OSTBAYERISCHE
TECHNISCHE HOCHSCHULE
REGENSBURG

Entwicklung einer künstlichen Intelligenz für Brettspiele und deren Anbindung an eine Touch-Hardware über mobiele Endgeräte

An der Fakultät für Informatik und Mathematik der
Ostbayerischen Technischen Hochschule Regensburg
im Studiengang
Techniche Informatik

eingereichte

Bachelorarbeit

zur Erlangung des akademischen Grades des
Bachelor of Science (B.Sc.)

Vorgelegt von: Korbinian Federholzner
Matrikelnummer: 3114621

Erstgutachter: Prof. Dr. Carsten Kern
Zweitgutachter: Prof. Dr. Daniel Jobst

Abgabedatum: 31.08.2020

Erklärung zur Bachelorarbeit

1. Mir ist bekannt, dass dieses Exemplar der Abschlussarbeit als Prüfungsleistung in das Eigentum der Ostbayerischen Technischen Hochschule Regensburg übergeht.
2. Ich erkläre hiermit, dass ich diese Abschlussarbeit selbständig verfasst, noch nicht anderweitig für Prüfungszwecke vorgelegt, keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie wörtliche und sinngemäße Zitate als solche gekennzeichnet habe.

Regensburg, den 19. Juli 2020

_____

Korbinian Federholzner

Zusammenfassung

In der folgenden Arbeit wird ...

Inhaltsverzeichnis

I Abkürzungsverzeichnis **VII**

1 Einleitung **1**
1.1 Motivation . . . 1
1.2 Aufgabenstellung . . . 1
1.3 Struktur dieser Arbeit . . . 1

2 Grundlagen **2**
2.1 Dame . . . 2
2.1.1 Internationale Dame . . . 2
2.2 Minimax . . . 3
2.3 Iterative Deepening . . . 4
2.4 Alpha-Beta Pruning . . . 4
2.5 Zugsortierung . . . 4
2.6 Monte Carlo Tree Search (MCTS) . . . 4

3 Anforderungsanalyse **5**
3.1 Anwendungsszenario . . . 5
3.2 Anforderungen an die Software . . . 6
3.2.1 Funktionale Anforderungen . . . 6
3.2.2 Nichtfunktionale Anforderungen . . . 7
3.2.3 Zusammenfassung der Anforderungen . . . 7

4 Architektur der Software **8**
4.1 Überblick . . . 8
4.2 Gameserver . . . 8
4.3 ReversiXT GUI . . . 9
4.4 KI Client . . . 9

5 Hardware **9**
5.1 Raspberry Pi . . . 9
5.2 Touch Monitor . . . 9

6 Imlementierung **9**
6.1 Eingesetzte Sofwarekomponenten . . . 9
6.1.1 Programmiersprachen und Frameworks . . . 9
6.1.2 Datentransferprotokolle . . . 9
6.2 Gameserver . . . 9
6.2.1 Netzwerkspezifikation des Gameservers . . . 9

6.3 Graphische Oberfläche 9
6.3.1 Gegebene React Anwendung 9
6.3.2 Erweiterungen 9
6.4 KI Client 9
6.4.1 Vergleich der KI Algorithmen 9

7 Testing **9**
7.1 Integrationstest 9
7.2 Ergebnisse 9

8 Fazit und Ausblick **10**

Anhang **I**

A Domändenmodell **I**

Abbildungsverzeichnis

1 DameSpielfeld 3
2 minimax 3

Tabellenverzeichnis

1 Anforderungstabelle 8

I Abkürzungsverzeichnis

1 Einleitung

Das Thema dieser Arbeit ist die Implementierung einer künstlichen Intelligenz für diverse Brettspiele. Dabei soll die resultierende Software auf einem von einem Raspberry Pi gesteuerten Touch-Bildschirm laufen, durch welchen ein Benutzer die künstliche Intelligenz herausfordern kann. Außerdem gibt es die Möglichkeit, sich mit der Hardware über ein mobiles Endgerät wie einem Smartphone zu verbinden, um die KI oder den Spieler, der den Touch-Bildschirm bedient, herauszufodern.

1.1 Motivation

Das Feld der künstlichen Intelligenz ist momentan eines der sich am schnellsten entwickelnden Felder der Informatik. Dabei spielt die Spieltheorie schon seit Anfang eine große Rolle. So bieten klassische Brettspiele wie z. B. Schach, Dame oder Mühle nicht nur eine klar definierte Abstraktion von Problemen der realen Welt, sondern sie können auch von dem Großteil der Bevölkerung verstanden und gespielt werden. Das Meistern einer dieser Brettspiele wird auch oft mit hohem Grad an Intelligenz gleichgesetzt. Viele Algorithmen, die in der Spieltheorie entwickelt wurden, haben sich auch erfolgreich auf andere Felder der Informatik übertragen lassen. Ebenso hat sich die Art, wie Brettspiele gespielt werden, durch das Verwenden von künstlicher Intelligenz auch verändert, da die KI Züge in Betracht zieht, die auf den ersten Blick recht ungewöhnlich und nachteilhaft aussehen, sich aber als extrem stark herausstellen. Diese Arbeit versucht eine künstliche Intelligenz für Brettspiele, wie Dame, zu implementieren.

1.2 Aufgabenstellung

Im Rahmen der Bachelorarbeit soll eine künstliche Intelligenz entwickelt werden, welche Brettspiele, wie Dame, spielen kann. Gegeben ist eine Software, bei welcher man in der Lage ist, das Spiel ReversiXT (Reversi Extreme) gegen eine KI, sowie sich selbst zu spielen. Diese Software soll um einen Game Server, die KