiner Strategie seinen erwarteten Nutzen verbessern kann.

Da Poker grundsätzlich eine intransitive Natur (siehe Abschnitt 3.1) besitzt, könnte es möglich sein, dass kein solches Nash-Equilibrium existiert und man für jede konkrete Strategie eine spezialisierte, überlegene Gegenstrategie finden kann. Diese Überlegung ist auch richtig, sofern man nur reine Strategien betrachtet. Unter Verwendung von reinen Strategien existiert in der Regel kein Nash-Equilibrium, was sich anschaulich schon an einem einfachen Spiel wie Schere-Stein-Papier zeigt. Mit einer reinen Strategie müsste sich ein Spieler auf einen der drei Gegenstände festlegen und würde gegen die beste Gegenstrategie, die den dazu dominierenden Gegenstand wählt, immer verlieren.

Für gemischte Strategien sieht das Ergebnis anders aus. In seinem berühmten Beweis [7] hat John Nash schon 1951 gezeigt, dass für jedes endliche, strategische Spiel mindestens ein gemischtes Nash-Equilibrium existiert. Der Beweis basiert auf „Brouwers Fixpunktsatz“ [28] aus dem Jahr 1912, der Existenzaussagen über die Lösungen reeller, nicht linearer Gleichungssysteme erlaubt.

4.1 Berechnung der optimalen Gegenstrategie

Für ein gegebenes Strategieprofil (A, B) in einem zwei-Spieler-Spiel, kann mithilfe der Berechnung der optimalen Gegenstrategie (der besten Antwort) überprüft werden, ob ein Nash-Equilibrium vorliegt.

Die optimale Gegenstrategie G_A in Bezug auf eine Strategie A ist eine reine Strategie, die im Spiel gegen A den höchstmöglichen, erwarteten Nutzen erzielt. Wenn $\mu(A, B)$ der Nutzen ist, den Strategie A im Spiel gegen B erzielt, dann gilt unter den Bedingungen $\mu(G_A, A) \leq \mu(B, A)$, sowie $\mu(G_B, B) \leq \mu(A, B)$, dass das Strategieprofil (A, B) ein Nash-Equilibrium sein muss. Denn durch die Bedingungen ist sichergestellt, dass kein Spieler seinen erwarteten Nutzen verbessern kann, weil der erwartete Nutzen bereits mindestens so groß ist wie der erwartete Nutzen der jeweiligen besten Gegenstrategie.

4.1.1 Darstellung einer Strategie

Im Fall von NLHE HU (*No Limit Texas Hold’em* Heads-up) legt eine Strategie für alle *Handkarten* und alle Aktionen eine Wahrscheinlichkeitsverteilung fest, mit der die jeweilige Aktion unter den jeweiligen *Handkarten* ausgewählt wird. Da der Spieler nicht unterscheiden kann welche *Handkarten* sein Gegner besitzt, gelten für alle Knoten innerhalb derselben Informationsmenge des Spielers auch immer dieselben Wahrscheinlichkeiten für die verfügbaren Aktionen. Innerhalb einer Strategie gibt es also für jede Informationsmenge genau eine Wahrscheinlichkeitsverteilung über die verfügbaren Aktionen. Jede Informationsmenge eines Spielers ist ein Element aus seiner sichtbaren Vorgeschichte. Die sichtbare Vorgeschichte eines Spielers entspricht für NLHE HU den eigenen *Handkarten* und allen Aktionen, also den eigenen, denen des Gegners und dem Aufdecken von *Gemeinschaftskarten*.

Einen Auszug der Strategie eines Spielers wird für NLHE HU in Abbildung 4 dargestellt. Jede Kante in der Abbildung entspricht einer konkreten Aktion. Die Strategie wird innerhalb eines reduzierten Baumes (siehe Abschnitt 2.1.1) gezeigt und daher fasst ein Knoten alle Informationsmengen zusammen, die sich nur dadurch unterscheiden, dass der Spieler verschiedene *Handkarten* besitzt. In der Abbildung gibt es vier verschiedene Typen von Knoten:

- Einen *DEALER*-Knoten, der die per Zufall ausgewählte, letzte *Gemeinschaftskarte* aufdeckt.
- *IN*-Knoten des betrachteten Spielers *A*, der sich „in *Position*" befindet.
- *OUT*-Knoten des Gegenspielers *B*, der „keine *Position*" besitzt.
- Blätter, die aufgrund eines *Fold* oder eines *Showdowns* über die Verteilung des *Pots* entscheiden.

Der dargestellte Ausschnitt zeigt im *DEALER*-Knoten in der *Wettrunde River* die Auswahl der Aktion „*Gemeinschaftskarte* 9♦" aufdecken gewählt. Anschließend erhöht der *OUT*-Spieler um $2 und danach wählt der *IN*-Spieler eine Aktion (*Fold*, *Call* oder *Raise*) anhand seiner Strategie aus.

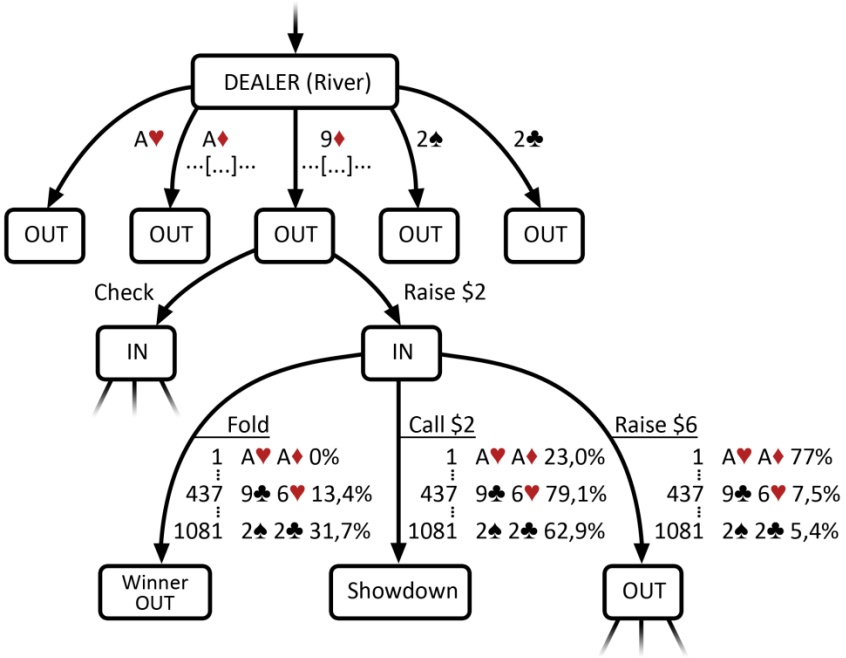


Abbildung 4: Darstellung des Ausschnitts einer Strategie in der *Wettrunde River* innerhalb eines reduzierten Baumes.

Die Strategie definiert für jede der möglichen $\binom{47}{2} = 1081$ *Handkarten* (fünf *Gemeinschaftskarten* wurden aufdeckt, es verbleiben 47 Karten) und für jede Aktion eine Wahrscheinlichkeit. Durch frühere Aktionen des Spielers ist es aber möglich, dass die Wahrscheinlichkeit für bestimmte *Handkarten* h_* bereits vor der Wahl einer Aktion im dargestellten *IN*-Knoten die Wahrscheinlichkeit $p(h_*) = 0$ annimmt. Das ist dann der Fall, sofern die Strategie des Spielers für eine zu diesem Knoten hinführende Aktion eine Wahrscheinlichkeit von 0% für h_* vorsieht. Eine Wahrscheinlichkeitsverteilung über die nächsten Aktionen ist demnach nur für *Handkarten* h nötig, die den aktuellen Knoten mit einer Wahrscheinlichkeit von $p(h) > 0$ erreichen.

In der Abbildung werden für die Aktionen des *IN*-Spielers alle 1081 Wahrscheinlichkeitsverteilungen in Abhängigkeit von den *Handkarten* angedeutet. Beispielsweise werden in der dargestellten Strategie für A♥A♦[12] die Aktionen *Fold* mit 0%, *Call* mit 23% und *Raise* mit 77% getroffen.

Die Abbildung 7 stellt nur ein erdachtes Beispiel dar. Ein Auszug aus einer echten Strategie, die mit der zu dieser Arbeit parallel entwickelten Pokersoftware das Nash-Equilibrium approximiert, befindet sich im Anhang am Ende dieser Arbeit.

4.1.2 Schritt 1: Berechnung der *Handkarten*-Wahrscheinlichkeiten in Blättern

Für die Berechnung der optimalen Gegenstrategie G_A ist es notwendig in allen Blättern des reduzierten Baumes die Wahrscheinlichkeitsverteilung über alle *Handkarten* (künftig als *Range* $r_k(h_i)$ bezeichnet) für die Strategie *A* zu kennen. Diese Wahrscheinlichkeiten werden benötigt, um den erwarteten Nutzen von verschiedenen Aktionen desselben Knotens zu vergleichen und in G_A die Aktion mit dem höchsten Nutzen auszuwählen.

Der Algorithmus 1 berechnet und speichert die *Ranges* von Strategie *A* in allen Blättern. Dazu startet er in der Wurzel und besucht mittels rekursiver Tiefensuche alle Knoten des reduzierten Baumes. Für alle *Handkarten* wird beim Durchlaufen des Baumes eine Wahrscheinlichkeit (p[h] im Algorithmus) mitgeführt, deren Wert beim Start in der Wurzel mit 1 initialisiert wurde. Beim Betreten einer Kante, die einer Aktion des Spielers A entspricht, werden die Wahrscheinlichkeiten für alle *Handkarten* mit den Wahrscheinlichkeiten der Strategie *A* für diese Aktion (entspricht s(v, h)) multipliziert.

Algorithmus 1: *Handkarten*Wahrsch(Knoten k, double p[], Strategie s)

```
Definition: k ist im ersten Aufruf der Methode die Wurzel
Definition: es handelt sich um einen reduzierten Baum, siehe Abbildung 4
Definition: p[h] wird mit 1 für alle Handkarten initialisiert
Definition: h sind Handkarten, also z.B. A♥9♦
Definition: s ist die Strategie A (gemäß Abbildung 4), für die die Gegenstrategie berechnet werden soll
Definition: v.k ist der auf die Kante v folgende Knoten k
Definition: s(v, h) ist die Wahrscheinlichkeit (double), dass die Kante v im bei Besitz von Handkarten h durch s gewählt wird
Definition: „valide" Handkarten sind Handkarten, die mit den Gemeinschaftskarten disjunkt sind

1: Wenn (k ist Knoten von Spieler A)
2:    Für alle v, die zu Kindern von k führen:
3:         pKopie = kopiere(p) //liefert eine Kopie von Array p
4:         Für alle validen Handkarten h
5:              pKopie[h] *= s(v, h)
6:         HandkartenWahrsch(v.k, pKopie, s)
7: Wenn (k ist Blatt)
8:    Speichere p als Range r_k des Spielers A in k
```

[12] Auch als „Pocket Rockets" bezeichnet, weil es sich um die stärksten *Handkarten Preflop* handelt.

Da der Algorithmus in jeder Kante v die Wahrscheinlichkeit aller *Handkarten* aktualisiert, beträgt die Laufzeit $O(|V| * |H|)$. Für NLHE ist die Anzahl $|H|$ der verschiedenen *Handkarten* $\binom{52}{2} = 1326$. Die Anzahl der Kanten $|V|$ beträgt ohne Abstraktion (Reduktion der Aktionen) $\geq 10^{70}$ [29] (siehe Abschnitt 4.4.1).

4.1.3 Schritt 2: Berechnung des erwarteten Nutzens von Aktionen

Angenommen Spieler B spielt gegen Strategie A und besitzt die *Handkarten* A♥9♥ in einem Knoten, wobei er zwischen drei verschiedenen Aktionen (z.B. *Check*, *Raise* oder *All-In*) wählen muss. B spielt dann die optimale Gegenstrategie, wenn er die Aktion mit dem höchsten, erwarteten Nutzen auswählt. Wie später in diesem Abschnitt gezeigt wird, kann der erwartete Nutzen mithilfe der *Range* an *Handkarten* von Spieler A getroffen und exakt berechnet werden. Dementsprechend ist G_A eindeutig definiert, wenn für jedes Element aus der sichtbaren Vorgeschichte von B der erwartete Nutzen der verfügbaren Aktionen bekannt ist und die Aktion mit dem höchsten, erwarteten Nutzen mit Wahrscheinlichkeit eins (reine Strategie) ausgewählt wird.

Die Berechnung des erwarteten Nutzens beginnt in den Blättern des reduzierten Baumes. Durch Verwendung der *Ranges* $r_k(h_i)$ von Spieler A, die für jedes Blatt berechnet werden können (siehe Abschnitt 4.1.2), wird im zweiten Schritt der erwartete Nutzen für alle möglichen *Handkarten* in den Blättern berechnet.

Die unten stehende Formel (1) berechnet für ein Blatt k_T den erwarteten Nutzen $\mu_c(h, k_T)$ für konkrete *Handkarten* h. Der Index c steht für „Counterfactual“ (zu Deutsch kontrafaktisch) und bedeutet, dass der erwartete Nutzen unter einer in aller Regel unzutreffenden Annahme bestimmt wird. Durch diese unzutreffende Annahme trifft der Spieler die zu diesem Knoten hinführenden Entscheidungen so, dass er mit den *Handkarten* h auf jeden Fall den Endknoten k_T erreicht. Dazu wählt er alle zu k_T hinführenden Aktionen mit Wahrscheinlichkeit 1 und alle anderen Aktionen mit Wahrscheinlichkeit 0 [12].

Dieser Ansatz ist sinnvoll, weil noch offen ist, welche Wahrscheinlichkeiten für die Aktionen in Strategie G_A den Nutzen maximieren. Da für jedes Blatt die vorhergehenden Aktionen mit Wahrscheinlichkeit von 1 angenommen werden, können verschiedene Pfade an Aktionen miteinander vergleichen werden. Dieser Vergleich ermöglicht es, nur die Aktionen mit dem höchsten erwarteten Nutzen mit Wahrscheinlichkeit 1 in G_A festzulegen.

In der Formel (1) werden die bedingte Wahrscheinlichkeit und die Gewinnfunktion benötigt:

- $z(karten) = \#karten$ (Funktion, welche die Anzahl der Karten zurückliefert)
- $p(a|b) = \begin{cases} \frac{1}{\binom{52-z(b)}{z(a)}} \\ 0, \ a \cap b \neq \emptyset \end{cases}$ (bedingte Wahrscheinlichkeit Karten *a* unter Auftreten Karten *b*)
- $w(h, b) = \{pokerrang \in \mathbb{Z} \mid pokerang\ Definition\ siehe\ Abschnitt\ 2.2\}$

- $w(h_1, h_2, b) = \begin{cases} \begin{cases} 1, w(h_1, b) > w(h_2, b) \\ -1, w(h_1, b) < w(h_2, b) \\ 0, w(h_1, b) = w(h_2, b) \end{cases} \\ g,\ z(b) < 5\ \{g \in [-1, 1], x \in \mathbb{R}\} \end{cases}$ (Gewinnfunktion vergleicht 2 *Handkarten*)

- Die Anzahl der *Chips* im *Pot* entsprechen c_k.
- Die *Gemeinschaftskarten* auf dem *Board* in einem Blatt k_T sind b_{k_T}.

Das Ergebnis der Gewinnfunktion $w(h_1, h_2, b)$ ist im Falle von noch ausstehenden *Gemeinschaftskarten* (z.B. im Falle eines vorzeitigen *Showdown*s auf dem *Flop*) eine rationale Zahl g im Bereich $[-1, 1]$, die dem Durchschnitt über den Vergleichen des *Pokerrang*s auf allen noch möglichen *Boards* entspricht.

Mit diesen Funktionen kann nun der erwartete Nutzen $\mu_c(h, k_T)$ für beliebige *Handkarten* h in allen Blättern k_T definiert werden:

$$\mu_c(h, k_T) = c_k * p(b_{k_T} | h) * \sum_{i=1}^{n} \left(r_{k_T}(h_i) * p\left(h_i \middle| (b_{k_T} + h)\right) * w(h, h_i, b_{k_T}) \right) \quad (1)$$

In μ_c wird der erwartete Nutzen von h gegen alle möglichen *Handkarten* des Gegners h_i unter Berücksichtigung *Range* $r_{k_T}(h_i)$, also der Wahrscheinlichkeit des Auftretens h_i im Blatt k_T aufsummiert. Diese Summe wird dann mit der Anzahl der *Chips* und der Wahrscheinlichkeit der *Gemeinschaftskarten* multipliziert. Wie man sieht gibt es allerdings keinen Term der die Wahrscheinlichkeit für das Auftreten von h berücksichtigt und deshalb ist der erwartete Nutzen counterfactual.

Im nächsten Schritt wird für h der erwartete Nutzen für jede Aktion berechnet. Dazu werden die Ergebnisse $\mu_c(h, k_T)$ der Blätter in Richtung Wurzel propagiert und so alle Knoten von Spieler A erreicht. Abbildung 5 zeigt am Beispiel der *Handkarten* A♥9♥ die Berechnung des erwarteten Nutzens für die Aktionen *Check* und *Raise* $2 des *OUT*-Spielers. Zuerst wird in den Knoten des *IN*-Spielers die Summe über die $\mu_c(h, k_T)$ in den Blättern gebildet. Diese Summe wird dann weiter nach oben gereicht. Es ist korrekt den erwarteten Nutzens aufzusummieren, weil bereits in den Blättern die *Range* der gegnerischen Strategie berücksichtigt wurde.

In einem *OUT*-Knoten, also einem Knoten des Spielers für den die Strategie bestimmt wird, wird der erwartete Nutzen aller Aktionen dieses Knotens miteinander verglichen. Der erwartete Nutzen einer Aktion (Kante) ist gleich dem erwarteten Nutzen des Knotens, zu dem diese Aktion hinführt. Im Beispiel beträgt der erwarteten Nutzen der Aktion „*Check*“ 0,7 und der Aktion „*Raise* $2“ 1,0. In der optimalen, reinen Strategie wird in jedem Knoten die Aktion mit dem maximalen erwarteten Nutzen gewählt. Diese beste Aktion wird also in der Strategie mit Wahrscheinlichkeit 1 gewählt und deshalb wird auch nur der erwartete Nutzen dieser besten Aktion Richtung Wurzel propagiert. Im Beispiel wird in der optimalen Gegenstrategie die Aktion „*Raise* $2“ mit Wahrscheinlichkeit 1 gespeichert und deren erwarteter Nutzen 1,0 nach oben im Baum weiter gereicht.

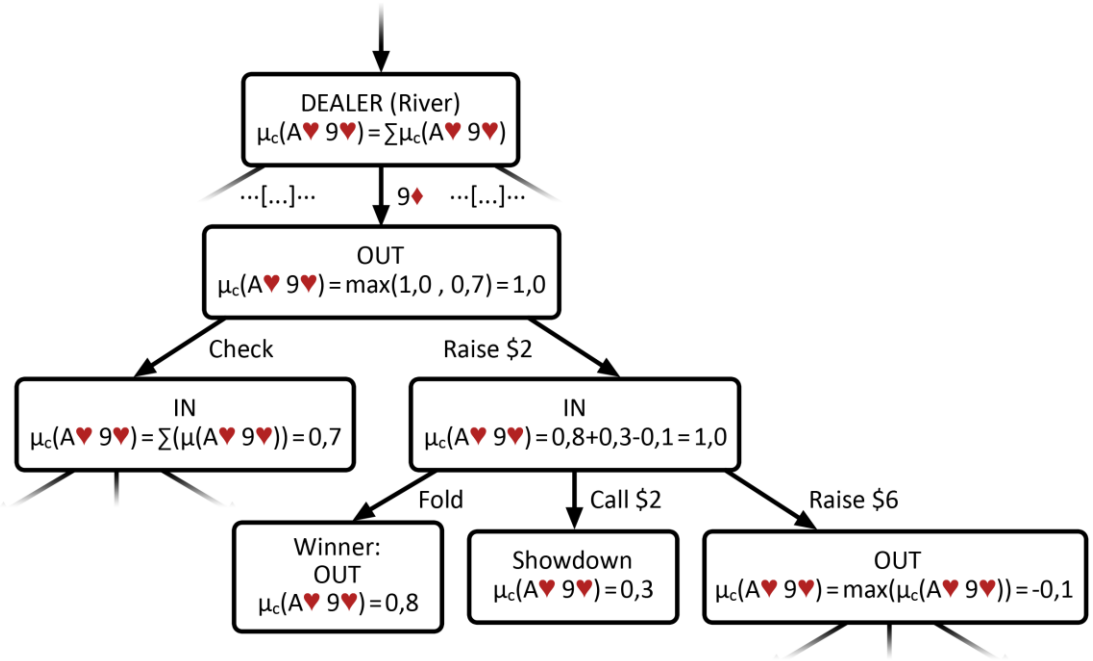


Abbildung 5: In Blättern berechneter erwarteter Nutzen wird hin zur Wurzel propagiert.

Wenn dieses Verfahren jetzt für alle möglichen $\binom{52}{2} = 1326$ *Handkarten* des *OUT*-Spielers wiederholt wird, entsteht die vollständige, optimale Strategie. Damit sind auch für alle *Handkarten* im Wurzelknoten k_W die Werte von $\mu_c(h, k_W)$ berechnet. Um den tatsächlichen, nicht-kontrafaktuellen, erwarteten Nutzen der Strategie $\mu(G_A, A)$ zu bestimmen, müssen nur noch die Wahrscheinlichkeiten der *Handkarten* berücksichtigt werden, die im kontrafaktuellen Nutzen nicht enthalten waren.

$$\mu(G_A, A) = \sum_{i=1}^{n} \mu_c(h_i, k_W) * p(h_i) \quad mit \;\; p(h_i) = \frac{1}{\binom{52}{z(h_i)}}$$

4.1.4 Bestimmung von ε

In der Einleitung in Abschnitt 4.1 wurde gezeigt:

$\mu(G_A, A) \leq \mu(B, A) \;\wedge\; \mu(G_B, B) \leq \mu(A, B) => (A, B)\; ist\; Nash - Equilibrium$

Da G_A und G_B per Definition die Strategien mit dem höchsten, erwarteten Nutzen sind, gilt außerdem

$\mu(G_A, A) \geq \mu(B, A) \;\wedge\; \mu(G_B, B) \geq \mu(A, B)$ und daraus folgt die Gleichheit des erwarteten Nutzens $\mu(G_A, A) = \mu(B, A) \;\wedge\; \mu(G_B, B) = \mu(A, B)$, sofern (A, B) ein Nash-Equilibrium ist.

Da NLHE HU ein Nullsummenspiel für 2-Spieler ist, muss außerdem gelten:

$\mu(B, A) + \mu(A, B) = 0$ und im Falle, dass (A, B) ein Nash-Equilibrium ist $\mu(G_A, B) + \mu(G_B, B) = 0$.

In Abschnitt 2.1.3 wurde ε als Abstand zum Nash-Equilibrium eingeführt. Mithilfe der optimalen Gegenstrategien G_A und G_B kann ε jetzt exakt für (A, B) bestimmt werden:

$\varepsilon = \mu(G_A, A) + \mu(G_B, B)$

Damit ist ε gleichzeitig ein Maß für die Exploitierbarkeit einer Strategie. Im Falle von $\varepsilon = 0$ liegt ein Nash-Equilibrium vor.

4.2 Berechnung des exakten Nash-Equilibriums

Für die Berechnung des Nash-Equilibriums ist die Bestimmung von ε eine zentrale Voraussetzung. Mithilfe von ε kann die Qualität jeder berechneten Strategien bestimmt werden. Bei iterativen Verfahren, die eine Strategie mit jeder Iteration näher an das Nash-Equilibrium bringen, kann zudem die Konvergenzgeschwindigkeit bestimmt werden, mit der ε gegen Null strebt.

Berechenbarkeit und Komplexitätsklasse

Obwohl der Beweis der Existenz des Nash-Equilibriums von 1951 stammt, gelangen erst ab dem Jahr 2006 die ersten, vielversprechenden Versuche sich mithilfe von Software dem Nash-Equilibrium für NLHE HU zu nähern [12].

Auch die Einordnung in die Komplexitätsklasse für die allgemeine Berechnung des Nash-Equilibriums wurde erst im Jahr 2006 bewiesen. In [30] wird gezeigt, dass die Berechnung des Nash-Equilibriums in PPAD (Polynomial Parity Argument, Directed version) liegt, einer Komplexitätsklasse die 1991 entdeckt wurde [31]. FNP (F für funktional) ist die um funktionale Probleme erweiterte Klasse von NP. TFNP (T für total) liegt in FNP und enthält nur Probleme, die garantiert existieren. PPAD wiederrum liegt in TFNP und enthält alle Probleme, deren Beweise auf gerichteten Graphen basieren.

Abbildung 6 zeigt, wo sich die Klasse PPAD im Komplexitätsuniversum befindet. Wie man sieht liegt PPAD zwischen P und NP. Es wurde sogar in [32] gezeigt, dass das Problem der Berechnung eines Nash-Equilibriums für ≥ 2 Spieler PPAD-vollständig ist. Damit ist es sogar das schwierigste Problem innerhalb dieser Komplexitätsklasse. Solange nicht die vermutete Annahme $P \neq NP$ bewiesen wurde, kann es natürlich sein, das alle anderen Komplexitätsklassen in der Klasse P zusammenfallen.

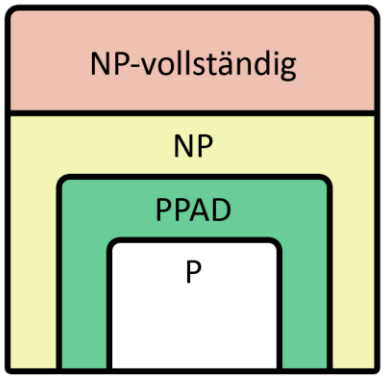


Abbildung 6: Einordnung der Komplexitätsklasse PPAD, die auch die Berechnung des Nash-Equilibriums enthält.

Im speziellen Fall von zwei-Spieler Null-Summen Spielen lässt sich aber zeigen, dass das Nash-Equilibrium mittels linearen Programmierens gefunden werden kann. Die Ellipsoidmethode, das

Innere-Punkte-Verfahren oder der noch schnellere Simplex Algorithmus finden das Nash-Equilibirum in polynomieller Zeit [30].

Lineares Programmieren

Obwohl das exakte Nash-Equilibrium mit linearem Programmieren für NLHE HU in polynomieller Zeit gefunden werden kann, ist es in der Praxis aufgrund der Komplexität des Spielbaums nicht machbar. Selbst für Rhode Island Poker, eine extrem reduzierte Version von Limit *Hold'em* Poker mit nur 10^9 (vergleiche NLHE HU mit 10^{71} Knoten) im Spielbaum [33], führt lineares Programmieren zu einer Matrix mit jeweils 91.224.226 Zeilen und Spalten, welche zu groß ist, um sie mit heutigen Rechnern zu lösen [34].

4.3 Annäherung mit Counterfactual Regret Minimization (CFR)

In den letzten Jahren hat sich bei zur Berechnung des Nash-Equilibrium für NLHE HU mit dem 2007 entdeckten Counterfactual Regret Minimalization (CFR) Algorithmus [12] ein iteratives Verfahren durchgesetzt, bei dem ε gegen Null konvergiert.

Das Verfahren basiert auf der Überlegung, das Bedauern (Regret) beim Treffen von Entscheidungen zu minimieren. Dahinter steckt die Idee der Opportunitätskosten, bekannt aus der BWL. Bei den Opportunitätskosten berücksichtigt man nicht nur den Nutzen, der beim Treffen einer bestimmten Entscheidung entsteht, sondern betrachtet auch den Nutzen von alternativen Entscheidungen, die man nicht getroffen hat. Opportunitätskosten entstehen dadurch, dass vorhandene Optionen zur Nutzung von Ressourcen nicht wahrgenommen werden. Sie entsprechen dem Abstand zwischen dem Nutzen der tatsächlich gewählten Alternative und dem Nutzen einer anderen, nicht getroffenen, Entscheidung. Regret Minimization ist eine Entscheidungsregel, die diejenige Option mit den geringsten erwarteten Opportunitätskosten auswählt. Bei Spielen mit mehreren Teilnehmern wählt diese Entscheidungsregel nicht die Option mit dem maximalen Nutzen, sondern die Option mit den geringsten negativen Folgen im „Worst-Case".

Damit hat das Konzept große Parallelen zum Nash-Equilibrium. Beide basieren auf der Grundidee Entscheidungen bzw. eine Strategie zu wählen, die für den ungünstigsten Fall an Ereignissen bzw. Entscheidungen des Gegners den höchsten erwarteten Nutzen erreicht.

Das Bedauern wird mithilfe des erwarteten Nutzens einer Strategie berechnet. Im Falle einer gemischten Strategie A, lässt sich in jedem Knoten k und seinen Kindern, den Knoten k_i, für die *Handkarten* h der erwartete Nutzen berechnen. Die Strategie ist in $s_A(h, k_i) \in [0,1]$ gespeichert und gibt für alle *Handkarten* h und jeden Kind-Knoten von k_i die Wahrscheinlichkeit an, dass die Aktion, die von k zu k_i führt, ausgewählt wird.

$$\mu_c(h,k) = \sum_{i=1}^{n} \mu_c(h,k_i) * s_A(h,k_i)$$

Der erwartete Nutzen des Knotens wird hierbei über die Summierung des Nutzens seiner Kinder berechnet. Diese Berechnung fängt in den Blättern unter Verwendung der in Abschnitt 4.1.3 eingeführten Formel (1) an.

Dementsprechend ist das zu erwartende Bedauern für einen Knoten k_i in der Iteration t:

$$R_t(h, k_i) = \mu_c(h, k_i) - \mu_c(h, k)$$

Wie man sieht ist R_t genau dann positiv, wenn der erwartete Nutzen durch die reine Wahl der zu k_i hinführenden Aktion größer ist als der Nutzen $\mu_c(h, k)$ bei Verwendung der Verteilung der Wahrscheinlichkeiten aufgrund der gemischten Strategie A. Ein positives R_t drückt damit das Bedauern aus, die zu k_i führende Aktion nicht mit einer noch größeren Wahrscheinlichkeit als in s_A festgelegt, gewählt zu haben. Im CFR Algorithmus wird R_t in jeder Iteration t berechnet und dazu genutzt, die Strategie A zu verbessern. Dazu wird das Bedauern über alle Iterationen in R^T akkumuliert:

$$R^T(h, k_i) = \sum_{t=1}^{T} R_t(h, k_i)$$

Für die Gewichtung der neuen Wahrscheinlichkeiten in der Strategie wird zuerst die Summe aller positiven $R^T(h, k)$ berechnet:

$$D_k = \sum_{i=1}^{n} max\big(0, R^T(h, k_i)\big)$$

Anschließend werden die neuen Wahrscheinlichkeiten $s_{A_{neu}}$ für alle Aktionen für die Strategie A im Knoten k bestimmt:

$$s_{A_{neu}}(h, k_i) = \begin{cases} \dfrac{\max\big(0, R^T(h, k_i)\big)}{D_k} \\ \dfrac{1}{z(k)}, \quad falls\ D_k \leq 0 \end{cases}$$

Für eine zu k_i hinführenden Aktion wird die neue Wahrscheinlichkeit $s_{A_{neu}}(h, k_i)$ nur dann > 0 sein, wenn $R^T(h, k_i)$ positiv ist. Weil die R^T mit D_k normiert wird, entsprechen die $s_{A_{neu}}$ dem relativen Verhältnis aller positiven R^T.

Anschaulich wird dadurch die Strategie immer wieder so angepasst, dass Aktionen mit positivem Bedauern in der nächsten Iteration mit einer höheren Wahrscheinlichkeit gewählt werden. Dies geschieht in jeder Informationsmenge (Knoten x *Handkarten*) unabhängig. In Abschnitt 4.3.1 zeigt sich, dass durch diese isolierte Optimierung der Strategie Wahrscheinlichkeiten jeder einzelnen Informationsmenge das Bedauern insgesamt und damit auch ε sinkt. Durch Verbesserung in allen Teilen konvergiert also auch die gesamte Strategie gegen das Nash-Equilibrium.

Aufbauend auf dem Beispiel in Abbildung 5 ist in Abbildung 7 dargestellt, wie der CFR Algorithmus für einen Knoten des *OUT*-Spielers die Strategie für die *Handkarten* A♥9♥ in der 2. Iteration verbessert. Die alte Strategie s_A sah für die Aktionen *Check* und *Raise* $2 jeweils eine Wahrscheinlichkeit von 50% vor. Da der erwartete Nutzen von *Check* mit 0,7 aber geringer als der von *Raise* $2 mit 1,0 ausfällt, wird auf Basis des akkumulierten Bedauerns R^T die Strategie so angepasst, dass *Raise* $2 nach der 2. Iteration mit 68,75% und *Check* nur noch mit 31,25% gewählt wird. Das hat zur Folge, dass der erwartete Nutzen der neuen Strategie im betrachteten Knoten von 0,85 auf 0,91 ansteigt. Die in der Abbildung unter 5. dargestellte Berechnung des Nutzens $\mu_{c_{neu}}$ auf

Basis der Änderungen der Strategie ist nicht in der Originalversion des CFR enthalten und wird in der Evaluierung in Abschnitt 7.4 untersucht.

Der dargestellte Fall, dass das akkumulierte Bedauern R^{T-1} für beide Aktionen positiv (jeweils 0,4) ist, kommt durchaus vor. Zum Beispiel dann, wenn durch Änderung der Strategie B der erwartete Nutzen einer Aktion a_1 für Strategie A steigt und a_1 vorher nur mit Wahrscheinlichkeit 0 gewählt wurde und eine andere Aktion a_2 mit Wahrscheinlichkeit 1. In diesem Fall wird das Bedauern R_t von a_2 0 betragen, das Bedauern für a_1 positiv sein und die Summe des Bedauerns steigt um einen Wert größer Null.

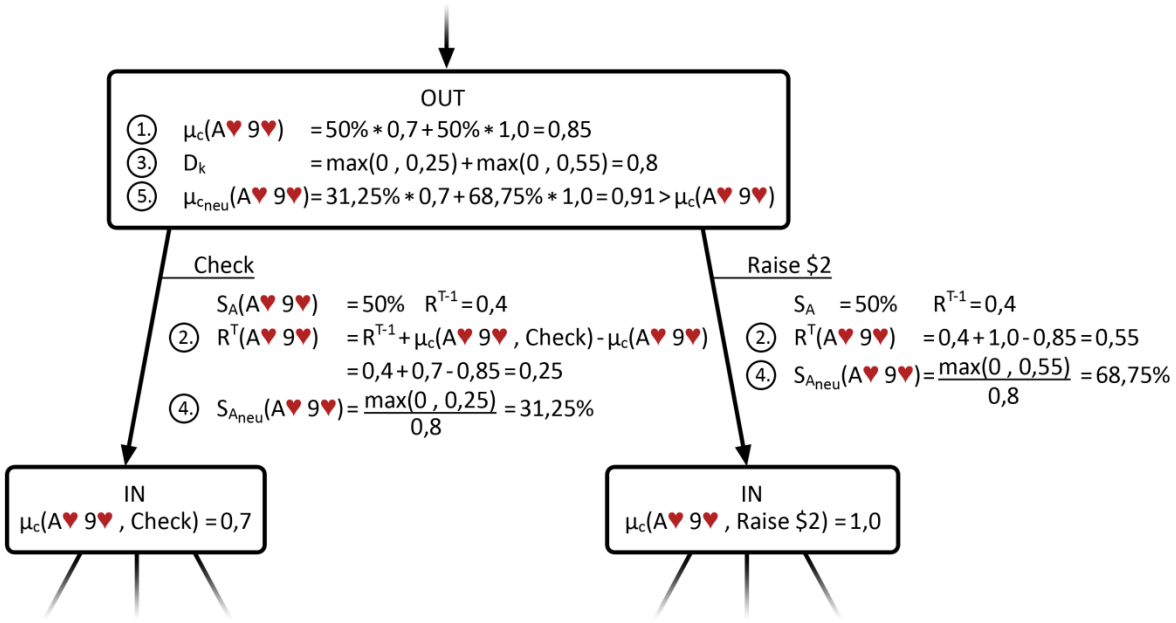


Abbildung 7: Beispiel der 2. Iteration des CFR, Aktualisierung der Strategie Wahrscheinlichkeiten zweier Aktionen.

Bis hierhin wurde gezeigt, wie eine Strategie für bestimmte *Handkarten* innerhalb eines Knotens optimiert wird. Der rekursive Algorithmus 2 zeigt, wie eine Iteration des CFR für eine Strategie abläuft.

Als Eingabe erhält der CFR die zu optimierende Strategie A und die aktuelle Strategie B des zweiten Spielers. Über beide Strategien liegen alle Informationen vollständig vor. Der CFR verbessert also die Strategie A anhand des durch B induzierten, erwarteten Nutzens. Zuerst werden wie in Abschnitt 4.1.2 dargestellt die *Handkarten* Wahrscheinlichkeiten in den Endknoten bestimmt. Damit kann dann der erwarteten Nutzen $\mu_c(h, k_T)$ in den Blättern k_T berechnet werden. Dieser wird anschließend (s.o.) zur Wurzel des Baumes propagiert. In den Knoten des Gegners oder des *Dealer*s wird der erwartete Nutzen summiert und in Richtung Wurzel weiter gegeben. In Knoten des Spielers A wird hingegen die Strategie, durch Minimierung des Bedauerns, verbessert.

Die Originalversion des CFR sieht vor, dass der erwartete Nutzen nach oben propagiert wird, der auf Grundlage der alten Strategie berechnet wurde. In Algorithmus 2 wird noch eine optimale Änderung eingeführt, bei der stattdessen der erwartete Nutzen durch die soeben aktualisierte Strategie nach oben propagiert wird. Die Konvergenzgeschwindigkeit dieser Variante des CFR wird mit der Originalversion in der Evaluierung in Abschnitt 7.4 verglichen.

Algorithmus 2: double cfrIteration(Strategie S_A, Strategie S_B, Knoten k, *Handkarten* h, Bedauern R^T)

```
Definition: Der Rückgabewert (double) ist der erwartete Nutzen in k
Definition: S_A und S_B sind Strategien
Definition: k ist im ersten Aufruf der Methode die Wurzel
Definition: v.k ist der auf die Kante v folgende Knoten
Definition: R^T(h,k) bildet Handkarten und Knoten auf die Summe des
Bedauerns aller vorhergehenden Iterationen ab (siehe Formel R^T)

1: Wenn(k ist Blatt)
2:    return μ_c(h, k) (siehe Abschnitt 4.1.3)
3: Wenn(k Dealer-Knoten oder Knoten von B)
4:    μ_Sum = 0
5:    Für alle v, die Kanten von k in Richtung Blatt sind:
6:          μ_Sum += cfrIteration(S_A, S_B, v_i.k, h, R^T)
7:    return μ_Sum
8: Wenn (k ist Knoten von A)
9:    Erzeuge neues Array μ_c[] mit Länge Anzahl Kanten von k + 1
10:   Für alle v_i die Kanten von k in Richtung Endknoten sind:
11:         μ_c[v_i.k] = cfrIteration(S_A, S_B, v_i.k, h, R^T)
12:         μ_c[k] += μ_c[v_i.k] * S_A(v_i, h)
13:   Aktualisiere R^T(h,k) mit μ_c[] (siehe Abschnitt 4.3)
14:   Aktualisiere Strategie (S_A_neu) mit R^T(h,k) (siehe Abschnitt 4.3)
15:   return μ_c[k]

Optional: Neuen statt alten Nutzen zurückgeben (statt Zeile 15)
15:   μ_c_neu[k] = 0
16:   Für alle v_i die Kanten von k in Richtung Endknoten sind:
17:         μ_c_neu[k] += μ_c[v_i.k] * S_A_neu(v_i, h)
18:   return μ_c_neu[k]
```

Algorithmus 2 optimiert im Hinblick auf die Konvergenz eines *Strategieprofil*s genau eine von zwei Strategien und entspricht deshalb einer halben Iteration.

Darauf aufbauend zeigt Algorithmus 3, wie *T* Iterationen des CFR ein ganzes Strategieprofil optimieren. Dazu werden zwei Strategien mit einer einfachen Regel (z.B. immer *Call*) initialisiert und anschließend abwechselnd durch Algorithmus 2 optimiert. Anschaulich durchlaufen beide Strategien eine Evolution, bei der sie sich in jedem Schritt dadurch verbessern, dass sie durch Ausnutzen der Schwächen der jeweils anderen Strategie voneinander lernen. Das zurückgegebene Strategieprofil entspricht den durchschnittlichen Strategien über alle Iterationen.

Der CFR sieht vor, dass nach jeder Iteration die aktuelle Strategie zur Summe über die Strategien aller vorherigen Iterationen S_Σ hinzu addiert wird. Die finale Strategie, die am Ende aller Iterationen als Ergebnis zurückgegeben wird, ergibt sich aus S_Σ normiert mit der Anzahl der Iterationen T.

Algorithmus 3 CFR(Strategie[] S, int I)

Definition: S enthält ein *Strategieprofil* S[0]=S_A (*IN*) und S[1]=S_B (*OUT*), das mit einer einfachen Regel (z.B. immer „*Call*") initialisiert wurde

Anforderung: T (int) ist die Anzahl der Iterationen

Anforderung: k_s (Knoten) ist der Wurzelknoten der Strategie s

Anforderung: R^T (Regret) ist für alle Strategien s, Knoten k und *Handkarten* h mit 0 initialisiert

Anforderung: p[] (Wahrscheinlichkeit) wird für alle *Handkarten* mit 1 initialisiert

Anforderung: In S_Σ werden die Strategien über alle Iterationen aufsummiert

```
1: Initialisiere S_Σ[] (Aufsummierung der Strategien)
2: Für i=0 bis T
3:    Für j=0 bis S.length
4:          strategie = S[j]
5:          HandkartenWahrsch(k_strategie, new p[],strategie)
6:          Für alle Handkarten h
7:                cfrIteration(S[j], S[1-j], k_si, h, R^T)
8:          S_Σ[j] += S[j]
9: Normiere S_Σ mit T (alle Wahrscheinlichkeiten durch T dividieren)
10: return S_Σ
```

Komplexität Algorithmus 2 wird von Algorithmus 3 in einer Iteration für jede der beiden Strategien und alle *Handkarten* aufgerufen. Algorithmus 2 benötigt in allen Blättern eine Operation für die *Handkarten* des Gegners (siehe Formel $\mu_c(h, k_T)$ in Abschnitt 4.1.3). Damit entspricht die Komplexität einer Iteration des CFR Algorithmus $O(|H|^2 * |K_T|)$, wobei $|K_T|$ die Anzahl der Blätter ist. Das entspricht gleichzeitig der Anzahl der Spielzustände $O(|H|^2 * |K|)$ da $|K_T| \geq \frac{|K|}{2}$ gilt, weil die meisten Knoten des Baumes Blätter sind.

Für den praktischen Einsatz des CFR ist neben der Komplexität vor allem die Konvergenzgeschwindigkeit entscheidend.

4.3.1 Konvergenz und Sampling

Wie dargestellt, reduziert der CFR Algorithmus durch Änderung der Strategie in jeder Iteration das Bedauern einer Informationsmenge. Das durch die Minimierung des Bedauerns in jeder Informationsmenge auch das Bedauern der gesamten Strategie begrenzt wird, wurde in [12] von

Zinkevich et al. (Theorem 3) bewiesen. Für das gesamte Bedauern gilt also $R_i^T \leq \sum_{h \epsilon H_i, k \epsilon K_i} R_t(h,k)$, wobei H_i die Menge aller *Handkarten* und K_i die Menge aller Knoten der Strategie i sind. Im gleichen Paper wird gezeigt, dass der CFR Algorithmus mit jeder Iteration das gesamte Bedauern reduziert.

Jetzt muss nur noch der Zusammenhang zum Nash-Equilibrium hergestellt werden. Für den Fall eines zwei-Spieler Null-Summen Spiels ist der Zusammenhang zwischen Bedauern und ε eines Nash-Equilibriums bekannt:

Ist das Bedauern jeder Strategie eines Strategieprofils kleiner als $\frac{\varepsilon}{2}$, dann ist das Strategieprofil ein ε-Nash-Equilibrium. Wäre diese Aussage falsch, dann müsste ein Paar aus Gegenstrategien existieren, deren Summe über dem erwarteten Nutzen größer ε ist. Das kann aber nicht der Fall sein, weil sonst die Summe des Bedauerns beider Strategien größer ε sein müsste, welches aber nie größer als ε wird. Da der CFR das Bedauern minimiert, wird ε mit jeder Iteration kleiner. In [12] wurde auch gezeigt, dass für die Rate, um die ε kleiner wird, eine obere Schranke existiert. Damit ist bewiesen, dass der CFR gegen das Nash-Equilibrium konvergiert.

4.3.1.1 Konvergenzgeschwindigkeit und Sampling

Der in Abschnitt 4.3 vorgestellte CFR Algorithmus besucht in jeder Iteration alle Knoten des Baumes und aktualisiert alle Wahrscheinlichkeiten in den Strategien. Diese vollständige Variante wird auch als „Vanilla“ CFR bezeichnet, weil es sich um die Reinform des Algorithmus handelt.

Seit der Entdeckung des CFR wurden verschiedene Sampling Techniken untersucht, mit dem Ziel die Laufzeit einer Iteration zu reduzieren. So konnte gezeigt werden, dass Monte Carlo CFR (MCCFR), bei dem in jeder Iteration immer nur ein Teilbaum betreten wird, pro Zeit im Vergleich zum Vanilla CFR schneller konvergiert [35]. Die Konvergenzgeschwindigkeit pro Iteration ist zwar geringer, was aber durch eine geringere Zeit pro Iteration überkompensiert wird. Folgende Varianten von Monte Carlo Sampling, die sich in der Größe des betretenen Teilbaumes unterscheiden, werden in aktueller Pokersoftware eingesetzt.

Outcome Sampling Outcome Sampling entspricht der extremsten Form von Sampling. Dabei wird in jeder Iteration immer nur ein einziger Pfad von der Wurzel zu einem bestimmten Blatt des Baumes betreten. Nur für die auf diesem Pfad liegenden Informationsmengen werden neue Wahrscheinlichkeiten in den Strategien berechnet.

External Sampling Hierbei werden die Aktionen des Gegners und die Zufallsentscheidungen in *Dealer*-Knoten gesampelt. Pro Zeiteinheit konnte für External Sampling in vielen einfachen Spielen (z.B. 1-Karte-Poker) eine bessere Konvergenzgeschwindigkeit im Vergleich zu Outcome Sampling gezeigt werden. [35]

Public-Chance Sampling Die nach aktuellen Ergebnissen beste Sampling Technik ist die des Public-Chance Sampling. Dabei werden nur die Aktionen in *Dealer*-Knoten, also die zufällige Auswahl von *Gemeinschaftskarten*, gesampelt. Das entspricht auch der Eigenschaft der imperfekten Informationen von Poker und deshalb verträgt sich diese Sampling Technik wesentlich besser mit der Verwendung von verlustbehafteten Abstraktionen wie Imperfect Recall (siehe Abschnitt 4.4.1). Außerdem lässt sich zeigen, dass diese Technik eine Beschleunigung des CFR Algorithmus in Blättern erreichen lässt, bei der sich dessen Komplexität von $O(n^2)$ auf $O(n)$ reduziert [36].

4.4 Komplexitätsreduktion

Der Grund für die Schwierigkeit der Berechnung eines Nash-Equilibriums für Poker liegt an der enormen Komplexität des Spiels. Der Ursprung der Komplexität lässt sich auf zwei Bereiche zurückführen, einmal den *Wettrunden*, die Aktionen von Spielern abbilden, und einmal den Karten. Die Karten lassen sich noch in die zwei Unterbereiche *Handkarten* und *Gemeinschaftskarten* (*Board*) aufgliedern.

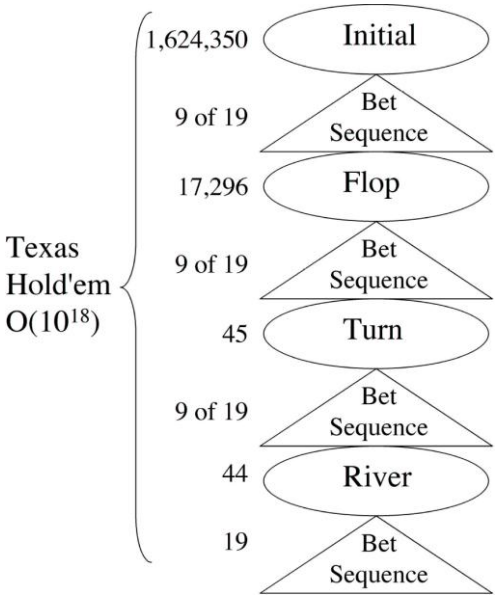


Abbildung 8: Komplexität von Limit *Hold'em* in Anzahl der Spielzustände.

In Abbildung 8 (aus [37]) ist die Komplexität von *Limit Hold'em* dargestellt. Am Anfang bekommt jeder Spieler *Handkarten*, daraus ergeben sich $\binom{52}{2} * \binom{50}{2} = 1.624.350$ verschiedene Kombinationen. Anschließend findet eine *Wettrunde* statt. Da bei *Limit Hold'em* (im Unterschied zu *No Limit Hold'em*) nur mit einer einzigen, durch die Regeln festgelegten, Höhe an *Chips* für ein *Raise* gespielt wird, ergeben sich nur neun verschiedene Sequenzen an Setzoptionen[13] (cC, cRc, cRrC, cRrRc, cRrRrc, rC, rRc, rRrC, rRrRc) die diese Runde nicht mit *Fold* beenden. Auf dem *Flop* erscheinen drei *Gemeinschaftskarten* und damit $\binom{48}{3}$ Kombinationen, eine weitere jeweils auf *Turn* $\binom{47}{1}$ und *River* $\binom{46}{1}$. Auf das Erscheinen neuer *Gemeinschaftskarten* folgt jedes Mal eine *Wettrunde*, wobei in der letzten *Wettrunde* zusätzlich Sequenzen berücksichtigt wurden, die auf *Fold* enden. Multipliziert man die Kombinationen der einzelnen Runden miteinander, ergibt sich eine Komplexität von $\approx 10^{18}$ Spielzuständen. Selbst für Limit *Hold'em* ist bis zum heutigen Tag noch keine exakte Berechnung des Nash-Equilibriums gelungen. Es existieren aber schon Approximationen mit sehr geringem ε [25].

Komplexität

Im Vergleich zu *Limit Hold'em* steigt die Anzahl der Aktionen, zwischen denen ein Spieler mit einer *Stack* Größe von 100 *Big Blinds* bei *No Limit Hold'em* in einem einzigen Knoten wählen kann, von zwei (z.B. *Check* oder *Raise*) auf ungefähr einhundert (*Check*, *Raise* 1, *Raise* 2, …, *Raise* 98, *All-In*) an. Innerhalb einer *Wettrunde* kann ein Spieler mehrfach an die Reihe kommen und damit Potenziert sich die Anzahl der möglichen Sequenzen an *Wettrunden*. Deshalb ist die Komplexität von NLHE HU mit $\approx 10^{71}$ ungleich größer.

[13] c = *Check*, r = *Raise*, mit Groß- und Kleinbuchstaben werden die zwei Spieler unterschieden

Die tatsächliche Komplexität hängt noch von der Höhe der *Stack*s beider Spieler ab. Bei einer *Stack* Größe von 500 *Big Blinds* ist die Anzahl der Spielzustände sogar $7{,}16 * 10^{75}$ [14][29]. Auch die Anzahl der Informationsmengen bewegt sich mit $7{,}23 * 10^{72}$ in nicht beherrschbaren Dimensionen, sowohl im Hinblick auf die Rechenzeit als auch auf den notwendigen Speicher. Damit ist NLHE HU ohne Reduktion der Komplexität auf heutigen Rechnern nicht lösbar.

4.4.1 Abstraktionen

Eine Reduktion der Komplexität kann durch Abstraktionen erreicht werden. Die Grundidee daran ist das ursprüngliche Spiel in ein weniger komplexes Spiel zu transformieren. Erreicht wird diese durch Reduktion des Baumes. Dabei werden Spieler Knoten mit *Gemeinschaftskarten* zusammengefasst, die eine ähnliche strategische Situation darstellen. Die Approximation des Nash-Equilibriums wird dann auf dem abstrahierten Spiel ausgeführt und benötigt weniger Speicher und Rechenzeit. Wichtig ist, dass die in dem kleineren Spiel gefundenen Strategien wieder zurück auf das größere Spiel transformiert werden können. Die Transformation muss also in beide Richtungen funktionieren, dargestellt in Abbildung 9 (aus [38]).

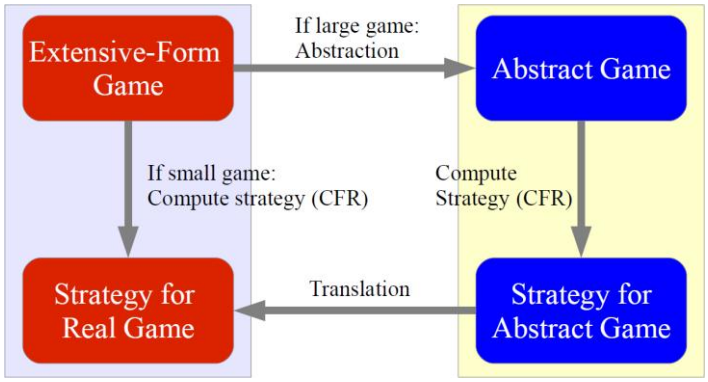


Abbildung 9: Abstraktion eines komplexen Spieles, Berechnung auf dem abstrahierten Spiel und Rücktransformation.

4.4.1.1 Verlustfreie Abstraktionen

Verlustfreie Abstraktionen fassen nur Knoten zusammen, die strategisch gesehen äquivalent sind. Das heißt, dass alle Teilbäume dieser Knoten identisch im Hinblick auf deren Struktur und die Gewinne in den Blättern sind. Eine solche, verlustfreie Abstraktion ist der Informationsmengen Baum (siehe Abschnitt 2.1.1), bei dem für einen Spieler nicht unterscheidbare Knoten zusammengefasst werden.

Eine verlustfreie Abstraktion lässt sich dadurch definieren, dass sich jede auf der Abstraktion berechnete Strategie in eine Strategie für das ursprüngliche Spiel transformieren lässt, ohne dass sich der erwartete Nutzen beider Strategien unterscheidet. Somit ist auch der erwartete Nutzen der optimalen Gegenstrategie, die innerhalb der Abstraktion berechnet wurde, gleich dem erwarteten Nutzen der optimalen Gegenstrategie, die im ursprünglichen Spiel ohne Verwendung der Abstraktion berechnet wurde. Über den Vergleich von ε der besten Gegenstrategien jeweils mit und ohne Abstraktion, lässt sich somit überprüfen, ob eine Abstraktion tatsächlich verlustfrei ist.

Über die Abstraktionen des Informationsmengen Baumes hinausgehend gibt es bei NLHE HU auch Spielsituationen, die für den Spieler unterscheidbar und gleichzeitig strategisch äquivalent sind. Zum

[14] 71593792563005030000147335394162504942066342923910716468991711327781134142 00

Beispiel sind für einen Spieler auf dem *Board* A♠K♦9♠7♠5♦ die sich nur in den Farben unterscheidenden *Handkarten* A♣2♣, A♣2♥, A♣2♦, A♥2♣, A♥2♥, A♥2♦, A♦2♣, A♦2♥ und A♦2♦ strategisch äquivalent. Da beim Poker Farben keine Reihenfolge der Wertigkeit besitzen, spielen sie nur dann eine Rolle, sofern der Spieler einen „*Flush*" (fünf Karten derselben Farben) bekommen könnte. Im Beispiel wäre das nur mit der Farbe ♠ möglich, die auf dem *Board* dreimal auftritt.

Über solche Farb-Isomorphismen hinaus existieren noch weitere, strategisch äquivalente Mengen an *Handkarten* und *Gemeinschaftskarten*. Zum Beispiel haben auf dem *Board* A♥K♥Q♥J♥T♥[15] alle 1.081 Kombinationen an möglichen *Handkarten* denselben Pokerrang, da für jeden Spieler die fünf *Gemeinschaftskarten* gewertet werden und die *Handkarten* irrelevant sind.

An diesen verlustfreien Beispielen lässt sich bereits das Potential der Komplexitätsreduktion Abstraktionen erkennen. Die größte Reduktion an Komplexität kann aber mithilfe von verlustbehafteten Abstraktionen erreicht werden.

4.4.1.2 Verlustbehaftete Abstraktionen

Bei verlustbehafteten Abstraktionen sind auch Zusammenfassungen von Knoten zugelassen, die aus strategischer Sicht nicht äquivalent sind. Damit werden zwangsläufig auch die Regeln des Spiels geändert, sodass sich die Ergebnisse aus einer verlustbehafteten Abstraktion nicht mehr isomorph auf das ursprüngliche Spiel übertragen lassen. Der Grund zur Nutzung verlustbehafteter Abstraktionen ist eine noch stärkere Reduktion der Komplexität. Es entsteht ein Trade-off zwischen geringerer Komplexität einerseits und einer Abstraktion, die die strategischen Gegebenheiten des Originalspiels möglichst realistisch abbildet.

In Fortführung des Beispiels aus dem letzten Abschnitt könnte eine verlustbehaftete Abstraktion zusätzlich noch die *Handkarten* A♦2♠, A♦2♠ und A♦2♠ auf dem *Board* A♠, K♦, 9♠, 7♠, 5♦ zusammenfassen. Diese Zusammenfassung ist zwar im Hinblick auf den *Pokerrang* (Paar Asse mit K, 9 und 7 als weitere Karten) identisch. Allerdings gilt für alle *Handkarten*, die 2♠ enthalten**,** dass dem Gegner eine Karte weniger für einen ♠-*Flush* zur Verfügung stünde und damit die Wahrscheinlichkeit auf einen gegnerischen *Flush* zu treffen geringer ausfällt. Deshalb ist diese Abstraktion nicht verlustfrei.

Allerdings gehen die Abstraktionen, die in der neuesten Generation an Pokerbots wie Slumbot NL (2. Platz 2013 ACPC) verwendet werden, im Hinblick auf die Verlustbehaftung weit über dieses Beispiel hinaus. Es werden sich ähnelnde Spielerkarten und *Gemeinschaftskarten* in Gruppen zusammengefasst. Die Anzahl der Gruppen wird im Vorfeld fest vorgegeben. Die Einteilung der Karten erfolgt mit einem Algorithmus zur Vektorquantisierung (wie z.B. k-Means [39]), der mit einem Abstandsmaß arbeitet [26]. Zusätzlich verwenden die neuesten Poker-Programme sogenannte Imperfect Recall Abstraktionen.

Imperfect Recall Abstraktionen Abstraktionen, die Spielzustände mit unterschiedlichen, sichtbaren Vorgeschichten zusammenfassen heißen Imperfect Recall Abstraktionen. Diese Abstraktionen, die aus Sicht eines Menschen ungewöhnlich erscheinen mögen, vergessen vergangene Aktionen aus vorhergehenden Wettrunden. Aus Sicht des Poker-Programms wird in jeder *Wettrunde* ein eigenes Spiel gespielt, indem der genaue Ablauf einer vorherigen *Wettrunde* nicht berücksichtigt

[15] Entspricht einem „Royal" *Flush*, der bestmöglich Straight Flush und damit die bestmögliche Hand bei Poker

wird. Als Information aus der Vorrunde wird lediglich die Anzahl der *Chips* im *Pot* übergeben. Selbstverständlich handelt es sich hierbei um verlustbehaftete Abstraktionen.

Auf Imperfect Recall Abstraktionen sind weder die Existenz eines Equilibriums garantiert, noch ist der CFR Vanilla Algorithmus wohldefiniert [40]. Allerdings ist der CFR mit korrekt justiertem Public-Chance Sampling (siehe Abschnitt 4.3.1.1) doch wohldefiniert, wenn durch das Sampling der nicht-wohldefinierte Teil des Spielbaums jeweils ausgeblendet wird [35, 36].

Interessanterweise zeigt sich in der Praxis, dass bei gleicher Anzahl an Knoten in der Abstraktion, Strategien die auf Abstraktionen mit Imperfect Recall berechnet wurden näher an das Nash-Equilibrium des ursprünglichen Spiels herankommen, als Strategien die mit verlustbehafteten Perfect Recall Abstraktionen erzeugt wurden [41]. Das Ergebnis liegt auf der Hand, denn Imperfect Recall Abstraktionen vergrößern den Raum erlaubten Abstraktionen. Diese Aussage ist allerdings nur bis zu einer bestimmten, hohen Grenze in Anzahl der Knoten richtig, ab der es nur noch möglich ist die Qualität der Perfect Recall Abstraktionen zu steigern, aber nicht mehr die Qualität der Imperfect Recall Abstraktionen.

Overfitting Bei der Annäherung an ein Nash-Equilibrium innerhalb einer verlustbehafteten Abstraktion kann es zum Overfitting Effekt kommen. Obwohl der CFR Algorithmus eine Strategie im reduzierten Spiel dem Nash-Equilibrium in jeder Iteration annähert, entfernt sie sich zugleich vom Nash-Equilibrium im Originalspiel.

Dieser Effekt kann durch Berechnungen der optimalen Gegenstrategie auf dem ursprünglichen Spiel sichtbar gemacht werden. Abbildung 10 (aus [41]) zeigt den Verlauf von ε innerhalb vieler CFR-Iterationen (auf der X-Achse ausgedrückt in Sekunden) einmal für das abstrahierte (rot) und einmal für das ursprüngliche Spiel (grün). Der Overfitting-Effekt tritt etwa ab einem Drittel der Zeitachse (X-Achse) in dem Punkt auf, ab dem ε für das ursprüngliche Spiel wieder ansteigt. Außerdem zeigt sich, dass durch die verlustbehafteten Abstraktionen ε im Minimum beim ursprünglichen Spiel ungefähr 280 mbb/g (280 Tausendstel *Big Blinds* pro Spiel) vom Nash-Equilibrium entfernt liegt. Bedenkt man, dass diese Strategie bei 100 Spielen gegen die optimale Gegenstrategie insgesamt 28 *Big Blinds* (*BB*/100 *Hände*) verliert, dann ist das ein großer Abstand zum Nash-Equilibrium. Im Vergleich gewinnen Profis gegen Amateure im Erwartungswert ≥ 5 *BB*/100 *Hände* [42].

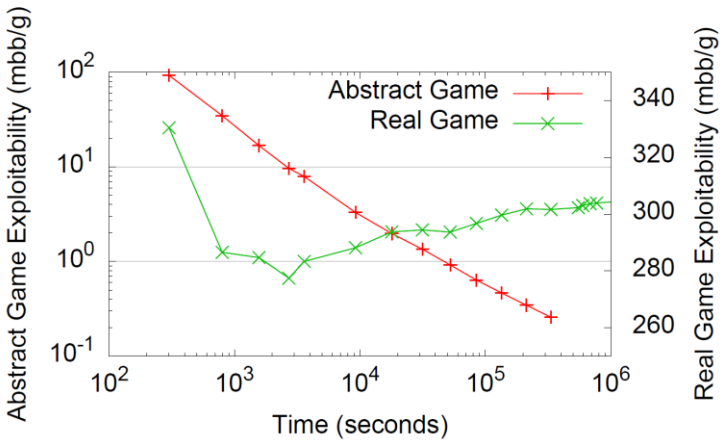


Abbildung 10: Zeigt Overfitting Effekt, durch den ε in der Abstraktion sinkt und gleichzeitg im ursprünglichen Spiel steigt.

4.4.1.3 Wettrunden

Im Hinblick auf die Komplexität durch die Karten sind *No Limit* und *Limit* identisch. Der Unterschied zwischen diesen beiden Varianten von ≈ 50 Größenordnungen geht allein auf die Komplexität der *Wettrunden* zurück. Deshalb reduzieren alle aktuellen Poker-Programme die verschiedenen Aktionen (Einsatz von *Chips*) des Spielers von ca. 100 auf ca. 3-6. Da dem Spieler durch diese Reduktion nicht alle Aktionen des ursprünglichen Spiels zur Verfügung stehen, ist sie verlustbehaftet.

Obwohl das Ausmaß dieser Reduzierung dramatisch wirkt, bedeutet sie in der Praxis keinen allzu großen Verlust an strategischer Vielfalt. Im praktischen Spiel kommen die allermeisten, der möglichen Einsatzhöhen, so gut wie gar nicht vor. Die Einsatzhöhe ist nämlich nicht absolut zu betrachten, sondern relativ zur Größe des *Pot*s. Ein Einsatz von \$1 bei einem *Pot* von \$50 ist aus strategischer wie mathematischer Sicht nicht sinnvoll. Das *Pot Odds* Konzepts sieht für diese Situation vor, dass der Gegenspieler nur in $\frac{1}{51} < 2\%$ der Fälle die besseren Karten besitzen muss, um damit sich Aktion *Call* amortisiert und diese damit automatisch erfolgt [43]. Der Einsatz ist sowohl für einen *Bluff* als auch für eine Value-*Bet*[16] zu gering.

Die Abstraktion der Einsatzhöhen berücksichtigt also immer nur den relativen Anteil am aktuellen *Pot*. Typischerweise werden die Einsatzhöhen 0,5 1,0 2,0 in Relation zum *Pot* sowie *All-In* für den ersten Einsatz in einer *Wettrunde* angeboten. Häufig wird in der Pokersoftware die Anzahl der verschiedenen Einsatzhöhen nach dem ersten Einsatz innerhalb derselben *Wettrunde* auf nur noch zwei Einsatzhöhen 1,0 und *All-In* beschränkt, um die Komplexität weiter zu reduzieren.

Um bei der Rücktransformation einer auf der Abstraktion gefunden Strategie ins ursprüngliche Spiel auf alle tatsächlich möglichen Einsatzhöhen reagieren zu können, gibt es verschiedene Optionen. Einerseits kann die Strategie jeden beliebigen Einsatz im Originalspiel auf den Einsatz mit der kürzesten Distanz im reduzierten Spiel abbilden. Das führt allerdings zu einer größeren Verzerrung der Realität. Besser ist es, die nach oben und unten hin nächstliegen Aktionen beide gleichzeitig zu wählen und die Strategien dieser beiden Teilbäume miteinander zu mischen.

Insgesamt lässt sich feststellen, dass es ohne Abstraktion der Setzstruktur derzeit nicht realistisch ist, den dafür notwendigen Speicherbedarf bereitzustellen. Deshalb werden die Aktionen der Spieler stark reduziert, was aber durch Einsatzhöhen relativ zum *Pot* nur zu überschaubaren Einschränkungen der strategischen Optionen führt.

4.4.1.4 Karten

Auch bei den Karten arbeiten die neuesten Poker-Programme mit verlustbehafteten Abstraktionen, die ähnliche Karten mithilfe eines Abstandsmaßes zu einer konstanten, vorab festgelegten Anzahl an Gruppen zusammenfassen. Die Gruppen repräsentieren dann Spielsituationen, die sich aus Sicht des Spielers strategisch ähneln.

Die wichtigsten Verfahren für das Abstandsmaß bei *Handkarten* sind die relative Handstärke und das Potential-aware Maß. Bei der relativen Handstärke handelt es sich um eine einzelne rationale Zahl $\in [0,1]$, die für *Handkarten* abhängig von der aktuellen *Wettrunde* den erwarteten Gewinnanteil am *Pot* gegen alle möglichen gegnerischen *Handkarten* ausdrückt. Beispielsweise beträgt die relative Handstärke für zwei Asse A♠A♦ als *Handkarten* bei zwei Spielern in der *Wettrunde Preflop* 85,2% und sinkt auf dem *Flop* mit den *Gemeinschaftskarten* 4♥, 5♥, 6♥ auf 60,1%.

[16] Eine Value-Bet erfolgt mit der stärkeren Hand und dient dazu, am Ende einen größeren Pot zu gewinnen

Der Nachteil dieses Maßes sind die fehlenden Informationen über den Verlauf der relativen Handstärke in den folgenden *Wettrunden*. Außerdem gibt die relative Handstärke in den *Wettrunden* 1 bis 3 keine Auskunft darüber, wie die Verteilung der relativen Handstärke über die noch möglichen *Board*s in der 4. *Wettrunde* (*River*) aussieht. Angenommen zwei verschiedene *Handkarten* h_1 und h_2 haben *Preflop* dieselbe relative Handstärke von 75%. h_1 besiegt aber in 100% der Blätter 75% der gegnerischen *Handkarten* und h_2 besiegt hingegen in 50% der Blätter alle gegnerischen *Handkarten* und in den anderen 50% der Blätter liegt h_2 vergleichsweise nur im Mittelfeld. Durch diesen Unterschied sind beide *Handkarten* strategisch unterschiedlich zu behandeln. Das Potential von h_2 ist wesentlich größer, weil es sich für die Höhe des Gewinns in einer *Spielrunde* sehr positiv auswirkt, sofern der Gegner selbst *Handkarten* mit hoher, relativer Handstärke besitzt, die aber trotzdem von den eigenen *Handkarten* geschlagen werden.

Beide Probleme werden in dem Potential-aware Maß gelöst, welches mit Histogrammen über die relative Handstärke auf der 4. *Wettrunde* (*River*) arbeitet. Dabei wird für alle *Handkarten* ein Histogramm erstellt, dass die verschiedenen, relativen Handstärken auf der 4. *Wettrunde* berücksichtigt [44]. Für das Beispiel mit h_1 und h_2 ergeben sich dann unterschiedliche Histogramme. Das Histogramm von h_2 zeigt zwei große Ausschläge, einmal bei einer relativen Handstärke von 100% und einmal bei 50%.

In der Abbildung 11 (aus [45] entnommen) ist auf der linken Hälfte ein Beispiel für zwei Histogramme der *Handkarten* 4♠4♥ und J♠T♠ dargestellt. Auf der x-Achse ist die relative Handstärke für die letzte *Wettrunde* (*River*) aufgetragen, auf der y-Achse die relative Häufigkeit der relativen Handstärke. Obwohl die relative Handstärke beider *Handkarten Preflop* bei 57% liegt, unterscheiden sich die Histogramme klar. Die *Handkarten* 4♠4♥ haben höhere Ausschläge bei einer Handstärke im Bereich 0,35 bis 0,65, wohingegen J♠T♠ stärkere Ausschläge im oberen und unteren Bereich hat. In der Praxis zeigt sich, dass *Handkarten* wie J♠T♠ mit mehr Potential im oberen Bereich stärker sind als *Handkarten* die häufig im Mittelfeld liegen. Die Karten mit hohem Potential kommen häufiger in Situationen in denen der Gegner auch sehr starke *Handkarten* hält, aber trotzdem unterliegt. In solchen Situationen wird meist *All-In* gespielt und maximal viele *Chips* gewonnen.

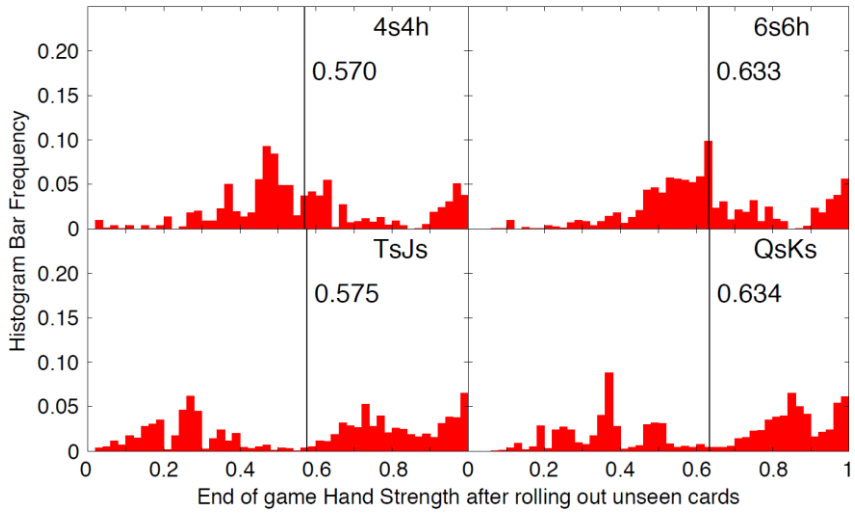


Abbildung 11: Histogramme von *Handkarten* mit gleicher, relativer Handstärke aber unterschiedlichem Potential.

In den heutigen Programmen werden *Gemeinschaftskarten* und *Handkarten* in Gruppen zusammengefasst, die möglichst aus Sicht des Spielers ähnliche, strategische Situationen repräsentieren. Für jede Wettrunde gibt es eine feste Anzahl dieser Gruppen. Beträgt in der ersten *Wettrunde* w_1 (*Preflop*) die Anzahl der verschiedenen Gruppen g, dann werden auf der zweiten

Wettrunde (*Flop*) g^2 und auf der i. Wettrunde g^i verschiedene Gruppen benötigt, sofern sich der Spieler vergangener Zustände (Perfect Recall) bewusst sein soll. Der große Teil der Komplexität entsteht also durch die Speicherung der Informationen zur Unterscheidung von Spielzuständen auf vorherigen *Wettrunden*. Hier zeigt sich die Mächtigkeit von Imperfect Recall Abstraktionen, die nicht darauf angewiesen sind, die vergangen Zustände unterscheiden zu können.

Slumbot NL hat neben den Gruppen die *Handkarten* x *Gemeinschaftskarten* enthalten auch noch eigene Gruppen, die nur *Gemeinschaftskarten* enthalten. Die Gruppenhierarchie ist aber so aufgebaut, dass es für jede Gruppe an *Boardkarten* eigene Gruppen von *Handkarten* x *Gemeinschaftskarten* gibt [26]. Dieses Modell ermöglicht auf Kosten einer geringeren Reduktion eine bessere Abbildung des Originalspiels, bei dem bis auf Farb-Isomorphismen *Gemeinschaftskarten* strategisch unterschiedlich zu behandeln sind.

An dieser Stelle setzt diese Arbeit an und führt im nächsten Kapitel Abstraktionen ein, die im Vergleich zum Stand der Technik wesentlich verlustfreier sind.

5 Modell: Verlustfreiere Abstraktionen

Dieses Kapitel konzentriert sich auf die eigenen Beiträge, die über den Stand der derzeitigen Forschung hinausgehen. Im Zentrum steht die Frage, wie die im letzten Kapitel vorgestellten Konzepte angepasst werden müssen, um ein Nash-Equilibrium mithilfe von weniger verlustbehafteten oder sogar verlustfreien Abstraktionen effizient berechnen zu können.

Die Berechnung des exakten Nash-Equilibriums für NLHE HU ist mit heutiger Rechenleistung nicht realistisch. Im Bereich Computer-Poker stammen die wichtigsten Forschungsergebnisse von der CRPG der Universität von Alberta, die seit 1997 an diesem Thema forscht. Aufgrund der vergleichsweise geringeren Speicher- und Rechenressourcen in dieser Zeit fokussierte sich die Forschung auf starke Komplexitätsreduktionen durch verlustbehaftete Abstraktionen, die nach einem Abstandsmaß ähnliche Spielsituationen zusammenfassen, sowie auf die Beschleunigung des CFR mittels Sampling-Techniken. Damit ist es gelungen für ein wesentlich weniger komplexes Spiel ein Nash-Equilibrium mit einem sehr geringen ε innerhalb der Abstraktion zu berechnen. Die Ergebnisse der ACPC sprechen eine deutliche Sprache und zeigen die Richtigkeit dieses Ansatzes. Programme, die den CFR Algorithmus zur Berechnung von ε-Nash-Equilibria verwenden, waren Programmen mit anderen Ansätzen bei weitem überlegen.

Die Qualität der Strategien wird allerdings durch die Verwendung von verlustbehafteter Abstraktion des Originalspiels beschränkt. Zwar ist innerhalb einer verlustbehafteten Abstraktion eine beliebig nahe Annäherung von ε an Null möglich, aber wie in 4.4.1.2 gesehen und in [41] gezeigt wurde, tritt bei Verwendung von verlustbehafteten Abstraktionen ab einem bestimmten Punkt der Overfitting-Effekt auf und ε steigt im Originalspiel trotz sinkendem ε im abstrahierten Spiel. Die Ursache sind die Grenzen von verlustbehafteten Abstraktionen. Um weitere Verbesserung in der Qualität der Strategien zu erreichen, ist es notwendig die Verlustbehaftung der Abstraktion zu reduzieren und letztlich mit verlustfreien Abstraktionen zu arbeiten.

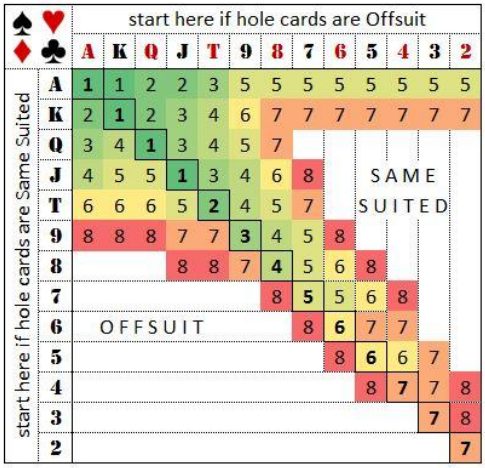
Im Kern geht es darum, die Qualität der Abstraktionen zu verbessern. Zur Abschätzung der Qualität wird zuerst eine Strategie unter Verwendung der Abstraktion berechnet. Anschließend wird für diese Strategie ε innerhalb der Abstraktion und ε im ursprünglichen Spiel bestimmt. Entscheidend ist der Abstand dieser beiden ε zueinander. Je größer der Abstand, desto mehr werden die ursprünglichen Spielregeln durch die Abstraktion verzerrt und desto verlustbehafteter ist die Abstraktion. Bei einem Abstand von Null liegt eine verlustfreie Abstraktion vor.

Deshalb ist ein wichtiges Ziel dieser Arbeit den Trade-off zwischen Komplexitätsreduktion und verlustfreier Abstraktion zugunsten von verlustfreien Abstraktionen zu wählen. Es werden Karten-Abstraktionen verwendet, die wesentlich geringere bzw. keine Verluste im Vergleich zum ursprünglichen Spiel aufweisen. Um die dadurch gesteigerte Komplexität zu beherrschen, wird das Gesamtproblem mittels Dekomposition aufgeteilt.

5.1 Dekomposition in Post-Flop Probleme

Zur Beherrschung der gesteigerten Komplexität durch Verwendung weniger verlustbehafteter Abstraktionen entstand die Idee ein ε-Nash-Equilibrium für NLHE HU erst ab der 2. *Wettrunde*, dem *Flop*, bis zur letzten *Wettrunde* (*River*) zu berechnen. Sofern eine Strategie mit geringem ε für die 1. *Wettrunde* (*Preflop*) bereits bekannt ist, können alle möglichen Kombinationen an *Gemeinschaftskarten* auf dem *Flop* getrennt voneinander berechnet werden. Damit wäre es möglich, eine Strategie für das Gesamtproblem mit einer bisher unerreichten Qualität zu finden.

Interessanterweise wird in realen Pokerspielen im professionellen Bereich für die *Preflop Wettrunde* regelmäßig auf feste Strategien zurückgegriffen. In Analogie zum Schach gibt es auch bei NLHE Eröffnungszüge die in Abhängigkeit von den *Handkarten* die Einsätze zu Beginn einer Spielrunde vorgeben.



Rank	Hole Cards	Playable Positions
1	AA, KK, QQ, JJ; AKs	Early, Middle, Late
2	TT; AQs, AJs, KQs; AKo	Early, Middle, Late
3	99; ATs, KJs, QJs, JTs ; AQo	Early, Middle, Late
4	88; KTs, QTs, J9s, T9s, 98s; AJo, KQo	Early, Middle, Late
5	77; *A9s thru A2s* , Q9s, T8s, 97s, 87s, 76s; KJo, QJo, JTo	Early*, Middle, Late
6	66, 55; K9s, J8s, 86s, 75s, 54s; ATo, KTo, QTo	Middle**, Late
7	44, 33, 22; *K8s thru K2s,* Q8s, T7s, 64s, 53s, 43s; J9o, T9o, 98o	Late***
8	J7s, 96s, 85s, 74s, 42s, 32s; A9o, K9o, Q9o, J8o, T8o, 87o, 76o, 65o, 54o	

* Playable in early position if game is loose/passive.
** Playable in middle position if game is loose/passive.
***Playable in late position if you are the first to bet.

Abbildung 12: Bekannte *Preflop Handkarten* Wertung (links) aus der dann eine *Preflop* Strategie (rechts) abgeleitet wird.

In Abbildung 12 (aus [46]) ist eine bekannte, inzwischen veraltete Bewertung der 169 verschiedenen Äquivalenzklassen für *Handkarten Preflop* aus dem Jahr 1999 zu sehen, die jeder Klasse eine Bewertung im Bereich von 1-9 zuweist. Auf Grundlage der Bewertung wird für jede *Preflop* Situation eine bestimmte Aktion vorgeschlagen.

5.1.1 Reduktion der Komplexität

Die Komplexitätsreduktion durch die Dekomposition hat also ihren Ursprung in der Beschränkung auf genau einen *Flop* und genau eine Sequenz an *Preflop*-Aktionen.

Insgesamt gibt es $\binom{52}{3} = 22100$ verschiedene Kombinationen an *Gemeinschaftskarten* auf dem *Flop*. Werden *Gemeinschaftskarten* zusammengefasst, die aufgrund von Farb-Isomorphismen äquivalent sind, sinkt diese Zahl um mehr als eine Größenordnung.

Zum Beispiel ist der *Flop* 9♠7♠3♠ äquivalent zu den *Flop*s 9♦7♦3♦, 9♣7♣3♣, 9♥7♥3♥. Eine Strategie die auf einem dieser *Flop*s berechnet wurde, kann durch Transformation der entsprechenden Farbe auf die anderen drei *Flop*s übertragen werden. Zur Bestimmung der Anzahl der Äquivalenzklassen wird zuerst die Anzahl der *Flop*s unter der Annahme bestimmt, dass die Karten zwar Wertigkeiten aber keine Farben besitzen.

1. *Flop*s mit 3 verschiedenen Wertigkeiten (z.B. 5, 7, K): $\binom{13}{3} = 286$.
2. *Flop*s mit 2 verschiedenen Wertigkeiten (z.B. 5, 5, K): $13 * 12 = 156$.
3. *Flop*s bei denen alle Karten dieselbe Wertigkeit besitzen (z.B. 5, 5, 5): 13.

Nun werden die Farben hinzugenommen, wodurch es für jeden dieser drei Fälle eine unterschiedliche Zahl an verschiedenen Äquivalenzklassen von *Flop*s gibt.

1. Alle Karten haben a) dieselbe Farbe, b) unterschiedliche Farben oder die Karten haben c) zwei verschiedene Farben (zwei Karten mit derselben Farbe) wovon es drei Permutationen existieren. Insgesamt sind das fünf verschiedene Äquivalenzklassen.
2. Eine Karte des Paares hat a) eine unterschiedliche Farbe oder b) dieselbe Farbe wie die ungepaarte Karte. Daraus ergeben sich zwei verschiedenen Äquivalenzklassen.
3. Alle Karten haben verschiedene Farben. Dies ergibt nur eine Äquivalenzklasse.

Daraus ergeben sich $286 * 5 + 156 * 2 + 13 * 1 = 1755$ verschiedene Äquivalenzklassen an *Gemeinschaftskarten* auf dem *Flop*.

Die Anzahl der möglichen *Preflop* Setz-Sequenzen wird mit 835 angenommen, was der Anzahl an Sequenzen von Slumbot NL entspricht [26], einer Pokersoftware, die alle vier *Wettrunden* durch Verwendung von stark verlustbehafteten Karten-Abstraktionen und Imperfect Recall abdeckt. Es müssen alle Setz-Sequenzen unterschieden werden, weil jede davon zu einer unterschiedlichen *Range* an *Handkarten* beider Spieler auf der 2. *Wettrunde* führt und eventuell auch zu einer unterschiedlichen Größe des *Pot*s.

Damit ist für eine Kombination an *Gemeinschaftskarten* auf dem *Flop* das Problem durch die Dekomposition insgesamt um Faktor $1755 * 835 \approx 1{,}4 * 10^6$ kleiner, als das Gesamtproblem. Dieser Vergleich gilt unter Verwendung verlustfreier Abstraktionen der *Gemeinschaftskarten* auf dem *Flop*.

Durch die verringerte Komplexität ist es außerdem möglich ε auch auf dem ursprünglichen Spiel zu berechnen. Damit kann die Qualität der Abstraktionen in der Evaluierung in Abschnitt 7.3 wesentlich zuverlässiger bestimmt werden, als in den neuesten, wissenschaftlichen Papern (wie [45]).

5.2 Abstandsmaße

Die in dieser Arbeit verwendeten Karten-Abstraktionen werden ebenfalls mithilfe eines Abstandsmaßes bestimmt. Im Unterschied zum etablierten Verfahren wird keine feste Anzahl von Gruppen vorgegeben. Stattdessen wird statt Ähnlichkeit die Identität verwendet und eine Gruppe enthält nur *Handkarten* mit einem paarweisen Abstand von 0. Dadurch ist die Anzahl der Gruppen davon abhängig, welche *Handkarten* auf Basis des Abstandsmaßes identisch sind.

Um den Grad der Verlustbehaftung zu variieren, werden drei aufsteigend striktere Abstandsmaße eingeführt.

5.2.1 Absoluter Pokerrang

Dieses Maß bewertet *Handkarten* in einem Spielzustand nach dem absoluten *Pokerrang* auf allen möglichen *Gemeinschaftskarten* auf dem *River*. Mit dem absoluten *Pokerrang* (siehe Abschnitt 2.2) ist die Stärke nach den Pokerregeln gemeint, z.B. „ein Paar Zehner mit weiteren Karten K, 5, 4“.

Mit dem *Pokerrang* werden Farb-Isomorphismen zusammengefasst (z.B. sind auf dem *Flop* Q♥J♥2♣ die *Handkarten* 7♠7♦, 7♠7♣, 7♣7♦ identisch). Neben den Farb-Isomorphismen werden viele weitere Kombinationen, die nach den Pokerregeln identisch sind, zusammengefasst (z.B. werden auf dem *River* Q♥J♥2♣Q♦J♣ alle *Handkarten* außer einem Paar Zweier (22) zusammengefasst, die nur Karten mit einer Wertigkeit unter J enthalten, wodurch eine Gruppe mit insgesamt 627 *Handkarten entsteht*).

Aus der Identität des *Pokerrang*s zweier *Handkarten* folgt aber nicht die strategische Äquivalenz. Im obigen Beispiel sind die *Handkarten* 7♠7♦ und 7♠7♣ nicht strategisch äquivalent. Die *Handkarten*

7♠7♣ sind strategisch etwas wertvoller, weil durch den Besitz einer ♣–Karte auf bestimmten *Board*s ein ♣–*Flush* des Gegners geringfügig unwahrscheinlicher wird.

Ein Vorteil des Verfahrens ist die geringe Laufzeit. Für n *Handkarten* muss für jedes Paar an *Handkarten* der *Pokerrang* auf allen *River-Board*s berechnet werden. Anschließend wird die Identität für alle anderen *Handkarten* geprüft. Der Algorithmus hat damit zwar einen Aufwand von $O(n^2 * b)$. Dabei ist n für *Hold'em* auf $\binom{49}{2} = 1176$ und b auf $\binom{47}{2} = 1081$ beschränkt.

5.2.2 Relative Handstärke

Die relative Handstärke gibt die Wahrscheinlichkeit an, einen *Showdown* gegen alle möglichen *Handkarten* des Gegners zu gewinnen. Wie in Abschnitt 4.4.1.4 beschrieben, verwenden die neuesten Poker Programme die relative Handstärke im „potential-aware-Maß".

Mit der relativen Handstärke tritt das Problem aus dem letzten Beispiel nicht auf, da für 7♠7♣ (relative Handstärke 58,2%) im Vergleich zu 7♠7♦ (relative Handstärke 58,0%) weniger mögliche *Handkarten* des Gegners existieren, die einen *Flush* bilden. Aufgrund der unterschiedlichen relativen Handstärke werden beide *Handkarten* bei Verwendung der Identität in getrennte Gruppen eingeordnet. Diese Abstraktion führt daher zu einem geringeren Verlust als der *Pokerrang*.

Trotzdem führt die relative Handstärke selbst unter Verwendung der Identität nicht zu verlustfreien Abstraktionen. Zum Beispiel werden auf dem *Board* Q♥J♥2♣4♦7♥ die *Handkarten* A♥K♠ und A♠K♥ zusammengefasst, weil beide *Handkarten* jeweils dieselbe erwartete relative Handstärke von 37,1% haben. Trotzdem sind die *Handkarten* A♥K♠ strategisch gesehen stärker als A♠K♥, weil sie den höchsten ♥-*Flush* mit einem Ass und damit die bestmöglichen *Handkarten*[17] des Gegners ausschließen.

Die gesteigerte Genauigkeit führt aber zu einer höheren Laufzeit von $O(n^3 * b)$, weil zusätzlich zum absoluten *Pokerrang* für jedes Paar an *Handkarten* auch der *Pokerrang* für alle *Handkarten* des Gegners bestimmt werden muss.

5.2.3 Gegner-Sicht

Um den Fehler aus dem letzten Beispiel zu beheben, entstand die Idee, die relative Handstärke zusätzlich aus der Sicht aller gegnerischen Karten einzubeziehen. Damit wird für *Handkarten* nicht nur eine Zahl wie *Pokerrang* oder relative Handstärke bestimmt, sondern zusätzlich ein Vektor in der Länge der Anzahl der gegnerischen Hände. Für jedes Paar an *Handkarten* des Gegners und für die eigenen *Handkarten* existiert ein Eintrag im Vektor, der der relativen Handstärke des Gegners entspricht.

Dieses neue Abstandsmaß entspricht auch der Wahrnehmung von Pokerprofis, für die es bei NLHE eher darum geht Entscheidungen in Abhängigkeit der *Handkarten* des Gegners zu treffen, als Entscheidungen in Abhängigkeit der eigenen *Handkarten* [16]. Die Identität ist dann gegeben, wenn bei gleicher relativer Handstärker zusätzlich die zwei Gegner-Vektoren dieselben Einträge enthalten bzw. wenn beide Vektoren im sortierten Zustand identisch sind. Damit ist die Identität der Gegner-Sicht strikter definiert als die der relativen Handstärke.

[17] Die sog. „Nuts" sind die bestmöglichen Hand*karten*, die auf einem *Board* möglich sind

So werden die *Handkarten* A♥K♠ und A♠K♥ aus dem obigen Beispiel korrekt in zwei verschiedene Gruppen aufgeteilt. Im Unterschied zu A♥K♠ beinhaltet die Gegner-Sicht für A♠K♥ insgesamt acht *Handkarten* des Gegners mit einer relativen Handstärke von 100%, weil der höchste ♥-*Flush* möglich wird.

Aber auch für das Gegner-Sicht-Histogramm lassen sich Beispiele konstruieren, die die strategische Äquivalenz verletzen. Auf dem *Board* Q♥J♥J♣Q♦7♠ sind alle *Handkarten* die ein Ass und eine Karte unter einem Jack (ohne 7) enthalten aus Gegner-Sicht, sowie allen vorherigen Abstandsmaßen, identisch. Aus strategischer Sicht gibt es auf diesem *Board* trotzdem einen Unterschied zwischen A♥2♠ und z.B. A♥9♠. In der Strategie des Gegners sollten höhere Karten normalerweise häufiger gespielt werden und damit schließt A♥9♠ mehr *Handkarten* beim Gegner mit geringerer Handstärke aus, als A♥2♠.

Daran erkennt man auch, dass eine strategisch in allen Fällen korrekte Zusammenfassung von *Handkarten* nur mit einer auf Farb-Isomorphismen beschränkten Abstraktion möglich ist.

Die Laufzeit des Gegner-Sicht-Histogramms steigt noch einmal um Faktor n auf $O(n^4 * b)$, da die Handstärke aller möglichen anderen *Handkarten* für jede gegnerischen *Handkarten* berechnet werden müssen. Das entspricht für die Berechnung der Abstraktion auf einem *Flop* einer Komplexität von $\approx 10^{15}$.

5.3 Prüfung der Identität

Für jedes Paar an *Handkarten* und für alle im aktuellen Spielzustand noch möglichen *Gemeinschaftskarten* auf dem *River* erzeugen *Pokerrang* sowie relative Handstärke eine Zahl und die Gegner-Sicht einen Vektor. Mithilfe dieser Vergleichswerte kann die Identität mit drei unterschiedlichen Methoden überprüft werden. Die folgenden drei Methoden können mit allen Abstandsmaßen aus Abschnitt 5.2 kombiniert werden. Für jede Kombination wird in der Evaluierung die Komplexität (Abschnitt 7.2) und Qualität der erzeugten Strategien (Abschnitt 7.3) verglichen.

5.3.1 Globales Histogramm

Für jedes Paar an *Handkarten* gibt es ein Histogramm das die Häufigkeiten enthält, mit der die Vergleichswerte auftreten. Verschiedene *Handkarten* werden genau dann in dieselbe Gruppe zusammengefasst, wenn das zugehörige, globale Histogramm identisch ist.

5.3.2 Hierarchisches Histogramm

Im Vergleich zum globalen Histogramm, bewertet das hierarchische Histogramm den *Flop* unterschiedlich. Sofern sich der Spielzustand auf dem *Flop* befindet, also noch zwei *Gemeinschaftskarten* ausstehen, wird für jede mögliche *Gemeinschaftskarte* auf dem *Turn* ein eigenes Histogramm erzeugt. Zwei *Handkarten* h_1 und h_2 werden dann zusammengefasst, wenn es eine bijektive Abbildung für die Histogramme von h_1 auf h_2 gibt, die jeweils identische Histogramme miteinander verbindet. Anschaulich dargestellt gibt es für jede strategische Situation von h_1 in der *Wettrunde Turn* eine identische strategische Situation von h_2. Diese Situationen müssen nicht zwingend auf derselben *Gemeinschaftskarte* auftreten, aber die strategische Situation muss für beide *Gemeinschaftskarten* identisch sein.

Das Bilden von einem Histogramm für jede *Gemeinschaftskarte* auf dem *Turn* hat den Vorteil, dass *Handkarten* nur dann zusammengefasst werden, wenn die Situation nicht nur auf dem *River* identisch ist, sondern auch die strategische Situation auf dem *Turn*.

5.3.3 River Identität

Die River Identität vergleicht *Handkarten* nicht auf Basis von Histogrammen. Zwei verschiedene *Handkarten* sind nur dann identisch, wenn deren Vergleichswerte in allen Blättern des Baumes identisch sind. Der Sonderfall, dass eine der beiden *Handkarten* nicht möglich ist, weil die anderen *Handkarten* mindestens eine *Gemeinschaftskarte* enthalten, wird wiederrum mit einem Histogramm aufgelöst. In diesem Fall wird nur der Vergleichswert der validen *Handkarten* in ein Histogramm eingetragen. Für die Identität müssen dann zusätzlich die Histogramme der einseitig validen *Handkarten* übereinstimmen.

5.4 CFR mit *Handkarten*gruppen

Durch die Abstandsmaße aus Abschnitt 5.2 und die Methoden zur Identitätsprüfung aus Abschnitt 5.3 werden *Handkarten* auf dem *Flop* in Gruppen zusammengefasst. Jede Gruppe besteht aus *Handkarten*, die identisch sind. Der in Abschnitt 4.3 vorgestellte CFR Algorithmus basiert allerdings auf der Berechnung mit einzelnen *Handkarten*. Um den CFR auf einer Abstraktion mit Gruppen von *Handkarten* anzuwenden, sind zwei Anpassungen notwendig.

Wird der Pokerrang zweier Gruppen auf dem *River* verglichen, wird dazu einfach aus jeder Gruppe ein Element als Repräsentant verwendet. Das ist möglich, weil alle *Handkarten* einer Gruppe einen Abstand von 0 besitzen. Der Vergleich wird dann zwischen diesen beiden Repräsentanten durchgeführt.

Zur Berechnung von bedingten Wahrscheinlichkeiten für Gruppen G_a und G_b von *Handkarten* muss allerdings folgende Anpassung vorgenommen werden:

$$p(G_a|G_b) = \frac{\sum_{a\in G_a, b\in G_b, a\cap b=\emptyset} 1}{\sum_{a\in G_a, b\in G_b} 1} * \sum_{a\in G_a} \frac{1}{\binom{52-z(G_b)}{z(G_a)}}, \qquad z(G) = z(a \in G)$$

Der erste Faktor in der Formel entspricht der Anzahl der paarweise validen *Handkarten* (*Handkarten* ohne Kollision mit *Gemeinschaftskarten*) in Relation zu allen paarweisen Vergleichen. Es ist eine Zahl im Bereich [0,1], die angibt, wie viele Kombinationen der beiden Kartengruppen in der Realität auftreten. Der zweite Faktor summiert die Wahrscheinlichkeit für das Auftreten eines Elements in G_a auf. Die Formel ist nur unter der Bedingung gültig, dass alle Elemente einer Gruppe dieselbe Anzahl an Karten haben. Da Gruppen entweder nur *Gemeinschaftskarten* oder nur *Handkarten* enthalten ist diese Bedingung gegeben.

5.4.1 Änderung der *Handkarten*gruppen in jeder Wettrunde

Die für den *Flop* berechneten *Handkarten*-Gruppen können unverändert für die *Wettrunden Turn* und *River* verwendet werden. Damit kann der CFR mit den obigen Funktionen ausgeführt werden.

Alternativ ist es aber möglich, die *Handkarten*-Gruppen mit jeder neuen *Gemeinschaftskarte* neu zu bestimmen. Dadurch wird die Komplexität der Abstraktion weiter verringert, weil die Anzahl der

Gruppen mit jeder *Wettrunde* sinkt. Der Grund für diese Reduktion liegt darin, dass sich mit jeder weiteren *Gemeinschaftskarte* die Anzahl der erreichbaren Blätter unabhängig vom Abstandsmaß und des Verfahrens zur Identitätsprüfung verringert. Jeder Teilbaum eines Knotens hat weniger Blätter als der Teilbaum des Vorgängerknotens. Mit der Reduzierung der möglichen Blätter verringert sich auch die Varianz der Histogramme und damit die Anzahl der Gruppen.

Durch die Neuberechnung der Gruppen in jeder *Wettrunde* kommt es außerdem vor, dass sich die *Handkarten* einer Gruppe auf dem *Flop* auf mehrere Gruppen auf dem *Turn* aufteilen, obwohl es auf dem *Turn* insgesamt weniger Gruppen gibt.

Mit den sich ändernden Gruppen auf jeder *Wettrunde* muss auch die Berechnung der *Handkarten*-Wahrscheinlichkeiten in den Blättern (Abschnitt 4.1.2), sowie der CFR Algorithmus (Abschnitt 4.3) angepasst werden. Beide Algorithmen basierten bisher darauf, dass *Handkarten* sowohl in Richtung der Wurzel als auch in Richtung der Blätter einen eindeutigen Vorgänger bzw. Nachfolger hatten.

5.4.1.1 Anpassungen zur Berechnung der Handkarten-Wahrscheinlichkeiten in Endzuständen

Für die Berechnung der *Handkarten*-Wahrscheinlichkeiten in den Blättern wird die Menge der kleinsten Gruppen Γ_{min} benötigt, die die kleinste, gemeinsame Zusammenfassung von *Handkarten* in Gruppen über alle *Wettrunden* enthält.

Die Menge S enthält alle Wettrunden (*Flop*, *Turn*, *River*) und damit ist G_i die *Handkarten* Gruppe der i. Wettrunde, die die *Handkarten* h enthält. Der Algorithmus 1 wird jetzt nicht mit einzelnen *Handkarten*, sondern auf allen Gruppen von *Handkarten* berechnet, die in Γ_{min} enthalten sind. Da Γ_{min} durch den Schnitt über alle Gruppen gebildet wird, ist die Anzahl der Gruppen in Γ_{min} größer als die Anzahl der Gruppen der tatsächlich verwendeten Abstraktion. Da die Gruppen Γ_{min} durch den Schnitt über alle Gruppen gebildet wurden, kann jede Gruppe in den Blättern aus mehreren Elementen von Γ_{min} bestehen.

$$\Gamma_{min} = \bigcup_{h \in H} \left\{ G_{\min_h} \right\}, \qquad G_{\min_h} = \bigcap_{i \in S} G_i, h \in G_i$$

Diese Zerlegung der ursprünglichen Gruppen wird zur Berechnung der bedingten Wahrscheinlichkeiten in den Endzuständen benötigt. Angenommen es würde die bedingte Wahrscheinlichkeit einer Gruppe G auf dem *River* unter der Annahme berechnet werden, dass bestimmte Karten bereits bekannt wären, so könnten diese Karten einzelne *Handkarten* in G_i aufgrund von gleichen Karten invalideren und daher eine Wahrscheinlichkeit von Null haben (Kollision). Da der Gegner in späteren *Wettrunden* vor dem *River*, für die damals noch nicht zusammengefassten Elemente von G unterschiedliche Wahrscheinlichkeitsverteilungen in seiner Strategie haben könnte, reicht die Informationen über die Gesamtwahrscheinlichkeit der Gruppe im Endzustand nicht aus. Da einzelne Teile der Gruppe invalide sein könnten, müssen für diese einzelnen Teile separate Wahrscheinlichkeiten vorliegen. Nur dann kann μ_c für jedes Paar an möglichen *Handkarten* unter Berücksichtigung von Kollisionen korrekt berechnet werden.

5.4.1.2 Anpassung des CFR Algorithmus

Bei gleichbleibenden Gruppen konnte der CFR Algorithmus nach den bekannten Regeln den erwarteten Nutzen Richtung Wurzel nach oben propagieren. Bei auf jeder *Wettrunde* sich ändernden Gruppen kann es beim Übergang eines *Dealer*-Knotens hin zu Wurzel mehrere Vorgänger an

Handkarten-Gruppen geben. Der erwartete Nutzen einer Gruppe G_o oberhalb des *Dealer*-Knotens ergibt sich aus dem erwarteten Nutzen aller Gruppen Γ_U unterhalb des *Dealer*-Knotens, deren Schnitt mit G_o nicht leer ist:

$$\mu_c(G_o) = \frac{\sum_{G_U \in \Gamma_U, G_U \cap G_o \neq \emptyset} \mu_c(G_U) * |G_U \cap G_o|}{|G_o|}$$

Um den erwarteten Nutzen der oberen Gruppe G_o zu berechnen, wird zuerst die Summe über dem erwarteten Nutzen der unteren Gruppen Γ_U gebildet, die jeweils mit der Anzahl der gemeinsamen Elemente von unterer und oberer Gruppe gewichtet sind. Diese Summe wird dann in Relation zu der gesamten Anzahl der Elemente der oberen Gruppe gesetzt.

Auch wenn die Formel es anders vermuten lässt, gilt in aller Regel $G_O \subseteq G_U$, denn in späteren Wettrunden sind mehr *Gemeinschaftskarten* bekannt und damit können die *Handkarten* in größere Gruppen zusammengefasst werden. Die Übertragung des erwarteten Nutzens von G_U auf G_O ist allerdings nur dann korrekt, wenn alle *Handkarten* in G_U denselben Nutzen aufweisen. Diese Bedingung ist nur für verlustfreie Abstraktionen gegeben und deshalb entsteht hier ein Fehler, der von der Verlustbehaftung der Abstraktion abhängt. In der Evaluierung in Abschnitt 7.3 zeigt sich, dass dieser Fehler sehr gering ist.

5.5 Gemeinschaftskarten Äquivalenzgruppen

Neben den *Handkarten* können auch *Gemeinschaftskarten* in Gruppen zusammengefasst werden. Die Methode der Wahl bei den bekannten Poker-Programmen ist die Zusammenfassung von *Handkarten* und *Gemeinschaftskarten* in einer festen Anzahl von Gruppen mit ähnlichen Situationen [24].

Die Beziehung von *Gemeinschaftskarten* und *Handkarten* ist in dieser Arbeit hierarchisch aufgebaut. In einem *Dealer*-Knoten bildet jede *Gemeinschaftskarte*n-Gruppe einen Unterbaum. Für jeden dieser Unterbäume wird eine eigene *Handkarten*-Abstraktion berechnet.

Um äquivalente, verlustfreie Gruppen für die *Board*-Karten zu erzeugen, werden ausschließlich Farb-Isomorphismen verwendet. Deshalb werden in der Abstraktion nur Farben unterschieden, die einen *Flush* auf dem *River* ermöglichen. So hängt die Anzahl der *Board*-Gruppen von den bisher schon aufgedeckten Karten auf dem *Flop* ab.

Auf dem *Board* Q♥J♠5♣Q♦ ist für keine *Gemeinschaftskarte* auf dem *River* ein *Flush* möglich, weil jede der vier Farben genau einmal vorkommt. Ohne Abstraktion hätte der *Dealer*-Knoten 48 Kanten, eine für jede der 48 *Gemeinschaftskarten*. Auf diesem *Board* aber müssen für den *River* nur 13 verschiedene Äquivalenzklassen unterschieden werden. Jede der Äquivalenzklassen enthält alle Karten derselben Wertigkeit. Damit wird die Komplexität auf $\frac{13}{48}$ reduziert. Im Beispiel des *Boards* Q♥J♠5♣ hingegen können keine *Gemeinschaftskarte*n auf dem *Turn* zusammengefasst werden. Jede der 49 verschiedenen *Turn*-Karten benötigt einen eigenen Teilbaum, da die drei verschiedenen Farben auf dem *Board* alle noch einen *Flush* ermöglichen und die Farbe ♦ mit keinen anderen Farben zusammengefasst werden kann.

5.6 Alternative zum CFR

Durch eigene Überlegungen zum Nash-Equilibrium wurde eine Alternative zum CFR entwickelt. Diese Alternative kommt ohne akkumuliertes Bedauern R^T und ohne Bildung des Durchschnitts S_Σ über alle berechneten Strategien aus. Die eigene Alternative basiert auf drei Ideen zugleich:

1. Es wird ausschließlich die Wahrscheinlichkeit der Kante mit dem größten, erwarteten Nutzen erhört, bei gleichzeitiger Minderung der restlichen Wahrscheinlichkeiten.
2. Außerdem werden neuen Wahrscheinlichkeiten nicht in Relation zum Bedauern R^T gewählt, sondern die Änderung in der n-te Iteration mit Faktor $\frac{1}{n}$ in die bisherige Strategie gemischt.
3. Um zu vermeiden, dass Aktionen, denen einmal eine positive Wahrscheinlichkeiten zugewiesen wurde, zu keinem beliebigen, späteren Zeitpunkt mehr Null zugewiesen werden kann, wird zusätzlich noch eine mit $\frac{1}{n}$ gewichtete Konstante c subtrahiert.

Aufgrund der Selektion der besten Kante, der Mischung mit der bisherigen Strategie und der Verwendung einer Konstante c, wird für diesen Algorithmus künftig die Bezeichnung „Counterfactual Greedy Hedging“ (CGH c=c*) verwendet.

Im ersten Schritt des Algorithmus wird für *Handkarten* h und Knoten k der Kind Knoten k_{i^*} mit dem maximalen μ_c ausgewählt. Sollte es gleichzeitig mehrere Knoten k_i geben, die ein maximales μ_c aufweisen, dann wird der Knoten ausgewählt, zu der die Aktion mit der höchsten Einsatzhöhe führt.

$$k_{i^*}(h) = \arg_{k_i} max\,(\mu_c(h, k_i))$$

Anschließend werden die Wahrscheinlichkeiten für alle anderen, nicht optimalen, zu den k_i führenden Kanten beim n-ten Betreten des Knotens k um Faktor $\frac{1}{n_k}$ verringert. Außerdem wird noch eine mit $\frac{1}{n_k}$ gewichtete und mit i_{max} (Anzahl der Kanten in Knoten k) normalisierte Konstante c abgezogen.

$$s_{A_{neu}}(h, k_i) = \max\left(0, \quad \frac{n_k - 1}{n_k} * s_A(h, k_i) - \frac{1}{n_k} * \frac{c}{i_{max}}\right), \qquad i \neq i^*$$

Im letzten Schritt erhält die zu k_{i^*} führende Kante die zu 1 verbleibende Differenz der Auswahl-Wahrscheinlichkeit. So steigt die Wahrscheinlichkeit in der Strategie der zu k_{i^*} hinführenden Kante insgesamt um den Betrag, um den die anderen Kanten reduziert wurden.

$$s_{A_{neu}}(h, k_{i^*}) = 1 - \sum_{i=1}^{i_{max}} s_{A_{neu}}(h, k_i), \qquad i \neq i^*$$

Dieser Algorithmus hat im Vergleich zum CFR Algorithmus im Hinblick auf den Speicher den Vorteil, dass die neuen Wahrscheinlichkeiten einer Strategie nur aus μ_c und den alten Wahrscheinlichkeiten s_A berechnet werden. Da auf akkumuliertes Bedauern R^T und Bildung des Durchschnitts mithilfe von S_Σ verzichtet werden kann, beträgt der Speicherbedarf für die Strategien im Vergleich zum CFR ein Drittel plus die Speicherung eines Zählers n_k für jeden Knoten. Dieser Zähler wird allerdings nur für den Fall benötigt, dass in einer Iteration nicht alle Knoten betreten werden. Ansonsten gilt $n_k = n$, wofür ein globaler Zähler n für die Anzahl der Iterationen ausreichend ist.

In der Evaluierung in Abschnitt 7.4 zeigt sich außerdem, dass der CGH im Vergleich zum CFR in mit allen getesteten Abstraktionen eine im Vergleich zum CFR höhere Konvergenzgeschwindigkeit zeigt. Dieses Ergebnis bedarf noch weiterer Analysen, bevor es als gesichert angenommen werden kann.

6 Implementierung

In den vorherigen Kapiteln wurden die theoretischen Grundlagen und Konzepte vorgestellt, die für die Berechnung eines ε-Nash-Equilibriums mithilfe des CFR Algorithmus, unter Verwendung von komplexitätsreduzierenden Abstraktionen, von zentraler Bedeutung sind. In Abgrenzung zu anderen Arbeiten wurden neue Abstraktionen für *Handkarten* und *Gemeinschaftskarten* mit dem Ziel einer minimierten Verlustbehaftung eingeführt.

Aufbauend auf den vorgestellten Konzepten wird nun die Softwarearchitektur in Java dargestellt, mit der Strategien, die das Nash-Equilibrium annähern, für NLHE HU Poker berechnet werden.

Als Input erhält die Software einen konkreten *Flop* (drei *Gemeinschaftskarten*), die Größe des *Pots*, den *Stack* jedes Spielers und die *Range* der *Handkarten* beider Spieler. Die *Range* jedes Spielers ergibt sich aus deren *Preflop*-Strategien (siehe Abschnitt 5.1). Dabei gilt, dass je mehr *Chips* von beiden Spielern *Preflop* gesetzt wurden, desto stärker sinkt die Anzahl der *Handkarten* in den *Ranges*, da jeder weitere Einsatz von *Chips* dazu führt, dass der jeweils andere Spieler *Handkarten folded*.

Abbildung 13 stellt für die Software die Berechnung und Validierung eines *Strategieprofil*s für zwei Spieler in sieben Schritten dar. Für einen konkreten *Flop* wird zuerst eine Abstraktion der *Gemeinschaftskarten* unter der Verwendung von Farb-Isomorphismen berechnet.

Darauf aufbauend wird im zweiten Schritt die Abstraktion der *Handkarten* berechnet. Hier kommen die Abstandsmaße und Identitäts-Methoden aus Abschnitt 5.2 zur Anwendung.

Mithilfe beider Abstraktionen wird anschließend ein reduzierter Baum erzeugt. Knoten repräsentieren entweder einen der beiden Spieler oder den *Dealer*. Kanten entsprechen Einsätzen von *Chips* (Spieler) oder dem Aufdecken neuer *Gemeinschaftskarten* (*Dealer*).

Nachdem anschließend für beide Spieler eine Strategie initialisiert wurde, startet der CFR Algorithmus und minimiert mit jeder Iteration den Abstand der Strategien zum Nash-Equilibrium. Da zwei Strategien alternierend optimiert werden, sind es $2 * N$ Schritte.

Nach N Iterationen kann ε mithilfe der optimalen Gegenstrategie bestimmt werden. Am Ende können die Ergebnisse optional mithilfe einer Simulation validiert werden.

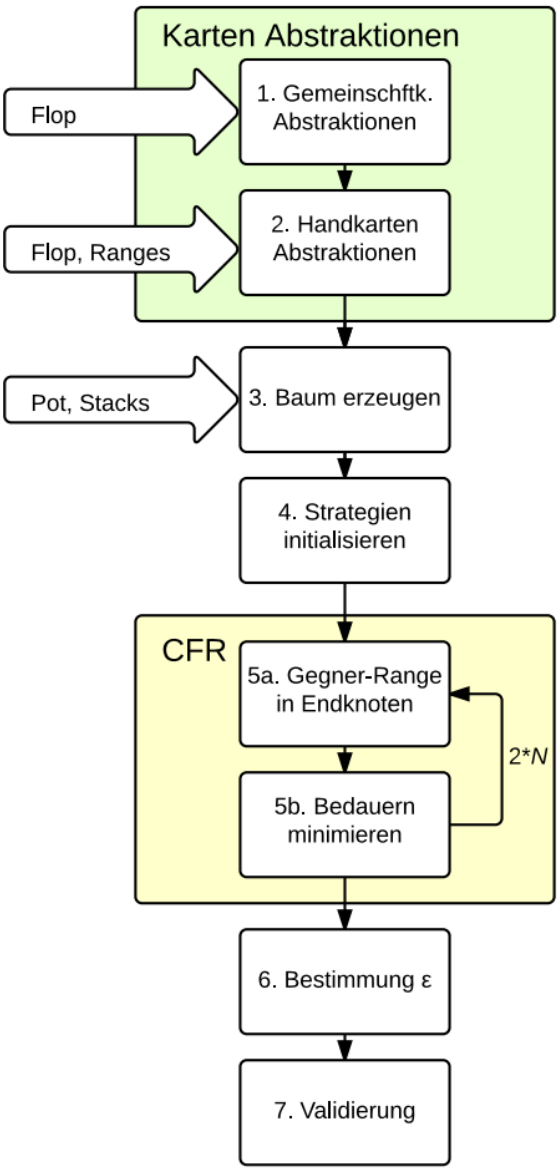


Abbildung 13: Ablauf der Berechnung einer Strategie.

6.1 Karten Abstraktionen

Die zwei zentralen Konzepte jeder Pokervariante sind Karten und *Wettrunden*. Um mit den derzeitigen Restriktionen im Hinblick auf Speicher und Rechenzeit haushalten zu können, werden für beide Konzepte komplexitätsreduzierende Abstraktionen benötigt. Im Folgenden wird zuerst die Realisierung der Karten Abstraktionen in der Software betrachtet.

Die Software muss Karten auf vier Ebenen abbilden können. Einzelne Karten, eine Menge von Karten (z.B. für Hand- oder *Gemeinschaftskarten*), Kartengruppen sind eine Menge von Mengen von Karten und schließlich der Menge von Kartengruppen (entspricht Γ in Abschnitt 5.4.1.1). Kartengruppen werden für Abstraktionen benötigt, bei denen mehrere Hand- und *Gemeinschaftskarten* zur Reduktion der Komplexität zusammengefasst werden. Die Menge von Kartengruppen kann schließlich eine komplette *Handkarten* Abstraktion darstellen, sofern sie alle möglichen *Handkarten* genau einmal enthält.

In Abbildung 14 ist in UML dargestellt, wie die ersten drei Ebenen an Karten in der Software realisiert werden. Eine Kartengruppe entspricht einem Objekt der Klassse *CardSetGroup*, welches aus beliebig vielen Objekten von *CardSet* besteht die ihrerseits wieder aus beliebig vielen Objekten des Typs *Card* bestehen. Sowohl *CardSet* als auch *CardSetGroup* implementieren beide das Interface *Cards* und sind damit eine Menge an Karten und gleichzeitig auch eine einelementige Kartengruppe.

Die Menge von Kartengruppen wird als *List<Cards>* bzw. *List<CardSetGroup>* dargestellt, ohne dass hierfür eine eigene Klasse eingeführt wurde.

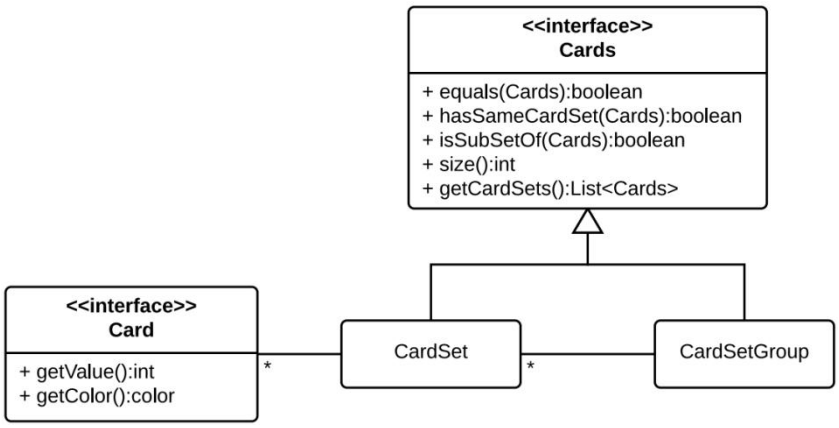


Abbildung 14: UML - Einzelne Karten, Mengen von Karten und Kartengruppen.

Schritt 1: Abstraktionen für *Gemeinschaftskarten* Zur Berechnung der Strategien werden zuerst Abstraktionen über *Gemeinschaftskarten* für einen konkreten *Flop* berechnet. Ein *Flop* entspricht in der Software einem *CardSet* Objekt mit drei *Card* Objekten. In *Gemeinschaftskarten* Abstraktionen werden strategisch äquivalente *Gemeinschaftskarten* jeweils für *Turn* und *River* in Kartengruppen zusammengefasst. Da für diese Zusammenfassung ausschließlich Farb-Isomorphismen verwendet werden, besteht eine Kartengruppe immer nur aus Karten mit derselben Wertigkeit aber unterschiedlichen Farben. Da die Farben der Karten bei Poker nur für den *Pokerrang Flush* (fünf Karten derselben Farbe) eine Bedeutung haben, werden in den *Gemeinschaftskarten* Abstraktionen die Farben zusammengefasst, die keinen *Flush* mehr bilden können. In der Evaluierung zeigt sich, dass diese einfache Abstraktion die Anzahl der Knoten des Baumes bereits um Faktor 2,3 reduziert.

Schritt 2 *Handkarten* **Abstraktionen:** Es folgt die Berechnung der *Handkarten* Abstraktionen, für die diese Arbeit neue Konzepte mit wesentlich geringeren Verlusten im Vergleich zum Stand der Technik vorgestellt hat. Um eine *Handkarten* Abstraktion zu erzeugen, werden zuerst alle auf dem

Flop noch möglichen $\binom{49}{2} = 1176$ *Handkarten* auf Basis eines Abstandsmaßes für jede Kombination an noch möglichen *Gemeinschaftskarten* (insgesamt $49 * 48 = 2352$ für *Turn* und *River*) bewertet. Für jede Kombination aus *Handkarten* und *Gemeinschaftskarten* existiert genau eine Bewertung. In der in Abbildung 15 dargestellten Architektur wird eine Bewertung durch ein *Valuation* Objekt repräsentiert. Die Menge aller Bewertungen ist in einem *HolecardsRanking* Objekt gekapselt.

Als Abstandsmaße können *Pokerrang*, Handstärke und Gegner-Sicht-Vektor verwendet werden (siehe Abschnitt 5.2), wobei es für jedes Abstandsmaß eine eigene Klasse gibt, die *Valuation* implementiert. Um für *Handkarten* mehrere *Valuation* Objekte mit wählbaren Abstandsmaß zu erzeugen, gibt es für jedes Abstandsmaß eine Fabrik *ValuationFactory*, die entsprechende *Valuation* Objekte erzeugt. Ein *HolecardsRanking* wird also mithilfe eines Abstandsmaßes erzeugt und enthält Bewertungen für *Handkarten*. Damit können beliebige *Handkarten* paarweise verglichen werden. Bei einem solchen Vergleich von zwei *Handkarten* kommen die in Abschnitt 5.3 eingeführten Methoden *River*-Identität, hierarchisches Histogramm oder globales Histogramm zur Anwendung. Jede dieser Methode verwendet zur Prüfung der 2352 Bewertungen einen anderen Algorithmus.

Die Klasse *HolecardsAbstraction* kombiniert *HolecardsRanking* mit einer Identitätsmethode (*Identity*). Ihre Methode *getAbstraction(Cards)* erhält als Parameter konkrete *Gemeinschaftskarten*, sowie die Gegner-*Range* und liefert als Ergebnis die Abstraktion, also eine Menge von Kartengruppen.

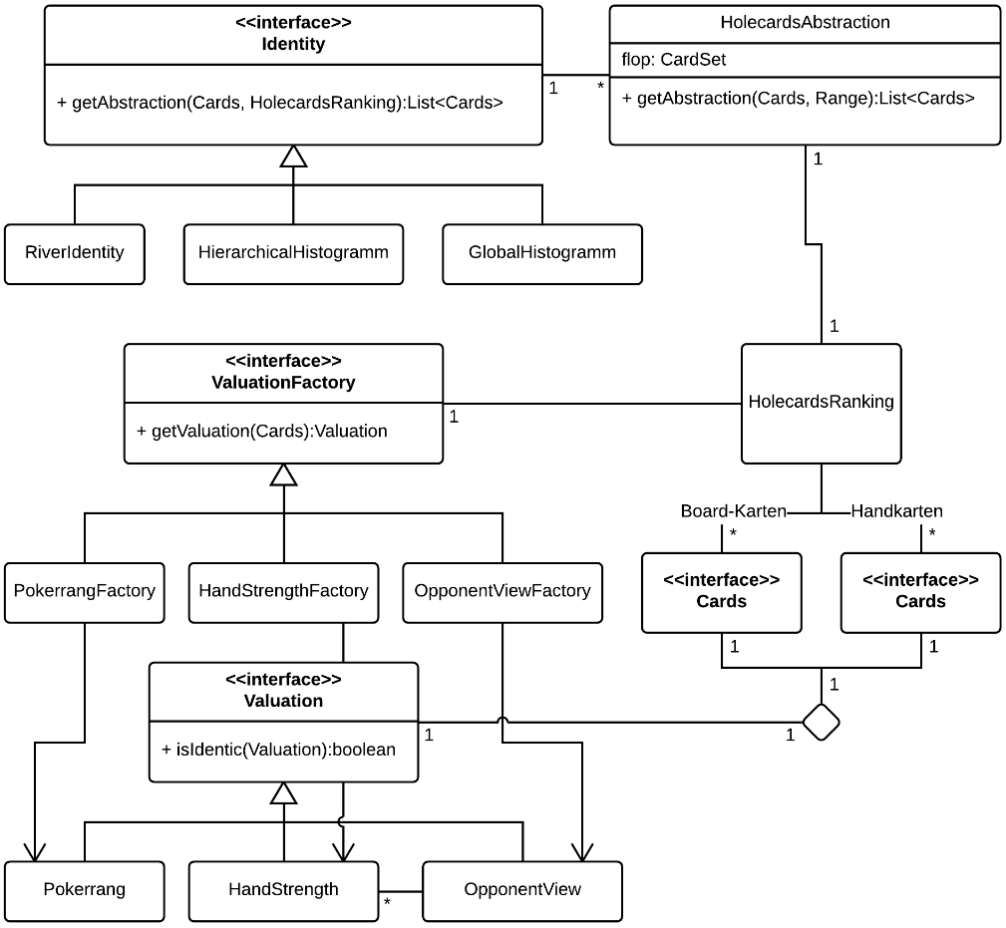


Abbildung 15: UML - *Handkarten* Abstraktion auf Basis der verschiedenen Abstandsmaße und Identitäs-Methoden.

Somit ist die Software in der Lage eine *Handkarten* Abstraktion für jede Kombination aus Abstandsmaß, Identitätsmethode und *Gemeinschaftskarten* zu erzeugen.

6.2 Baum und Strategie

Schritte 3 & 4 Baum erzeugen und Strategien initialisieren: Der erzeugte Baum entspricht dem in Abbildung 4 dargestellten Konzept, bei dem Kanten entweder Einsätzen der Spieler oder Aufdecken neuer *Gemeinschaftskarten* entsprechen. Der Terminus „reduzierter Baum“ wird deshalb verwendet, weil keine Informationen über die *Handkarten* der Spieler in der Struktur des Baumes kodiert sind. Es werden also in jedem Knoten eines Spielers alle *Handkarten* zusammengefasst.

Abbildung 16 zeigt die Umsetzung des Baumes und der Strategien in der Software. Analog zu gängigen Konzepten besteht ein Baum aus einer Anzahl von *Node* Objekten. Ein *Node* enthält Informationen über den aktiven Spieler, kennt die Kante zum Vorgänger (*Parent Edge*) und alle Kanten zu seinen Nachfolgern (*Child Edges*). Die Information über konkrete Aktionen (z.B. Spieler setzt 30 *Chips*) ist in den *Edge* Objekten abgelegt. Dazu hat jedes Edge Objekt ein *Action* Objekt. Die *Action* Objekte in Spieler-Knoten kennen die Einsatzhöhe, die *Action* Objekte für *Dealer*-Knoten enthalten die Information über die konkret aufgedeckten, neuen *Gemeinschaftskarten*.

Hier zeigt sich auch, dass die *Gemeinschaftskarten* Abstraktion direkt die Komplexität des Baumes reduziert. Durch diese Abstraktionen verfügen *Dealer*-Knoten über weniger Kanten, da in manchen Kanten *Gemeinschaftskarten*, die aufgrund der Farben isomorph sind, zusammengefasst werden.

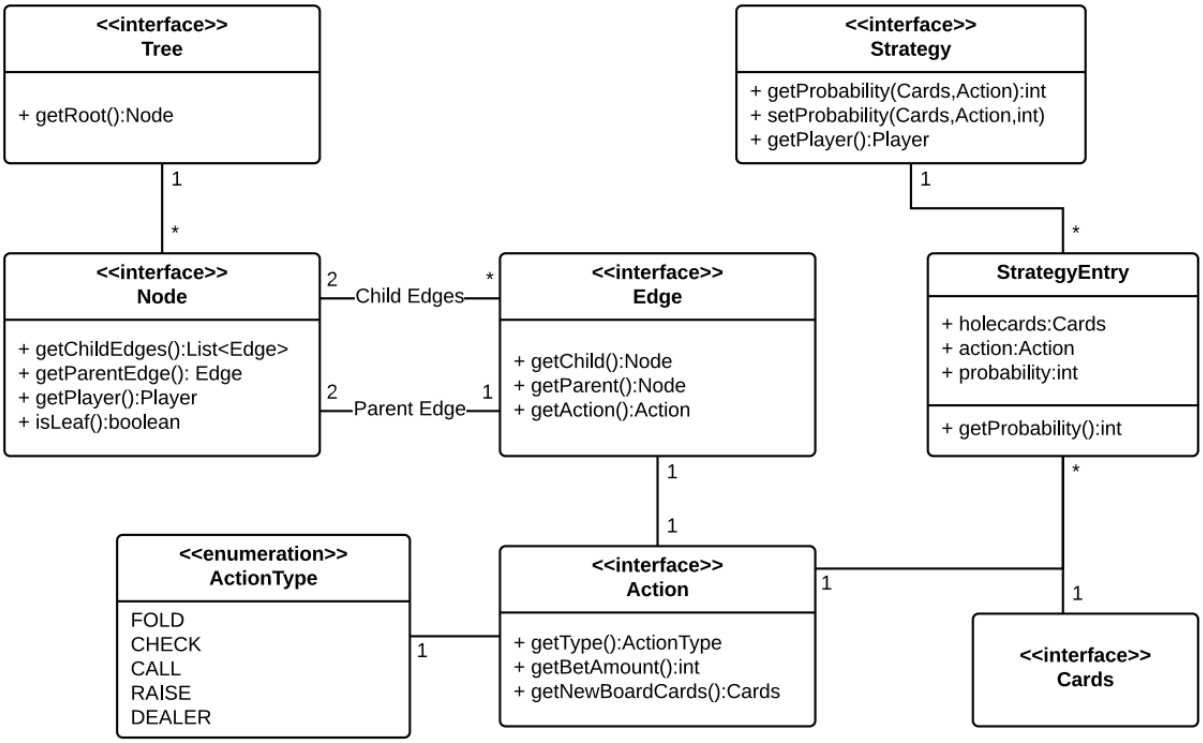


Abbildung 16: UML - Baum und Strategie

In dem Baum sind ebenfalls die *Wettrunden* kodiert. Da die Anzahl der Knoten des Baumes im Verhältnis zur Anzahl der durchschnittlichen Aktionen pro Knoten exponentiell ansteigt, trägt die *Wettrunden* Abstraktion den größten Anteil zur Komplexitätsreduktion bei. In einem Spieler-Knoten gibt es für jeden möglichen Einsatz eine Kante.

Hingegen werden die Informationen über die *Handkarten* bzw. der *Handkartengruppen* außerhalb des Baumes gespeichert. Es gibt für jeden Spieler ein *Strategy* Objekt, welches für jede Kombination aus *Handkarten* und Aktion die Wahrscheinlichkeit enthält, mit der der Spieler diese Aktion wählt. Eine

solche Kombination wird durch *StrategyEntry* dargestellt. Die Anzahl der *StrategyEntry* hängt damit direkt von der Größe der *Handkarten* Abstraktion ab.

6.3 CFR und Bestimmung von ε

Schritt 5 CFR Algorithmus: Der iterative CFR Algorithmus operiert auf den in Abbildung 16 dargestellten Datenstrukturen des Baumes und der Strategien. Es wird der Algorithmus 3 (siehe Abschnitt 4.3) mit *N* Iterationen ausgeführt. Dieser nutzt zuerst Algorithmus 1, um für jedes Blatt die Gegner-*Range* unter Kenntnis der gegnerischen Strategie zu berechnen. Dann wird unter Anwendung von Algorithmus 2 die aktuelle Strategie mithilfe der Minimierung des Bedauerns optimiert. Das Ergebnis mündet in zwei Strategien (*Strategieprofil*) die in Abhängigkeit von der Zahl der Iterationen umso näher am Nash-Equilibrium liegen.

Schritt 6 Berechnung von ε: In Abschnitt 4.1 wurde aufgezeigt, wie ε durch die Berechnung der optimalen Gegenstrategien für ein gegebenes *Strategieprofil* bestimmt werden kann. Der Abstand ε zum Nash-Equilibrium kann sowohl auf den vom CFR genutzten Abstraktionen erfolgen, als auch auf einer komplexeren und damit weniger verlustbehafteten Abstraktion. Der Grund dafür ist der geringere Speicherbedarf. Da es sich bei der optimalen Gegenstrategie immer um eine reine Strategie handelt, fallen die Anforderungen an den Speicherbedarf geringer aus. Ein Bit pro Eintrag in der Strategie ist ausreichend.

Wie in der Analyse gezeigt, konvergiert ε durch den CFR dabei gegen Null. Dies ist allerdings nur innerhalb der für die Berechnung verwendeten Abstraktionen richtig. Bei der Transformation der Strategien auf das Originalspiel tritt der Overfitting-Effekt auf und ε nimmt für das Originalspiel einen deutlich größeren Wert an, als ε innerhalb der Abstraktionen.

6.4 Validierung

Weil die für die Berechnung einer „Nash-Equilibrium nahen Strategie“ für Poker notwendigen Algorithmen und Datenstrukturen komplex sind, ist die Implementierung anfällig für Fehler. Auch, wenn für die Entwicklung an vielen Code-Stellen Validitäts- und Plausibilitätstests durchgeführt werden, sind Fehler in der Implementierung von Formeln bzw. Algorithmen, sowie in den Spielregeln nur schwer zu identifizieren. Die Berechnung der optimalen Gegenstrategie durchläuft im Wesentlichen denselben Code wie es auch der CFR Algorithmus tut und dadurch können Fehler systematisch unerkannt bleiben. Es kann sein, dass ein gegen Null konvergierendes ε auf einem Fehler basiert, der sowohl den CFR als auch die Berechnung der optimalen Gegenstrategie betrifft.

Die erste Überlegung, nämlich die in dieser Arbeit entwickelten Software gegen bekannte Pokersoftware spielen zulassen, gestaltete sich als schwierig. Keines der spielstarken Programme der letzten Jahre ist öffentlich zugänglich. Einige wenige werden zwar als kommerzielle Trainings-Software angeboten, aber nur mit einer Benutzerschnittstelle zum Spiel gegen menschliche Gegner.

Schritt 7 Validierung: Für die Validierung entstand deshalb ein eigenständiges Simulationsmodul. Dieses Modul lässt die berechneten Strategien in einem separaten Teil der Software gegeneinander antreten. Es wird dabei nicht auf die Berechnung von Erwartungswerten zurückgegriffen, sondern auf zufällig erzeugte Spielsituationen mit jeweils ganz konkreten Hand- und *Gemeinschaftskarten*. Anschließend werden die über viele Simulationen ermittelten Durchschnittswerte mit dem berechneten, erwarteten Nutzen sowie mit ε verglichen. Es können Durchschnittswerte für das gesamte Spiel verglichen werden, aber auch Werte für ganz bestimmte Teilbäume bis hin zur Ebene

von einzelnen Knoten. Mithilfe dieser Validierungsmethode konnten viele Fehler in der Software gefunden werden, speziell im Hinblick auf die Berechnung von bedingten Wahrscheinlichkeiten bei der anspruchsvollen Zusammenfassung und Trennung von *Handkartengruppen* in unterschiedlichen *Wettrunden* (siehe Abschnitt 5.4.1).

Da Poker ein Spiel mit sehr hoher Varianz ist, wurde die Laufzeit der Simulationssoftware mittels Vorberechnung des *Pokerrang*s (siehe Abschnitt 6.5), Caching und Parallelisierung optimiert. Die aktuelle Version kann ca. 500.000 Pokerspiele pro Sekunde auf einem Notebook[18] durchführen.

6.5 Laufzeitoptimierungen

Neben der Reduktion der Komplexität wurden eigene Ideen zur Laufzeitoptimierung in allen sieben Schritten eingesetzt. Im Folgenden werden die wichtigsten Optimierungen vorgestellt.

Vorberechnung des *Pokerrang*s Um die Gewinnerhand bei einem *Showdown* zu ermitteln, muss der *Pokerrang* der Hände beider Spieler verglichen werden. Zur Ermittlung des *Pokerrangs* müssen zuerst aus den sieben Karten (*Handkarten* plus *Gemeinschaftskarten*) die für den Spieler günstigsten fünf Karten ermittelt werden. Dafür gibt es $\binom{7}{5} = 21$ verschiedene Möglichkeiten. Für jede der 21 möglichen Hände wird der *Pokerrang* durch Prüfung der 9 verschiedenen *Pokerränge* bestimmt und miteinander verglichen.

Diese aufwendige Prüfung wurde durch einen direkten Zugriff auf ein Array ersetzt. Jeder Index des Arrays entspricht einer Kombination aus sieben Karten. Das dazugehörige Feld des Arrays enthält den *Pokerrang*. Der Array enthält somit $\binom{52}{7} = 1{,}34 * 10^8$ verschiedene Felder. Da es ca. 5.000 unterscheidbare *Pokerränge* gibt, kann die Information eines *Pokerrangs* in zwei Byte kodiert werden. Der so erzeugte Array benötigt dann 268 MB Speicher. Mit dem Array reduziert sich der Aufwand zur Ermittlung eines *Pokerrangs* auf die Berechnung des Index.

Der Index X des Arrays wird dabei mithilfe der Indizes der einzelnen Karten $(k_1, \dots, k_7)$ berechnet. Benötigt wird also eine Funktion $f(k_1, \dots, k_n) \rightarrow X$, wobei kein k_i mehrfach vorkommt (jede Karte existiert nur einmal) und aufsteigend sortiert ist. Durch diese kombinatorischen Bedingungen (keine Wiederholung mit Reihenfolge), kann der Index X mithilfe von Summen über Binomialkoeffizienten berechnet werden. Sei n die Anzahl der abzubildenden Karten (zwei *Handkarten* plus fünf *Gemeinschaftskarten*) und m die Anzahl aller möglichen Karten (52 Karten des Poker Blatts), dann lässt sich X wie folgt berechnen:

$$X = \sum_{i=1}^{n} \left(\sum_{a=k_{i-1}+1}^{k_i} \binom{m-a}{n-i} \right) \quad mit\ k_0 = 0$$

Anschaulich erklärt, wird für jeden möglichen Wert des Index der ersten Karte ein eigener Bereich im Array verwendet. Durch die Summierung der Binomialkoeffizienten wird der Index des Arrays berechnet, mit dem der für den konkreten Index der ersten Karte reservierte Bereich beginnt. Anschließend wird innerhalb des Bereichs dasselbe Verfahren für die zweite Karte angewendet und so ist der Index des Arrays am Ende die Summe über die Ergebnisse für die einzelnen Karten. Der innere

[18] Vier Kerne mit 2,5GHz vom Typ i7

Term der Formel, also die Summe über die Binomialverteilung, lässt sich durch Verwendung eines Caches wesentlich beschleunigen.

Insgesamt konnte durch diese Technik eine Beschleunigung bei der Bestimmung des *Pokerrang*s von mindestens Faktor 640 erreicht werden.

Mathias Retzlaff hat dieses Verfahren durch Einführung eines zweidimensionalen Arrays weiter verbessert. In der ersten Dimension dieses Arrays wird ein Index für *Gemeinschaftskarten* und in der zweiten Dimension ein Index für *Handkarten* verwendet. Für jede der $\binom{52}{5} = 2.598.960$ möglichen Kombination an fünf *Gemeinschaftskarten* und jede Kombination der $\binom{52}{2} = 1326$ *Handkarten* enthält dieses zweidimensionale Array einen *Pokerrang*. Damit ist das zweidimensionale Array zwar aufgrund der Abdeckung invalider Kartenkombinationen (*Handkarten* und *Gemeinschaftskarten* sind nicht disjunkt) mit 6,89 GB deutlich größer, ermöglicht aber aufgrund der eingesparten Summierung von Binomialkoeffizienten eine zusätzliche Beschleunigung um mindestens Faktor 4.

Vorberechnungen für CFR Der CFR-Algorithmus verwendet in einem Blatt k des Baumes zur Berechnung des erwarteten, *counterfactual* Nutzens für die *Handkarten* h_* die in Abschnitt 4.1.3 vorgestellte Formel:

$$\mu_c(h, k_T) = c_k * \ p\left(b_{k_T} \middle| h\right) * \sum_{i=1}^{n} \left(r_{k_T}(h_i) * \ p\left(h_i \middle| \left(b_{k_T} + h\right)\right) * w\left(h, h_i, b_{k_T}\right)\right)$$

Beim Analysieren dieser Formel stellt man fest, dass fast alle Terme in jeder Iteration dieselben Werte annehmen und daher vorberechnet werden können:

- $p(b_k|h)$ ist die bedingte Wahrscheinlichkeit, dass *Gemeinschaftskarten* unter *Handkarten* auftreten.
- $p\left(h_i\middle|(b_k + h)\right)$ entspricht der bedingten Wahrscheinlichkeit, dass gegnerische *Handkarten* unter eigenen *Handkarten* plus *Gemeinschaftskarten* auftreten.
- $w(h, h_i, b_k)$ bestimmt die stärkere Hand auf dem *Board* b_k.

Nur die in $r_k(h_i)$ gespeicherte *Range* des Gegners ändert sich in jeder Iteration durch die Evolution der Strategien.

Mit dieser Vorberechnung konnte die Laufzeit einer CFR-Iteration um Faktor ca. 12 verringert werden. Gleichzeitig stieg der Bedarf an Arbeitsspeicher für das gesamte Programm von 9,4 GB um Faktor 3,2 auf 30,2 GB aufgrund der vorberechneten Werte an.

Multithreading CFR Der zweite Teil des CFR Algorithmus (erwarteten Nutzen berechnen und Strategie verbessern) ist der aufwendigste Teil, da hier die Anzahl der Rechenoperationen quadratisch von der Anzahl der Handgruppen multipliziert mit der Anzahl der Knoten des Baumes abhängt. Deshalb wurde die Laufzeit durch Verwendung von Multithreading optimiert.

Wie an den Formeln in Abschnitt 4.1.3 erkennbar, kann der Schritt „Bedauern minimieren“ für alle *Handkarten* bzw. *Handkartengruppen* voneinander unabhängig durchgeführt werden. Die Änderungen einer Strategie für konkrete *Handkarten* sind innerhalb derselben Iteration unabhängig von den Veränderungen durch andere *Handkarten*. Deshalb können für diesen Schritt maximal so viele

Threads verwendet werden, wie verschiedene *Handkarten* existieren. Die durch dieses Multithreading erreichte Beschleunigung ist aufgrund der Unabhängigkeit linear zur Anzahl der echten Prozessorkerne. Die Anzahl der Threads ist aber nach oben durch die Anzahl der Handgruppen begrenzt.

Arrays statt Objekte Zur Beschleunigung des Datenzugriffs und zur Verringerung des Speicherbedarfs können fast alle Objekte zugunsten von Arrays wegrationalisiert werden. Karten, der Baum und die Strategien lassen sich durch verschachtelte Arrays darstellen. Die Kern-Objekte (wie z.B. *Handkarten* oder *Boards*) haben dann feste Indizes innerhalb jedes dieser Arrays.

Eine erste, teilweise Umsetzung dieser Methode brachte eine Reduktion des Speicherbedarfs um Faktor 2,7 und eine verbesserte Laufzeit um Faktor 2.

7 Evaluierung

An dieser Stelle werden mithilfe der fertigen Software die eigenen Beiträge evaluiert und die Leistung der Implementierung im Hinblick auf die zu Beginn definierte Aufgabenstellung betrachtet.

7.1 Umsetzung der Aufgabenstellung

Die Evaluierung sollte auf Basis einer selbst entwickelten Java-Software geschehen. Die nach der Aufgabenstellung in Abschnitt 1.3 zugelassenen Einschränkungen zur praktischen Umsetzbarkeit wurden folgendermaßen realisiert:

- Als Information für Spieleraktionen in der ersten *Wettrunde* (*Preflop*) erhält die Software eine Menge der möglichen *Handkarten* beider Spieler mit jeweiligen Wahrscheinlichkeiten (*Ranges*), die drei *Gemeinschaftskarten*, die durch den *Flop* sichtbar aufgedeckt wurden und die aktuelle Höhe des Pots, die in den Beispielen dieses Abschnitts 2 *Big Blinds* beträgt.
- Die Geldmenge beider Spieler (sog. *Stack*) ist auf das Fünfzigfache des *Blind*-Einsatzes nach oben beschränkt.
- Die Menge der möglichen Einsätze (sog. *Bet* bzw. *Raise*) ist auf maximal drei verschiedene Einsatzhöhen beschränkt.

Um die Einschränkung der Einsätze auf maximal drei verschiedene Einsatzhöhen umzusetzen, wurde die Abstraktion für die *Wettrunden* so konfiguriert, dass den Spielern die Einsatzhöhen von 0,5 bzw. 1,0 in Abhängigkeit vom *Pot* sowie *All-In* zur Verfügung stehen. Damit kann ein Spieler maximal zwischen fünf verschiedene Aktionen in einem Knoten auswählen, denn neben den Einsätzen können noch die Aktionen *Call* bzw. *Check* und *Fold* verfügbar sein.

Unter Verwendung dieser *Wettrunden* Abstraktionen ergibt sich damit ein reduzierter Baum mit den in Tabelle 2 dargestellten Ausdehnungen.

Tabelle 2: Baum nur unter Verwendung einer Abstraktion über die Setzstruktur.

Dimension	Wert
Knoten gesamt (reduzierter Baum)	3.595.530 (davon 2.259.712 Blätter)
Knoten Spieler	1.333.016 = 2* 666.508
Knoten *Dealer*	2.802
Knoten *Flop*	57
Knoten *Turn*	18.081
Knoten *River*	3.577.392
Informationsmengen	1.567.626.816
Aktionen	4.070.164.224
Spielzustände	4.203.485.577.648

Wie sich zeigt, hat der *River* die mit Abstand größte Anzahl an Knoten. Das liegt zum einen daran, dass Verzweigung des Baumes für den *River* allein schon aufgrund der nach dem *Flop* folgenden *Gemeinschaftskarten* $49 * 48 = 2352$ beträgt. Außerdem befinden sich die meisten Blätter auf dem *River*. Da ein reduzierter Baum verwendet wird, entspricht die Anzahl der Informationsmengen genau der Anzahl der Spieler-Knoten multipliziert mit der Anzahl der noch möglichen *Handkarten* (1176). Die durchschnittliche Anzahl an Aktionen pro Informationsmenge, gleichbedeutend mit dem durchschnittlichen Verzweigungsgrad eines Spieler-Knotens, beträgt 2,6. Diese Zahl liegt deutlich unter den maximal möglichen fünf Aktionen (*Fold*, *Call*, *Bet* 0,5 *Pot*, *Bet* 1,0 *Pot*, *All-In*). Das liegt an

der Verzweigung, bei der ein Spieler die Aktion *All-In* wählt und dem anderen Spieler nur die die zwei Aktionen *Fold* oder *Call* bleiben. Außerdem wird zur Reduktion ab dem 2. *Raise* die Option *Bet* 0,5 zugelassen und im Falle eines 4. *Raise* nur noch *All-In*.

Die Anzahl der Spielzustände der Extensivform entspricht der Anzahl an Knoten multipliziert mit allen Kombinationen an *Handkarten* beider Spieler. Da der CFR in jeder Iteration, für jede Strategie und jeden Spielzustand den erwarteten Nutzen berechnet, ist die Anzahl der Spielzustände multipliziert mit zwei (Strategien) gleichzeitig eine Abschätzung für die Anzahl der Operationen im zweiten Teil (erwarteten Nutzen berechnen und Strategie verbessern) des CFR-Algorithmus. Der erste Teil des CFR (*Range* in Blättern bestimmen) operiert ausschließlich auf der Strategie und damit lässt sich die Anzahl der Operationen mit der Anzahl der Aktionen abschätzen. Beim Vergleich beider Zahlen für Spielzustände ($\sim 10^{12}$) und Aktionen ($\sim 10^{9}$) lässt sich gut erkennen, dass der zweite Teil des CFR der wesentlich aufwendigere ist.

7.2 Komplexitätsreduktion durch Abstraktionen

Als nächstes wird die Komplexität des Baumes durch Verwendung einer *Gemeinschaftskarten* Abstraktion reduziert. Dazu werden die in Abschnitt 5.5 vorgestellten äquivalenten *Gemeinschaftskarten* genutzt. Diese Abstraktion ist aufgrund der ausschließlichen Nutzung von Farb-Isomorphismen eine verlustfreie Abstraktion.

Wie bereits diskutiert, ist die Größe der *Gemeinschaftskarten* Abstraktion von den konkreten *Gemeinschaftskarten* auf dem *Flop* abhängig. Deshalb sind die dargestellten Ergebnisse ab diesem Punkt Durchschnittswerte über ein Sample von acht verschiedenen, im Hinblick auf die Farb-Isomorphismen weitestgehend repräsentativen, *Flops*. Da mit diesem Sample an *Flops* verschiedene, teilweise sehr aufwendige Experimente durchgeführt wurden, konnte es aufgrund der beschränkten Ressourcen nicht größer gewählt werden.

Unter Verwendung der *Board* Abstraktion entsteht ein Baum mit den in Tabelle 3 dargestellten Ausmaßen.

Tabelle 3: Baum nur unter Verwendung einer Abstraktion über die Setzstruktur, Reduktionen im Vgl. zu Tabelle 2.

Dimension	Ø Wert	Ø Reduktion
Knoten gesamt	1.544.871 (davon 969.843 Blätter)	2,3
Knoten Spieler	572.650 = 2* 286.325	2,3
Knoten *Dealer*	2.379	1,2
Knoten *Flop*	57	1
Knoten *Turn*	15.340	1,2
Knoten *River*	1.529.474	2,3
Informationsmengen	673.436.064	2,3
Aktionen	1.748.921.664	2,3
Spielzustände	1.806.880.867.106	2,3

Da die Laufzeit des CFR von den Spielzuständen abhängt, ergibt sich eine Beschleunigung von Faktor 2,3. Auch der Speicherbedarf sinkt um denselben Faktor. Die geringere Reduktion der Knoten für den *Turn* ist logisch, denn durch die noch kommende letzte *Gemeinschaftskarte* ist es möglich, dass zwei verschiedene Farben zur Bildung eines *Flush* in Frage kommen. Auf dem *River* hingegen kann dies höchstens noch eine Farbe sein.

Handkarten Abstraktionen

Bis hierhin wurden die $\binom{49}{2} = 1176$ möglichen *Handkarten* jeweils einzeln betrachtet. Als nächstes werden die in dieser Arbeit größtenteils neu entwickelten *Handkarten* Abstraktionen verwendet und analysiert, wie sich diese auf die verschiedenen Komplexitätsmaße auswirken.

In 5.4 wurden Formeln vorgestellt, mit dem der CFR beliebige Veränderungen der *Handkartengruppen* nach einer neuen *Gemeinschaftskarte* verarbeiten kann. Damit ist es möglich, für jede Kante eines *Dealer*-Knotens eine eigene *Handkarten* Abstraktion zu verwenden. Das ist deshalb wertvoll, weil durch die weitere *Gemeinschaftskarte* die neue *Handkarten* Abstraktion eine geringere Komplexität besitzt. Im Gegensatz dazu nutzen die derzeit gängigen Verfahren auf dem *Flop* entweder eine große Abstraktion, die mit jeder *Gemeinschaftskarte* weiter aufgespalten wird oder verwenden mit Imperfect *Call* eine Methode, die frühere Zustände überhaupt nicht berücksichtigt.

In Tabelle 4 ist die Anzahl der *Handkartengruppen* für jede Kombination aus Abstandsmaße und Identitätsmethode auf dem *Flop*, *Turn* und *River* dargestellt. Da es sich um Durchschnittswerte handelt, sind zusätzlich noch die Maxima und Minima angegeben. Die *Handkartengruppen* wurden auf der oben vorgestellten *Gemeinschaftskarten* Abstraktion gebildet. Ohne *Gemeinschaftskarten* Abstraktion würde die Anzahl der *Handkartengruppen* geringfügig sinken, denn durch die zusammengefassten *Gemeinschaftskarten*, ist der Abstand bestimmter *Handkarten* nicht mehr identisch.

An den Max.- und Min.-Werten ist abzulesen, dass die tatsächliche Anzahl der *Handkartengruppen* in Abhängigkeit der konkreten *Gemeinschaftskarten* stark schwanken. Zum Beispiel ist auf dem *Board* A♥A♠A♣A♦2♥ nur noch die Karte aus den *Handkarten* mit der höchsten Wertigkeit relevant. In diesem Beispiel existieren dann nur noch 12 verschiedene *Handkartengruppen*. Hingegen sind auf dem *Board* 9♣7♣9♥4♦6♣ alle Pokerränge von *Paar* bis *Straight Flush* noch möglich und daher gibt es hier bis zu 288 verschiedene *Handkartengruppen*.

Tabelle 4: Anzahl der *Handkarten*gruppen nach Abstandsmaß und Identitätsmethode.

		River Identität			Hierarchisches Histogramm			Globales Histogramm		
		Ø Gruppen	Min	Max	Ø Gruppen	Min	Max	Ø Gruppen	Min	Max
Pokerrang	*Flop*	327	289	355	483	289	631	325	289	355
	Turn	172	81	266	172	81	266	172	81	266
	River	87	5	149	87	5	149	87	5	149
Handstärke	*Flop*	871	642	1176	868	643	1175	862	642	1160
	Turn	324	86	608	306	63	605	306	63	605
	River	114	7	249	114	7	249	114	7	249
Gegner-Sicht	*Flop*	871	643	1176	* identisch zu Zeile 3, Spalte 1			* identisch zu Zeile 3, Spalte 1		
	Turn	328	91	608						
	River	191	10	288						

Bei der aufsteigenden Konstruktion der Abstandsmaße *Pokerrang*, Handstärke und Gegner-Sicht wurde das Ziel verfolgt mit jedem weiteren Abstandsmaß eine bestimmte Klasse an Verlusten zu vermeiden. Dieses Vorgehen lässt sich gut an der strikt steigenden Anzahl der Gruppen jedes Abstandsmaßes nachvollziehen. Zum Beispiel beträgt die Anzahl der *Handkartengruppen* auf dem *River* für den *Pokerrang* 87, für die Handstärke 114 und für die Gegner-Sicht 191.

Die Wahl der Identitätsmethode hat einen vergleichsweise geringen Einfluss auf die Anzahl der *Handkartengruppen*. Für das Abstandsmaß Gegner-Sicht führen alle Abstandsmaßen zum gleichen Ergebnis, bei *Pokerrang* und Handstärke unterscheiden sich die Ergebnisse nur für *Flop* und *Turn*. Der größte Unterschied der Methoden tritt im Vergleich vom Hierarchischen Histogramm für den *Flop* mit 483 *Handkartengruppen* zum globalen Histogramm mit 325 *Handkartengruppen* für *Pokerrang* auf. Dieses Ergebnis ist interessant, weil das hierarchische Histogramm im Gegensatz zum globalen Histogramm die Entwicklung der Handstärke auf dem *Turn* berücksichtigt. Es zeigt sich, dass die Berücksichtigung der Entwicklung auf dem *Turn* im Hinblick auf den *Pokerrang* einen wichtigen Unterschied ausmacht.

Insgesamt zeigen die Zahlen, dass selbst die komplexesten Abstraktionen einen großen Beitrag zur Reduzierung der Laufzeit des CFR leisten werden. Denn die Laufzeit des CFR hängt von der Anzahl der Spielzustände ab, und diese wiederrum ist quadratisch von der Anzahl der *Handkartengruppen* abhängig. Glücklicherweise befinden sich ca. 99% der Spielzustände auf dem *River* und gerade dort ist die Reduktion durch die *Handkarten* Abstraktionen am deutlichsten.

Bei den Angaben muss noch berücksichtigt werden, dass die *Preflop Range* der *Handkarten* für beide Spieler mit 100% angenommen wurde. In einem realistischen Fall mit einer geringeren *Range* (z.B. 30%) wird die Anzahl der *Handkartengruppen* noch stärker reduziert.

Die genauen Auswirkungen der verschiedenen *Handkarten* Abstraktionen auf die Anzahl der Informationsmengen, Aktionen und Spielzustände werden in Tabelle 5 dargestellt. Die Anzahl der Spielzustände, von der ja die Laufzeit des CFR abhängt, ist im Vergleich zu Tabelle 2 ohne Abstraktionen von 4,2 Billionen auf bis zu 12,1 Milliarden gesunken ist, was insgesamt einem Faktor von 347 entspricht. Diese Reduktion geht zum größten Teil auf die *Handkarten* Abstraktionen zurück.

Auch der Speicherbedarf für eine Strategie, welcher von der Anzahl der Aktionen abhängt, sinkt von 4 Milliarden (16 Gigabyte) auf bis zu 133 Millionen (532 Megabyte) um insgesamt Faktor 30.

Selbst mit der größten, nahezu verlustfreien Abstraktion auf Basis der Gegner-Sicht bewegen sich die Größen immer noch in Bereichen, die ein gewöhnlicher PC bewältigen kann. Andererseits zeigt sich auch, dass eine Berechnung ohne die *Handkarten* Abstraktionen nur auf einem Supercomputer denkbar wäre.

Tabelle 5: Informationsmengen, Aktionen und Spielzustände mit *Handkarten* Abstraktionen. Reduktion im Vergleich zu Tabelle 2.

		River Identität		Hierarchisches Histogramm		Globales Histogramm	
		Ø Anzahl	Redu.	Ø Anzahl	Redu.	Ø Anzahl	Redu.
Pokerrang	Inform.	51.507.024	30,4	51.512.577	30,4	51.503.587	30,4
	Aktionen	133.764.370	30,4	133.778.789	30,4	133.755.443	30,4
	Spielz.	12.113.522.212	347,0	12.119.576.049	347,0	12.112.312.360	347,0
Handstärke	Inform.	68.672.546	22,8	68.405.359	22,8	68.405.042	22,8
	Aktionen	178.343.440	22,8	177.649.551	22,8	177.648.726	22,8
	Spielz.	21.440.071.118	196,1	21.271.632.264	196,1	21.271.082.730	196,1
Gegner-Sicht	Inform.	111.709.726	14,0	* identisch zu Zeile 3, Spalte 1		* identisch zu Zeile 3, Spalte 1	
	Aktionen	290.111.520	14,0				
	Spielz.	57.220.699.314	73,5				

7.3 Qualität der Strategien

Im letzten Abschnitt hat sich gezeigt, dass die in dieser Arbeit eingeführten Abstraktionen trotz der geringen Verlustbehaftung die Komplexität um Faktor 73,5 (Gegner-Sicht) bis zu Faktor 347 (Pokerrang) reduzieren. Dieser Abschnitt untersucht nun die Qualität der mit diesen Abstraktionen erzeugten Strategien.

Hierzu wird mithilfe des CGH (siehe Abschnitt 5.6) ein Strategieprofil (enthält für jeden Spieler eine Strategie) unter Verwendung einer konkreten Abstraktion erzeugt. Anschließend bestimmt man das ε des Strategieprofils sowohl innerhalb der Abstraktion als auch auf dem ursprünglichen (echten) Spiel. Für die Qualität entscheidend ist letztlich der Wert von ε auf dem ursprünglichen Spiel. Dieser entspricht der maximalen Exploitierbarkeit des Strategieprofils.

Da die Berechnung von ε auf dem ursprünglichen Spiel ohne Abstraktionen sehr aufwendig[19] ist, wurde zur Messung ein im Vergleich zu Tabelle 2 kleinerer Baum[20] verwendet.

Tabelle 6 zeigt die Werte für ε nach 200 Iterationen in Abhängigkeit der verwendeten Abstraktion. Zusätzlich wird noch der Abstand zwischen beiden Werten für ε dargestellt. Dieser Abstand entspricht dem Fehler, der durch Verwendung der Abstraktion entsteht.

Insgesamt lassen sich die Zahlen so interpretieren, dass durch die Verwendung der Identität statt der Ähnlichkeit zur Erzeugung von Abstraktionen, der Fehler durch die Abstraktionen gering (vergleiche Overfitting in Abschnitt 4.4.1.2) ist.

Es zeigt sich zudem, dass das in dieser Arbeit eingeführte Gegner-Sicht Abstandsmaß sowohl am besten das Nash-Equilibrium approximiert, als auch den geringsten Abstand zwischen ε in der Abstraktion und im echten Spiel aufweist. Auch gegenüber dem Abstandsmaß des hierarchischen Histogramms, welches in den neuesten Pokerprogrammen Verwendung findet, wird das Nash-Equilibrium durch das auf Gegner-Sicht Abstraktion berechneten Strategieprofils um ca. Faktor 2 besser approximiert.

Weiterhin zeigt sich, dass die Konvergenzgeschwindigkeit von ε innerhalb der Abstraktion für die verschiedenen Abstraktionen sich unterscheidet. Wiederrum konvergiert ε bei Verwendung der Gegner-Sicht am schnellsten. Zu erwarten wäre eigentlich, dass sich die Konvergenzgeschwindigkeiten kaum unterscheiden. Der Unterschied geht aber auf den in Abschnitt 5.4.1.2 dargestellten Fehler zurück, der bei Übertragung des erwarteten Nutzens von größeren auf kleinere *Handkarten* Gruppen entstehen kann.

[19] Ungefähr eine um Faktor 100 längere Laufzeit im Vergleich zur Abstraktion
[20] Mit insgesamt 158.103.526 Spielzuständen und einer *Stack* Höhe von 5 *Big Blinds*

Tabelle 6: Darstellung von ε in Tausendstel *Big Blinds* pro Spiel nach 200 CGH (c=0,175) Iterationen 1) innerhalb der Abstraktion, 2) im echten Spiel (keine *Handkarten* Abstraktion) und 3) Abstand beider Werte. Alle Werte werden jeweils mit denen der Gegner-Sicht verglichen.

		River Identität		Hierar. Histogr.		Glob. Histogr.	
		mbb	Vgl.	mbb	Vgl.	mbb	Vgl.
Pokerrang	ε Abstraktion	5,47	261%	4,68	223%	5,47	261%
	ε echtes Spiel	9,07	377%	8,28	344%	9,07	377%
	Abstand	3,59	1163%	3,60	1166%	3,59	1163%
Handstärke	ε Abstraktion	3,09	147%	4,27	203%	3,49	166%
	ε echtes Spiel	4,08	170%	4,96	206%	4,35	181%
	Abstand	0,99	320%	0,69	224%	0,86	278%
Gegner-Sicht	ε Abstraktion	2,10	100%	* identisch zur 1. Spalte		* identisch zur 1. Spalte	
	ε echtes Spiel	2,41	100%				
	Abstand	0,31	100%				

Tabelle 7 führt die Berechnung aus Tabelle 6 fort und zeigt den Zustand nach weiteren 200 Iterationen. Für alle Abstraktionen ist ε auch im echten Spiel weiter gesunken, der Overfitting-Effekt ist also noch nicht aufgetreten. Allerdings ist bei allen Abstraktionen auch der Abstand der beiden ε gestiegen. Der Abstand ist zugleich eine untere Schranke für ε im echten Spiel, denn für den Fall das ε in der Abstraktion Null beträgt, entspricht der Abstand genau dem ε im echten Spiel.

Die Gegner-Sicht Abstraktion zeigt die größte, relative Reduzierung von ε im echten Spiel. Außerdem zeigt sich, dass ε selbst für zweitbeste Abstraktion (Handstärke mit River Identität) mit 2,84 mbb/g noch über dem Wert der Gegner-Sicht Abstraktion nach den ersten 200 Iterationen liegt.

Tabelle 7: Fortschreibung der Werte aus Tabelle 6 nach insgesamt 400 Iterationen.

		River Identität		Hierar. Histogr.		Glob. Histogr.	
		mbb	Vgl.	mbb	Vgl.	mbb	Vgl.
Pokerrang	ε Abstraktion	3,34	456%	2,77	378%	3,34	456%
	ε echtes Spiel	7,76	636%	7,19	590%	7,76	636%
	Abstand	4,42	908%	4,42	910%	4,42	908%
Handstärke	ε Abstraktion	1,51	206%	2,16	295%	1,99	272%
	ε echtes Spiel	2,84	233%	3,30	270%	3,15	258%
	Abstand	1,33	273%	1,13	233%	1,15	237%
Gegner-Sicht	ε Abstraktion	0,73	100%	* identisch zur 1. Spalte		* identisch zur 1. Spalte	
	ε echtes Spiel	1,22	100%				
	Abstand	0,49	100%				

Abbildung 17 schließt an das Beispiel aus Tabelle 6 und Tabelle 7 an und setzt die Berechnung der Strategieprofile für die Abstraktionen Handstärke (mit Hierar. Histogr.) und Gegner-Sicht bis zur 2000. Iteration fort. Alle 50 Iterationen wird ε innerhalb der Abstraktion wie auch für das echte Spiel gemessen.

Es zeigt sich, dass ε für beide Abstraktionen sinkt, obwohl ε im echten Spiel ab der ca. 1000. Iteration bei 2,75 (Handstärke) bzw. 0,75 (Gegner-Sicht) stagniert. Hier zeigen sich der Overfitting-Effekt und damit gleichzeitig die maximale Qualität der Strategien, die auf der jeweiligen Abstraktion möglich ist.

Interessant ist außerdem noch, dass die Kurven nicht monoton fallen, sondern es Ausreißer nach oben gibt. Der größte Anstieg ist bei der Gegner-Sicht im Bereich der 1000. Iteration zu beobachten. Dieser Effekt, der nur beim CGH und nicht beim CFR auftritt, wird im nächsten Abschnitt noch genauer untersucht.

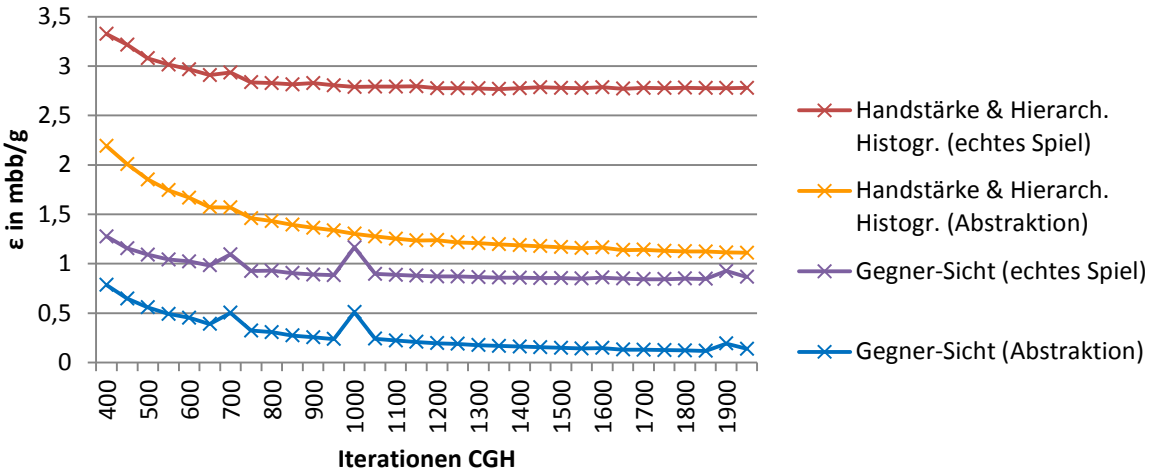


Abbildung 17: Verlauf von ε (gemessen alle 50 Iterationen) ab der 400. Iteration für Handstärke und Gegner-Sicht im echten Spiel und in der Abstraktion

7.4 Konvergenz

In Abschnitt 4.3 wurde der 2007 entdeckte CFR Algorithmus dargestellt, welcher seitdem in allen Poker-Programmen verwendet wurde, die bei der Weltmeisterschaft der Poker-Programme (siehe Abschnitt 3.1) die ersten drei Plätze belegten.

In dieser Arbeit wurde in Abschnitt 5.6 mit dem CGH Algorithmus eine Alternative zum CFR eingeführt. Der CGH benötigt im Vergleich zum CFR zur Optimierung von Strategien nur ca. $\frac{1}{3}$ des Speicherbedarfs, da weder das akkumulierte Bedauern noch die Summe über alle bisher berechneten Strategien gebildet werden muss. In Abbildung 18 wird die Konvergenz von ε von 400 Iterationen von jeweils CFR mit dem CGH auf einer logarithmischen Skala verglichen.

Das dafür verwendete Spiel hat die für den Flop Q♥J♥2♣ in Abschnitt 7.1 dargestellten Ausmaße. Zur zusätzlichen Reduktion wurden die möglichen *Handkarten* beider Spieler auf eine *Range* von 20%[21] eingeschränkt und die Aktion *Bet* 0,5 entfernt. Als Abstraktion diente die Gegner-Sicht, für die im letzten Abschnitt gezeigt wurde, dass die darauf erzeugten Strategieprofile die höchste Qualität aufweisen.

[21] Es wurden in der *Preflop Wettrunde* die 20% stärksten *Handkarten* ausgewählt: A9+, KT+, QT+, JT+, A4s+, K8s+, Q9s+, J9s+, T9s+, 55+

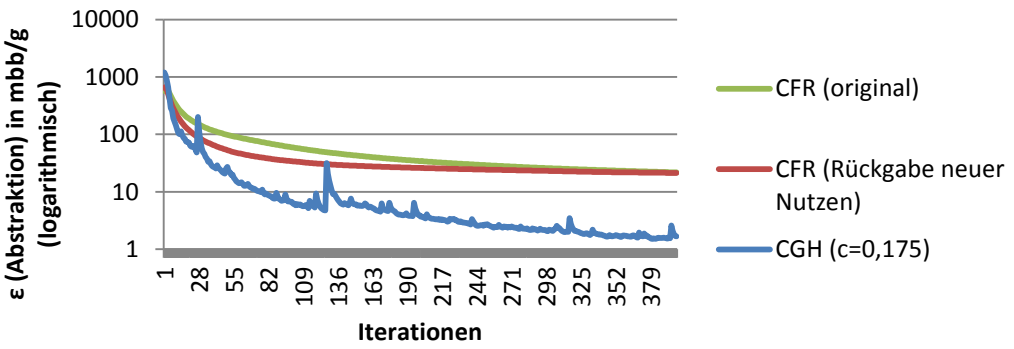


Abbildung 18: Konvergenz von ε (gemessen in jeder Iteration) durch CFR, CFR mit Rückgabe sowie durch den CGH.

Unter diesen Bedingungen zeigt der CGH eine wesentlich höhere Konvergenzgeschwindigkeit, als der CFR. Nach 400 Iterationen hat die durch den CGH berechnete Strategie ein ε von 1,7 erreicht, die des CFR nur ein ε von 21,8. Damit hat der CFR eine durchschnittliche Konvergenzgeschwindigkeit von 2,64% pro Iteration und der CGH eine signifikant höhere, durchschnittliche Konvergenzgeschwindigkeit von 4,28% pro Iteration.

Ein weiterer Vorteil neben der höheren Konvergenzgeschwindigkeit sowie der Speicherersparnis ist die Laufzeit. Durch die einfachere Formel benötigten die 400 Iterationen des CGH mit insgesamt 3984 Sekunden (nur die Zeit zur Strategieoptimierung und nicht zur Bestimmung von ε) eine um ca. 30% geringere Laufzeit im Vergleich zu 5814 Sekunden des CFR.

Das Ergebnis der höheren Konvergenzgeschwindigkeit ist überraschend, da es sich beim CFR um den bisher effizientesten Algorithmus zur Approximation des Nash-Equilibriums für Null-Summenspiele handelt. Deshalb können diese Ergebnisse nur unter dem Vorbehalt einer weiteren, wissenschaftlichen Evaluierung für richtig erachtet werden.

Es ist denkbar, dass trotz größter Sorgfalt bei Implementierung, Test und Kontrolle durch Dritte die schlechtere Konvergenzgeschwindigkeit des CFR auf Programmierfehler zurückzuführen ist. Dagegen spricht allerdings, dass die Eigenschaft des CFR eines streng monoton fallenden ε erfüllt ist.

Außerdem ist es möglich, dass der CGH nur in ganz bestimmten Fällen besser ist, als der CFR. Die in dieser Arbeit verwendeten Abstraktionen beruhen auf der Identität des Abstands und können deshalb nicht mit den wesentlich stärker verlustbehafteten Imperfect Recall Abstraktionen anderer Programme verglichen werden.

Es stellt sich außerdem die Frage, ob der CGH zusammen mit den neuesten Sampling Techniken ebenfalls so stark konvergiert. Alle aktuellen Programme nutzen Sampling Techniken und konnten damit pro Zeit wesentlich höhere Konvergenzgeschwindigkeit nachweisen, als ohne Sampling.

In Abbildung 18 wird außerdem die Konvergenz des CFR in der Originalversion mit der in Abschnitt 4.3 dargestellten Idee (Rückgabe neuer Nutzen) verglichen, bei der in jedem Knoten der Nutzen auf Basis der soeben geänderten Strategie für diesen Knoten zurückgegeben wurde. Obwohl der CFR in dieser Variante in den ersten ca. 60 Iterationen eine höhere Konvergenzgeschwindigkeit aufweist, konvergiert der CFR in der Originalversion ab diesem Punkt besser.

Am Graph des CGH Abbildung 18 zeigt sich außerdem, dass ε in einigen Iterationen ansteigt. Es stellt sich die Frage, ob der Algorithmus wohldefiniert ist und ε in allen denkbaren Spielen gegen Null

konvergiert. Im Vergleich zum CFR sind die Änderungen von ε beim CGH stärker ausgeprägt. Da die meisten dieser Änderungen ε nach unten verändern, werden die Ausschläge nach oben überkompensiert.

Der Einfluss der Konstante c des CGH wird in Abbildung 19 untersucht. Interessant ist hier die blaue Kurve, die den Verlauf von ε für $c = 0$ zeigt. Bis auf eine Ausnahme in der 130. Iteration verläuft diese Kurve wie auch die Kurve des CFR streng monoton fallend, aber deutlich unterhalb der Kurve des CFR. Die stärkste Konvergenz tritt allerdings bei $c = 0{,}2$ auf. Nach 150 Iterationen nimmt der CGH für $c = 0$ das höchste ε an und ist damit schlechter als alle Varianten, die ein positives c verwenden. Das deutet darauf hin, dass die Verwendung der Konstanten eine Verbesserung des Verfahrens darstellt.

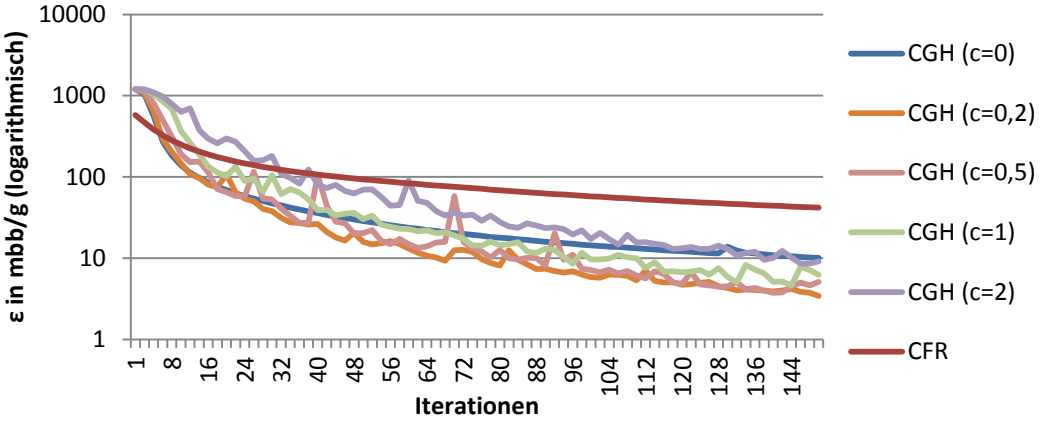


Abbildung 19: Konvergenz von ε (gemessen jede 2. Iteration) mit dem CGF in Abhängigkeit der Konstante c.

Zum Abschluss der Evaluierung werden in Abbildung 20 der CGH und CFR auf einem im Vergleich zu Abbildung 17 um ca. Faktor drei komplexeren Baum sowie einem anderen Flop (T♥6♠3♣) in den ersten 1.000 Iterationen verglichen. Auch hier ist die höhere Konvergenzrate des CGH gut erkennbar. Anschließend werden die erzeugten Strategieprofile innerhalb einer von der sonstigen Berechnung unabhängigen Simulation echter Spiele (siehe Abschnitt 6.4) gegeneinander getestet.

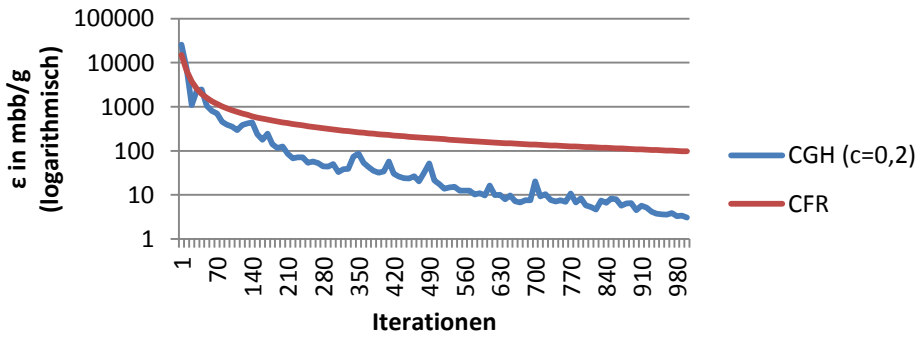


Abbildung 20: Vergleich von ε (gemessen alle 10 Iterationen) mit CFR und CGH über 1000 Iterationen auf dem Flop T♥6♠3♣.

In der 2. Spalte in Tabelle 8 ist der erwartete Nutzen für alle vier Strategien (jeweils *IN* und *OUT* eines Strategieprofils) gegen die jeweils optimale Gegenstrategie dargestellt und ist damit gleichzeitig eine garantierte, untere Schranke für den Nutzen. Die Summe von *IN* und *OUT* entspricht ε. In der letzten Spalte ist der tatsächlich realisierte Nutzen in 10^9 simulierten Spielen dargestellt. Diese Ergebnisse

wurden mit der in Abschnitt 6.4 vorgestellten Simulationskomponente gemessen. Die vom CGH berechnete Strategie für die Position *IN* gewann pro Spiel im Durchschnitt ca. 38 mbb (Tausendstel Big Blinds) gegen die mit dem CFR berechnete *OUT* Strategie. Maximal möglich wären allerdings ca. 70 mbb pro Spiel gewesen. Hier zeigt sich, dass das Nash-Equilibrium andere Strategien nur eingeschränkt exploitiert.

Tabelle 8: Fortführung von Abbildung 20, Darstellung des erwarteten Nutzens beider Strategieprofile (berechnet mit CGH bzw. CFR) nach 1000 Iterationen.

	Erwarteter Nutzen gegen optimale Gegenstrategie (mbb/g)	Ergebnisse nach Simulation von jeweils 10^9 Spielen gegen CFR bzw. CGH Strategie (mbb/g)
CGH (OUT)	-21,25	-0,50 ± 0,35
CGH (IN)	18,16	37,83 ± 0,35
CGH (OUT+IN)	-3,09	37,33 ± 0,24
CFR (OUT)	-70,23	-37,83 ± 0,35
CFR (IN)	-27,26	0,50 ± 0,35
CFR (OUT+IN)	-97,48	-37,33 ± 0,24

Abbildung 21 zeigt den realisierten Gewinn innerhalb der ersten 150.000 Spiele. Das mit dem CGH erzeugte Strategieprofil liegt am Ende mit 4741 *Big Blinds* (Gewinn CGH Summe) in Führung. Klar zu erkennen ist der strategische Vorteil eines Spielers in *Position* (*IN*). Die Strategie des CGH für den Spieler außerhalb der Position (*OUT*) verliert gegen die Strategie des CFR in *Position* (*IN*).

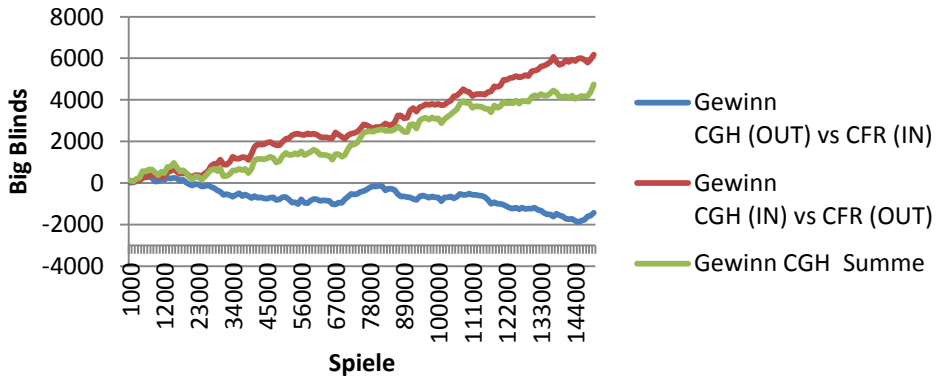


Abbildung 21: Simulation von 150.000 Spielen beider Strategieprofile gegeneinander, Darstellung des Gewinns.

Um einen Blick darauf zu werfen, wie eine ans Nash-Equilibrium approximierte Strategie im Detail aussieht, befindet sich im Anhang dieser Arbeit der Auszug einer, mit dem CGH optimierten, Strategie am Beispiel einer konkreten Situation unter Angabe der Wahrscheinlichkeiten für die dargestellten Aktionen.

8 Zusammenfassung

In diesem Kapitel werden die wichtigsten Beiträge dieser Arbeit diskutiert, ein Ausblick in die Zukunft gegeben, ein kurzer Überblick über die Erkenntnisse von NLHE HU Strategien gegeben und die Arbeit zusammengefasst.

8.1 Diskussion wichtigster Beiträge

Beitrag: Alternative zum CFR

Der im Jahr 2007 eingeführte, iterative CFR Algorithmus [12] optimiert eine Strategie durch das Minimieren des Bedauerns für jede einzelne Informationsmenge. Er ist der derzeit beste, bekannte Algorithmus, um ein Nash-Equilibrium in Null-Summenspielen zu approximieren.

In dieser Arbeit wurde mit dem CGH (Counterfactual Greedy Hedging, siehe Abschnitt 5.6) eine Alternative zum CFR Algorithmus vorgestellt, die Strategieprofile im Vergleich zum CFR in allen Experimenten (siehe Abschnitt 7.4) mit einer wesentlich höheren Konvergenzgeschwindigkeit an das Nash-Equilibrium approximiert hat. Im Gegensatz zum CFR ist ε beim CGH in Anzahl der Iterationen nicht streng monoton fallend.

Neben der höheren Konvergenzgeschwindigkeit pro Iteration benötigt der CGH im Vergleich zum CFR nur ein Drittel des Speichers. Zusätzlich zeigte sich in der Evaluierung, dass die Laufzeit einer Iteration des CGH aufgrund der einfacher zu berechnenden Formeln um 30% geringer ausfiel.

Dieses überraschende Ergebnis darf aber erst nach einer ausführlichen, wissenschaftlichen Evaluierung als gesichert angesehen werden.

Beitrag: Neue Methode zur Berechnung von Abstraktionen

Pokersoftware auf dem aktuellen Stand der Technik fasst zur Komplexitätsreduktion ähnliche *Gemeinschafts-* und *Handkarten* in eine vorgegebene Anzahl an Gruppen zusammen. Die Bestimmung der Ähnlichkeit erfolgt auf Basis des Potential-aware Abstandsmaßes. Zusätzlich werden zur weiteren Komplexitätsreduktion noch „Imperfect Recall“ Abstraktionen verwendet, durch die Aktionen in früheren *Wettrunden* für aktuelle Entscheidungen nicht mehr berücksichtigt werden können. Durch diese Ansätze wird die enorme Komplexität stark reduziert und beherrschbar gemacht. Das geschieht allerdings auf Kosten einer starken Verlustbehaftung und Verzerrung des ursprünglichen Spiels. Innerhalb der Abstraktion erreichen die berechneten Strategien zwar nahezu ein Nash-Equilibrium, bei der Transformation zurück auf das Originalspiel verlieren sie aber deutlich an Qualität. Dieses Problem lässt sich nur durch Verwendung von Abstraktionen mit geringeren oder bestenfalls keinen Verlusten lösen.

Deshalb untersucht diese Arbeit Perfect Recall Abstraktionen, die nicht auf Ähnlichkeit, sondern auf Identität beruhen. Für das in dieser Arbeit eingeführten Abstandsmaß Gegner-Sicht konnte eine Abstraktion erzeugt werden, auf der die berechneten Strategieprofile einen im Vergleich zum Stand der Technik wesentlich geringeren Abstand ε zum Nash-Equilibrium besitzen (siehe Abschnitt 7.3).

Beitrag: CFR Anpassungen für neue Abstraktionen

Die neu eingeführten Abstraktionen erzeugen *Handkarten* Abstraktionen, die sich in jeder *Wettrunde* in Abhängigkeit von den *Gemeinschaftskarten* unterscheiden. Das hat für den CFR Algorithmus (wie auch CGH) zur Konsequenz, dass beim übertragen des erwarteten Nutzens hin zur Wurzel des Spielbaumes die *Handkarten* gruppen keinen eindeutigen Vorgänger mehr haben. Gleiches gilt umgekehrt für die Berechnung der Gegner-*Range*, die von der Wurzel zu den Blättern hin erfolgt.

Um diesen Anforderungen gerecht zu werden, wurden neue mathematische Verfahren (siehe Abschnitt 5.4) eingeführt und in der Evaluierung erfolgreich getestet. Eine wichtige Idee war dabei die Verwendung einer neuen Menge an Kartengruppen, die für alle *Handkarten* jeweils aus dem Schnitt der verschiedenen Mengen der unterschiedlichen *Wettrunden* besteht, die diese *Handkarten* enthalten.

Beitrag: Optimierungen der Implementierung

Für die konkrete Implementierung wurde eine Reihe von Optimierungen (siehe Abschnitt 6.5) verwendet, durch die die Laufzeit wesentlich verringert werden konnte. Dazu zählt die kompakte Vorberechnung des *Pokerrangs* für gegebene *Hand-* und *Gemeinschaftskarten*, die sowohl für die Berechnung der Abstraktionen als auch für den CFR an allen zentralen Stellen benötigt wird. Auch die Methode der Vorberechnung wurde konsequent auf alle Berechnungen des CFR bzw. CGH ausgedehnt und am Ende verblieben als einzige Variablen nur noch die Wahrscheinlichkeitsverteilungen der Strategien.

Außerdem wurde die Speicherung aller Datenstrukturen, also des reduzierten Spielbaums, der Abstraktionen und der Strategien in Arrays statt Objekten dargestellt. Durch diese Darstellung konnten der Speicherbedarf und die Laufzeit weiter reduziert werden.

Diskussion der Beiträge

Die eigenen Beiträge verfolgen allesamt das Ziel, die Berechnung von Strategien auf Basis von verlustfreien Karten Abstraktion mit Perfect Recall für NLHE möglich zu machen. Dies ist ein wichtiges Zwischenziel, um dem letztendlichen Ziel, der Berechnung eines perfekten Nash-Equilibriums, ein Stück näher zu kommen. Es wurde gezeigt, dass durch Erweiterung der bisherigen Grundlagen bereits heute ein Nash-Equilibrium auf nahezu verlustfreien Karten Abstraktionen durch Dekomposition (siehe Abschnitt 5.1) des gesamten Problems erzeugt werden kann.

Das noch größere Problem besteht allerdings in der Abstraktion der *Wettrunden*. Bei NLHE sehen die Spielregeln vor, dass der Spieler eine beliebige Anzahl seiner *Chips* setzen kann, was eine dreistellige Anzahl an Aktionen des Spielers in einer großen Menge seiner Knoten zur Folge hat. Da die Komplexität des Baumes exponentiell vom Verzeigungsgrad abhängt, abstrahiert die bekannte Pokersoftware wie auch diese Arbeit die Anzahl der Kanten auf eine einstellige Anzahl. Dadurch wird der größte Teil der Komplexitätsreduktion im Rahmen von $\sim 10^{42}$ erzielt. Zur verlustfreien Abstrahierung dieser Komplexität, die die Voraussetzung zur Berechnung des exakten Nash-Equilibriums wären, existieren noch keine überzeugenden Ansätze.

8.2 Ausblick

Aus der Distanz betrachtet zeigen diese Beiträge auch, dass die Forschung um die Berechnung des spieltheoretischen Nash-Equilibriums noch am Anfang steht. Die Klärung folgender Fragen sollte sowohl im Hinblick auf NLHE als auch für Spiele im Allgemeinen zu weiteren Fortschritten führen:

1. In dieser Arbeit wurde mit dem CGH Algorithmus eine möglicherweise bessere Alternative zum CFR Algorithmus vorgestellt. In der Evaluierung zeigte sich eine höhere Konvergenzgeschwindigkeit, aber gleichzeitig auch die Eigenschaft, dass der Abstand zum Nash-Equilibrium nach manchen Iterationen ansteigt. Insgesamt ist das Verständnis des CGH noch oberflächlich und bedarf weiterer Untersuchungen. Halten die Evaluierungsergebnisse dieser Arbeit einer Überprüfung stand? Weist der CGH auch in anderen Spielen als Poker eine gegenüber dem CFR höhere Konvergenzgeschwindigkeit auf? Kann mathematisch gezeigt werden, dass ε immer gegen Null konvergiert? Ist es darüber hinaus möglich, eine mathematische Aussage über das Verhältnis der Konvergenzgeschwindigkeiten von CFR und CGH zu treffen? Wie hoch ist die Konvergenzgeschwindigkeit unter Verwendung der neuesten Sampling Techniken? Können die Anstiege von ε in manchen Iterationen vermieden werden, um eine nochmals verbesserte Konvergenzgeschwindigkeit zu erreichen?
2. Es gibt momentan keinen Ansatz, der verlustfreie Abstraktion der *Wettrunden* ermöglicht. Angesichts der enormen Komplexität der *Wettrunden* würde es selbst bei fortwährender Gültigkeit des Mooreschen Gesetzes noch ca. 100 Jahre dauern, bis Computer über die notwendige Rechenleistung verfügen. Wie stark ist der Verlust durch die verwendeten Abstraktionen? Welches Verfahren eignet sich, um nicht unterstützte Einsatzhöhen zu interpolieren? Kann das exakte Nash-Equilibrium trotz Abstraktionen berechnet werden?
3. Die Beiträge dieser Arbeit, insbesondere in Bezug auf die Berechnung von verlustarmen und -freien Abstraktionen, wurden nur zur Berechnung von Strategien für NLHE HU verwendet und evaluiert. Lässt sich das Abstandsmaß Gegner-Sicht auf andere Spiele oder reale Probleme übertragen und können damit Abstraktionen von ähnlich hoher Qualität erzeugt werden? Können die vorgestellten Verfahren zur Zusammenfassung und Trennung von Gruppen für den CFR Algorithmus auch in anderen Szenarien wichtig sein?
4. Wie sich zeigt, ist das Problem der Berechnung eines exakten Nash-Equilibriums für NLHE HU, also ein Spiel mit zwei Spielern, ein äußerst komplexes. Spätestens ab drei Spielern erscheint eine verlustfreie Berechnung eines Nash-Equilibriums eine Utopie. Daher muss die Komplexitätsreduktion von Abstraktionen gegen die Verlustbehaftung abgewogen werden. Wie ist das relative Verhältnis der Verlustbehaftung der verschiedenen Abstraktionen für Setzstruktur, *Hand*- und *Gemeinschaftskarten*? Wie steht dieses Verhältnis zum Nutzen, also zur Komplexitätsreduktion der jeweiligen Abstraktionen?
5. Derzeit werden Strategien mit einer simplen Regel (z.B. per Zufall) initialisiert. Gibt es einen besseren Algorithmus zur Initialisierung, dessen Laufzeit sich durch weniger benötigte Iterationen des CFR aufgrund einer zu Beginn höheren Qualität der Strategien amortisiert?
6. Bei den durch den CFR Algorithmus berechneten Strategien handelt es sich immer um gemischte, statische Strategien. Für Null-Summen Spiele kann man zeigen, dass der erwartete Nutzen einer solchen Strategie größer gleich Null beträgt. In der Realität sind die meisten Spiele allerdings keine Null-Summen Spiele und hierfür sind häufig dynamische Strategien, die entweder mit anderen Spielern kooperieren oder deren Schwächen gezielt exploitieren, wichtiger. Kann der CFR Algorithmus dazu beitragen solche Strategien zu berechnen? Welchen Beitrag kann das Exploitieren mithilfe eines Gegnermodells leisten?

8.3 Zusammenfassung

Ausgehend von den Ergebnissen der Weltmeisterschaft im Computer-Poker (ACPC) zeigt sich, dass die stärksten Poker-Programme auf der Approximation des Nash-Equilibriums beruhen. Im Kontext von Null-Summen Spielen, wie Poker, ist das auch folgerichtig, da das Nash-Equilibrium einen erwarteten Nutzen von größer gleich Null garantiert und von keiner Strategie dominiert werden kann.

Für die Approximation des Nash-Equilibriums verwenden mittlerweile alle Programme den 2007 entdeckten, iterativen Counterfactual Regret Minimization (CFR) [12] Algorithmus, der die Opportunitätskosten in jedem einzelnen Spielzustand reduziert und damit gleichzeitig die gesamte Strategie optimiert. In den Iterationen des CFR durchlaufen die Strategien der Spieler eine Evolution, in der sie in jedem Schritt durch Ausnutzen der Schwächen der jeweils anderen Strategie voneinander lernen.

In dieser Arbeit wurde mit dem CGH in Abschnitt 5.6 ein neuer, iterativer Algorithmus zur Approximation des Nash-Equilibriums eingeführt, der statt der Minimierung des Bedauerns auf einer $\frac{1}{n}$ Mischung der alten Strategie und der Aktion mit dem höchsten, erwarteten Nutzen beruht. In der Evaluierung in Abschnitt 7.4 zeigt sich, dass dieser Algorithmus in den verwendeten Szenarien eine, im Vergleich zum CFR, signifikant höhere Konvergenzrate aufweist und damit wesentlich schneller das Nash-Equilibrium approximiert. Aufgrund der späten Entdeckung innerhalb des Rahmens dieser Diplomarbeit sollte der Vergleich zum CFR aber erst nach weiterer Evaluierung als gesichert betrachtet werden.

Die Approximation des Nash-Equilibriums ist für NLHE HU Poker aufgrund der Komplexität (10^{71}) des Spiels nur unter Verwendung von komplexitätsreduzierenden Abstraktionen möglich. Selbst die neuesten Poker-Programme verwenden stark verlustbehaftete Abstraktionen, die ähnliche Spielzustände in einem einzigen Zustand zusammenfassen und damit die Spielregeln des ursprünglichen Spiels verzerren. Um die Qualität der erzeugten Strategien auch in Zukunft weiter zu verbessern ist es notwendig, die Abstraktionen stärker an das ursprüngliche Spiel anzugleichen.

In dieser Arbeit wurden in Kapitel 5 neue Verfahren zur Erzeugung von wesentlich verlustfreieren Abstraktionen für die Spielkarten vorgestellt, die nur strategisch identische Karten in Gruppen zusammenfassen. Durch die in Abschnitt 5.1 vorgestellte Dekomposition des Gesamtproblems ist es möglich geworden, die neuen Abstraktionen trotz deren vergleichsweise höherer Komplexität zu evaluieren. Für diese neuen Abstraktionen konnte in Abschnitt 7.3 gezeigt werden, dass die Qualität der berechneten Strategien nach der Transformation von dem abstrahierten auf das ursprüngliche Spiel verbessert wurde.

Auch wenn das Ziel der Berechnung des exakten Nash-Equilibriums für komplexe Spiele wie Poker noch in weiter Ferne liegt, wurden insbesondere in den letzten Jahren große Fortschritte, wie beispielsweise die Entdeckung des CFR, erzielt. Auch diese Arbeit konnte in den Bereichen Approximation des Nash-Equilibriums und Abstraktion des Spiels durch eigene Ideen einen Beitrag zum Fortschritt leisten.

Literaturverzeichnis

[1] "Wikipedia - Human–computer chess matches," ed, 2014.

[2] "Wikipedia - Computer Poker Players," ed, 2014.

[3] I. Fiedler and A.-C. Wilcke, "The market for online poker," *UNLV Gaming Research & Review Journal*, vol. 16, p. 1, 2012.

[4] A. Gilpin, T. Sandholm, and T. B. Sørensen, "A heads-up no-limit Texas Hold'em poker player: discretized betting models and automatically generated equilibrium-finding programs," in *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 2*, 2008, pp. 911-918.

[5] "Wikipedia - Game complexity," 2014.

[6] D. Billings, D. Papp, J. Schaeffer, and D. Szafron, "Poker as a Testbed for AI Research," in *Advances in Artificial Intelligence*, ed: Springer, 1998, pp. 228-238.

[7] J. Nash, "Non-cooperative games," *Annals of mathematics*, pp. 286-295, 1951.

[8] P. A. Piccione, "Mehen, mysteries, and resurrection from the coiled serpent," *Journal of the American Research Center in Egypt*, pp. 43-52, 1990.

[9] K. Salen and E. Zimmerman, *Rules of play: Game design fundamentals*: MIT press, 2004.

[10] M. J. Osborne and A. Rubinstein, *A course in game theory*: MIT press, 1994.

[11] H. W. Kuhn, *Lectures on the Theory of Games (AM-37)*: Princeton University Press, 2003.

[12] M. Zinkevich, M. Johanson, M. Bowling, and C. Piccione, "Regret minimization in games with incomplete information," in *Advances in neural information processing systems*, 2007, pp. 1729-1736.

[13] P. Morris, *Introduction to game theory* vol. 1: Springer New York etc, 1994.

[14] C. A. Holt and A. E. Roth, "The Nash equilibrium: a perspective," *Proc Natl Acad Sci U S A*, vol. 101, pp. 3999-4002, Mar 23 2004.

[15] M. B. Johanson, "Robust strategies and counter-strategies: Building a champion level computer poker player," in *Masters Abstracts International*, 2007.

[16] B. Greenstein, *Ace on the River*: Last Knight Publishing, 2005.

[17] D. Sklansky, *The theory of poker*: Two Plus Two Publishing LLC, 1999.

[18] J. Rubin and I. Watson, "Computer poker: A review," *Artificial Intelligence*, vol. 175, pp. 958-987, 2011.

[19] A. Davidson, "Opponent modeling in poker: Learning and acting in a hostile and uncertain environment," 2002.

[20] @mizdflop and @dave_j_thornton. (2014, OneBillionHands. Available: http://www.onebillionhands.com/our-data/

[21] M. Ghallab, "Adaptive play in texas hold'em poker," in *ECAI 2008: 18th European Conference on Artificial Intelligence, July 21-25, 2008, Patras, Greece: Including Prestigious Applications of Intelligent Systems (PAIS 2008): Proceedings*, 2008, p. 458.

[22] H. Quek, C. Woo, K. Tan, and A. Tay, "Evolving Nash-optimal poker strategies using evolutionary computation," *Frontiers of Computer Science in China*, vol. 3, pp. 73-91, 2009.

[23] R. Gibson, "Regret minimization in non-zero-sum games with applications to building champion multiplayer computer poker agents," *arXiv preprint arXiv:1305.0034*, 2013.

[24] S. Ganzfried and T. Sandholm, "Tartanian5: A heads-up no-limit Texas Hold'em poker-playing program," in *Computer Poker Symposium at the National Conference on Artificial Intelligence (AAAI)*, 2012.

[25] ACPC. (2014). *The annual computer poker competition*. Available: http://www.computerpokercompetition.org/

[26] E. Jackson, "Slumbot NL: Solving Large Games with Counterfactual Regret Minimization Using Sampling and Distributed Processing," in *Workshops at the Twenty-Seventh AAAI Conference on Artificial Intelligence*, 2013.

[27] M. Schmid, "Game Theory and Poker," Master, Faculty of Mathematics and Physics, Charles University, Prague, 2013.

[28] L. E. J. Brouwer, "Über abbildung von mannigfaltigkeiten," *Mathematische Annalen*, vol. 71, pp. 97-115, 1911.

[29] M. Johanson, "Measuring the size of large no-limit poker games," *arXiv preprint arXiv:1302.7008*, 2013.

[30] X. Chen and X. Deng, "Settling the Complexity of Two-Player Nash Equilibrium," in *FOCS*, 2006, p. 47th.
[31] C. H. Papadimitriou, "On the complexity of the parity argument and other inefficient proofs of existence," *Journal of Computer and system Sciences*, vol. 48, pp. 498-532, 1994.
[32] C. Daskalakis, P. W. Goldberg, and C. H. Papadimitriou, "The complexity of computing a Nash equilibrium," *SIAM Journal on Computing*, vol. 39, pp. 195-259, 2009.
[33] J. Shi and M. L. Littman, "Abstraction methods for game theoretic poker," in *Computers and Games*, ed: Springer, 2001, pp. 333-345.
[34] A. Gilpin and T. Sandholm, "Lossless abstraction of imperfect information games," *Journal of the ACM (JACM)*, vol. 54, p. 25, 2007.
[35] M. Lanctot, K. Waugh, M. Zinkevich, and M. Bowling, "Monte Carlo sampling for regret minimization in extensive games," in *Advances in Neural Information Processing Systems*, 2009, pp. 1078-1086.
[36] M. Johanson, N. Bard, M. Lanctot, R. Gibson, and M. Bowling, "Efficient Nash equilibrium approximation through Monte Carlo counterfactual regret minimization," in *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, 2012, pp. 837-846.
[37] D. Billings, N. Burch, A. Davidson, R. Holte, J. Schaeffer, T. Schauenberg, *et al.*, "Approximating game-theoretic optimal strategies for full-scale poker," in *IJCAI*, 2003, pp. 661-668.
[38] R. Gibson, "REGRET MINIMIZATION IN GAMES AND THE DEVELOPMENT OF CHAMPION MULTIPLAYER COMPUTER POKER-PLAYING AGENTS," University of Alberta, 2014.
[39] A. Gilpin and T. Sandholm, "Better automated abstraction techniques for imperfect information games, with application to Texas Hold'em poker," in *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*, 2007, p. 192.
[40] K. Waugh, M. Zinkevich, M. Johanson, M. Kan, D. Schnizlein, and M. H. Bowling, "A Practical Use of Imperfect Recall," in *SARA*, 2009.
[41] M. Johanson, N. Bard, N. Burch, and M. Bowling, "Finding Optimal Abstract Strategies in Extensive-Form Games," in *AAAI*, 2012.
[42] T. P. T. P. LLC. (2014). *Poker / Gaming Books and Strategy Forums*. Available: http://forumserver.twoplustwo.com/
[43] D. Harrington, "HARRINGTON ON HOLD 'EM VOLUMEN I: STRATEGIC PLAY," *Two Plus Two Publishing*, 2004.
[44] A. Gilpin, T. Sandholm, and T. B. Sørensen, "Potential-aware automated abstraction of sequential games, and holistic equilibrium analysis of Texas Hold'em poker," in *PROCEEDINGS OF THE NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE*, 2007, p. 50.
[45] M. Johanson, N. Burch, R. Valenzano, and M. Bowling, "Evaluating state-space abstractions in extensive-form games," in *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, 2013, pp. 271-278.
[46] D. Sklansky and M. Malmuth, *Hold'em poker for advanced players*: Two Plus Two Publishing LLC, 1999.

Abbildungsverzeichnis

Tabellenverzeichnis

Glossar – Poker Vokabular

All-In Bezeichnet eine Aktion, bei der der Spieler alle seine *Chips* auf einmal setzt. Bei einem *All-In* kann der Spieler aus der aktuellen *Spielrunde* nicht mehr aussteigen. Der Spieler gewinnt den *Pot*, sofern entweder alle anderen Spieler mit einem *Fold* reagieren oder er im *Showdown* die stärkste Hand zeigt.

Bet Mit dieser Aktion tätigt ein Spieler den ersten Einsatz von *Chips* in der aktuellen *Wettrunde*.

Big Blind Der größere, der beiden *Blind* Einsätze.

Blind Ein *Blind* ist ein verpflichtender Einsatz von *Chips* zu Beginn der *Preflop Wettrunde*. Bei NLHE gibt es zwei verschiedene *Blind* Einsätze, den *Small Blind* und den *Big Blind*.

Bluff Ein *Bluff* im Poker bezeichnet eine Aktion, bei der ein Spieler mit einer *Hand* von geringer Stärke *Chips* setzt und versucht den *Pot* dadurch zu gewinnen, dass alle gegnerischen Spieler aussteigen.

Board Bezeichnet die bereits aufgedeckten *Gemeinschaftskarten*.

Button Der Spieler mit der günstigsten *Position* wird mit einem *Button* (auch *Dealer-Button*) gekennzeichnet. Der Spieler „am *Button*" hat den Vorteil, dass er innerhalb einer *Wettrunde* immer als letzter das Setzen von *Chips* entscheiden kann.

Call Ist eine Aktion die bedeutet, dass der Spieler so viele weitere *Chips* in der aktuellen *Wettrunde* setzt, dass er mit dem aktuell höchsten Einsatz gleichzieht.

Cashgame Hierbei entsprechen die *Chips* echtem Geld. Spieler können einem *Cashgame* jederzeit beitreten oder es verlassen.

Check Mit dieser Aktion, die nur möglich ist sofern in der aktuellen *Wettrunde* noch keine *Chips* gesetzt wurden, setzt der aktuelle Spieler keine *Chips* und der nächste Spieler kommt an die Reihe.

Chip Jeder Spieler hat eine Anzahl von *Chips* vor sich liegen, die sein Spielgeld repräsentieren.

Dealer Die Rolle des Spielmanagers, der die *Handkarten* verteilt, neue *Gemeinschaftskarten* aufdeckt und am Ende einer *Spielrunde* den Gewinner feststellt.

Flop Hierbei handelt es sich um die 2. *Wettrunde* bei NLHE, bei der der *Dealer* zu Beginn drei *Gemeinschaftskarten* (auch „*Flop*" genannt) aufdeckt.

Flush Ein *Pokerrang*, der aus fünf Karten mit derselben Farben besteht und nur noch von einem Full-House, Vierling oder Straight *Flush* geschlagen wird.

Gemeinschaftskarte Eine Karte, die für alle Spieler zugleich für den *Pokerrang* gilt und damit Teil der Hand werden kann. Bei NLHE werden bis zur 4. *Wettrunde* insgesamt fünf *Gemeinschaftskarten* aufgedeckt.

Hand Eine Hand wird im Poker immer aus fünf Karten gebildet. Im Falle von NLHE werden dazu die für den Spieler besten fünf Karten aus *Handkarten* und *Board* gebildet.

Handkarten Bei NLHE erhält ein Spieler zu Beginn einer *Spielrunde* zwei Karten, die sogenannten *Handkarten*.

Heads up Eine Pokerrunde mit genau zwei Spielern.

Hold'em Pokervariante, bei der jeder Spieler zwei *Handkarten* erhält und insgesamt in vier *Wettrunden* fünf *Gemeinschaftskarten* aufgedeckt werden.

IN Eine Bezeichnung für die *Position*, die im Fall von NLHE HU immer der strategisch besseren *Position* am *Button* entspricht, bei der der Spieler (außer *Preflop*) als letzter über Einsätze entscheiden kann.

Limit Bei dieser Variante kann ein Spieler, im Gegensatz zu *No Limit*, nur eine fest vorgegebene Anzahl an *Chips* einsetzen. Damit werden die möglichen Aktionen jedes Spielers auf maximal drei (*Fold*, *Call*, *Raise*) beschränkt.

No Limit Poker Variante, bei der jeder Spieler eine beliebige Anzahl an *Chips* setzen kann.

OUT Bezeichnet im Falle von NLHE HU die schlechtere *Position*, bei der ein Spieler als erster über einen Einsatz von *Chips* entscheiden muss.

Pokerrang Der *Pokerrang* gibt an, wie stark eine Hand ist. Durch Vergleich des *Pokerrangs* wird im Falle eines *Showdowns* der Gewinner ermittelt. Bei gleichem *Pokerrang* findet ein *Split* statt.

Position Gibt an, wann der Spieler innerhalb einer *Wettrunde* handeln muss. Ein Spieler „in *Position*" ist immer nach einem Spieler „außerhalb der *Position*" an der Reihe.

Pot Bezeichnet die Summe der in den bisherigen und der aktuellen *Wettrunde* gesetzten *Chips*.

Pot Odds Ist das Verhältnis eines Einsatzes in Relation zur Größe des *Pots*. Diese Relation ist für die Bewertung eines Einsatzes wichtiger, als deren absolute Höhe.

Preflop Ist bei NLHE die erste *Wettrunde*, zu deren Beginn jeder Spieler seine *Handkarten* erhält.

Raise Eine Aktion, bei der der Spieler den von einem anderen Spieler getätigten Einsatz erhöht. Bei NLHE kann der aktuelle Einsatz pro *Wettrunde* maximal dreimal durch ein *Raise* erhöht werden.

Rake Das *Rake* ist ein, in aller Regel, prozentualer Anteil (i.d.R. 2% bis 10%), der durch den *Dealer* aus dem *Pot* entnommen wird. Das *Rake* entspricht damit den Einnahmen des Casinos.

Range Bezeichnet eine Wahrscheinlichkeitsverteilung über alle möglichen *Handkarten*. Die Gegner-*Range* ist also *Range* der *Handkarten* des jeweils anderen Spielers. Mithilfe der genauen Kenntnis einer Strategie kann für jeden Knoten im Spielbaum durch Multiplikation der Wahrscheinlichkeiten, der zu diesem Knoten hinführenden Aktionen (Kanten), die genaue *Range* berechnet werden. Sie wird benötigt, um im CFR Algorithmus in den Blättern den erwarteten Nutzen zu bestimmen.

River Bezeichnet die 4. und letzte *Wettrunde*, zu deren Beginn die fünfte *Gemeinschaftskarte* aufgedeckt wird.

Showdown Ein *Showdown* findet dann statt, wenn entweder alle Spieler *All-In* sind oder am Ende der letzten *Wettrunde* noch mehr als ein Spieler im Spiel ist. In diesem Fall decken die Spieler ihre Karten auf und der Spieler mit dem höchsten *Pokerrang* gewinnt den *Pot*.

Small Blind Der kleinere, der beiden *Blind* Einsätze.

Spielrunde Eine *Spielrunde* ist ein vollständiges Spiel, das mit der ersten *Wettrunde* beginnt und damit endet, dass der *Pot* vergeben wird. Der *Pot* kann dadurch vergeben werden, dass alle bis auf einen Spiel *geFolded* haben oder durch einen *Showdown*.

Split Sollte im Falle eines *Showdowns* der höchste *Pokerrang* bei mehreren Spielern gleichzeitig auftreten, wird der *Pot* unter diesen Spielern gleichmäßig aufgeteilt.

Stack Bezeichnet die *Chips* eines Spielers.

Straight Ein *Pokerrang*, der aus fünf Karten mit aufeinanderfolgender Wertigkeit besteht. Ein Ass kann hierbei sowohl als höchste, wie auch als niedrigste Karten verwendet werden.

Texas No Limit Hold'em Die aktuell populärste Poker-Variante, deren Lösung als eines großes Problem der in der künstlichen Intelligenz bzw. der Spieltheorie gilt. Die Regeln von NLHE sind in Abschnitt 2.2 dargestellt.

Turn Entspricht der 3. und vorletzten *Wettrunde*. Zu Beginn des *Turns* wird eine *Gemeinschaftskarte* aufgedeckt.

Wettrunde Innerhalb einer *Wettrunde* können alle noch verbleibenden Spieler Einsätze (*Bet*) tätigen, Einsätze erhöhen (*Raise*), aussteigen (*Fold*) oder den Einsatz nicht erhöhen (*Check*). Die *Wettrunde* endet, sofern entweder alle Spieler bis auf einen ausgeschieden sind oder der Einsatz eines Spielers von keinem anderen Spieler nochmals erhöht wurde.

Anhang - Auszug einer berechneten Strategie

Abbildung 22 und Abbildung 23 zeigen auszugsweise die gemischten Strategien zweier Spieler. Es werden nur Aktionen mit einer Wahrscheinlichkeit $> 0\%$ dargestellt. Die Anzahl der Spielzustände ist $1{,}9 * 10^{10}$. Es wird der Teilbaum vom Flop Q♠T♥7♥ bis River Q♠T♥7♥8♠5♠ gezeigt. Dieser Teilbaum entspricht $\frac{1}{9385190}$ der gesamten Strategie.

Der *OUT* Spieler besitzt die *Handkarten* K♥J♥ und der *IN* Spieler die *Handkarten* A♣A♠. Im *Pot* befinden sich zu Beginn des *Flop* $2, beide Spieler besitzen jeweils noch $49 in *Chips*. Es wurde für beide Spieler eine Range von 22% angenommen. Dies ist eine typische Situation, bei der der Spieler in *Position* mit starken *Handkarten* (AA) *Preflop* erhöht hat und der *OUT* Spieler *Preflop callt*. Der *Flop* ist günstig für den *OUT* Spieler, dem nur noch eine Karte zum ♥-*Flush* oder einer *Straight* fehlt. Klar zu erkennen ist, dass es fast immer zwei Aktionen mit einer positiven Wahrscheinlichkeit gibt. Abbildung 23 führt das Beispiel fort und zeigt die Strategien in der *Wettrunde River*. Da weder die richtige Karte für *Flush* noch *Straight* kam, besitzt der *OUT* Spieler jetzt eine wertlose *Hand* und *blufft* insgesamt in 60% der Fälle. Der *Bluff* gelingt so gut wie nie, sofern der *IN* Spieler das A♠ besitzt und damit ein ♠-*Flush* unwahrscheinlicher wird. Besitzt er stattdessen aber das A♦, dann *foldet* er beim Einsatz von $12 zu 32,1% und wenn der *OUT* Spieler *All-In* geht zu 86,1%.

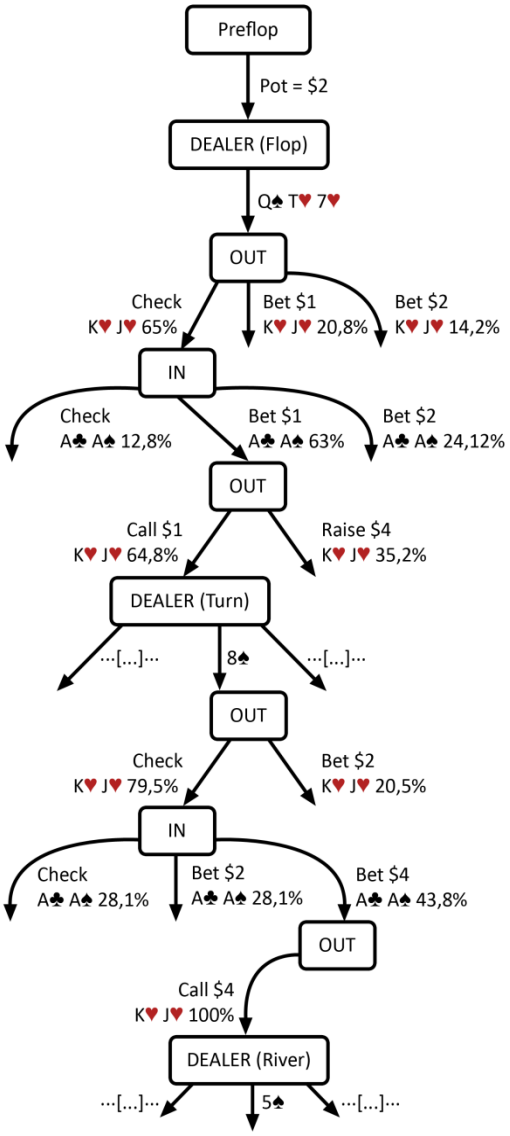


Abbildung 22: Auszug berechneter Strategie, die mit ε= 30,1 mbb/g das Nash-Equilibrium approximiert.

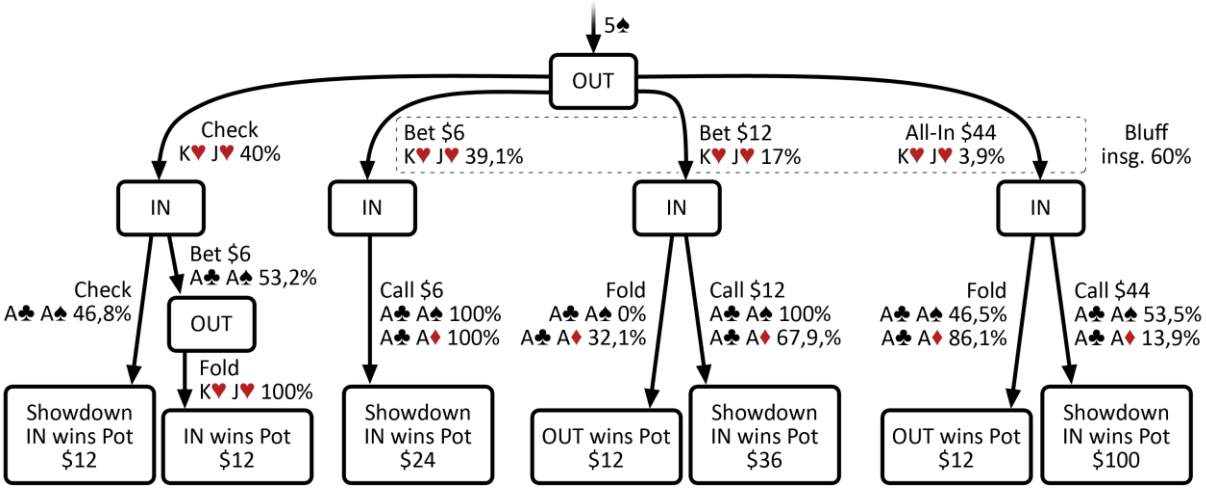


Abbildung 23: Setzt Abbildung 22 fort. Spieler *OUT blufft* und Spieler *IN*s Reaktion hängt vom Besitzt von A♠ ab.

Konkrete Erkenntnisse für *Texas No Limit Hold'em* Post-Flop

Zur Evaluierung der Ergebnisse dieser Arbeit wurden mithilfe der parallel entwickelten Software viele Strategien für konkrete Situationen analysiert. Aus der Betrachtung dieser Strategien ergaben sich die folgenden Beobachtungen für Strategien in der Nähe des Nash-Equilibriums:

- Ein Spieler sollte *Chips* setzen (*Bet* oder *Raise*), sofern er entweder über eine starke *Hand* verfügt und eine die große Menge an möglichen gegnerischen *Händen* schlägt (Value *Bet*) oder der Spieler eine schwache *Hand* hält, aber die relative Handstärke des Gegners aufgrund der zuletzt aufgedeckten *Gemeinschaftskarten* (wie im Beispiel in Abbildung 23) gesunken ist.
    - Ein solcher *Bluff* darf aber niemals ein Automatismus werden, sondern je nach Situation nur mit einer Wahrscheinlichkeit im Bereich von ca. 10% bis 70% erfolgen.
    - Mit *Handkarten* die sich eher im mittleren Bereich der relativen Handstärke bewegen, sollte sich der Spieler häufiger passiv verhalten

- Mit einer Strategie unter der Annahme „in Situation X ist immer die Aktion A das Richtige", also einer reinen Strategie, wird ein Spieler gegenüber Profis hoffnungslos unterlegen sein. Jede reine Strategie ist stark exploitierbar.

- Die *Gemeinschaftskarten* auf dem *Board* haben einen dominierenden Einfluss auf die Bewertung der *Hand* eines Spielers. Jede neue *Gemeinschaftskarte* kann die strategische Situation dramatisch ändern. Deshalb müssen Spieler nach jeder neuen *Gemeinschaftskarte* ihre strategische Situation neu bewerten.

- Die *Position* des Spielers hat einen großen Einfluss auf dessen Erwartungswert und auch auf seine konkrete Strategie.
    - Auffällig war in den Ergebnissen, dass ein Spieler, der sich beim Aufdecken einer neuen *Gemeinschaftskarte* außerhalb der Position (*OUT*) befindet, in aller Regel *Check* spielen sollte.

- Wurden die Strategien isoliert für nur eine Wettrunde berechnet, ergab sich ein wesentlich aggressiveres Setzverhalten als bei der Berechnung von *Flop* bis *River*. Bei letzterem wurden Aktionen in großer Voraussicht auf die möglichen, späteren Zustände getroffen.

Erklärung

Ich versichere, dass ich diese Diplomarbeit selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel verwendet habe.

_____ _____

Ort & Datum Holger Kujath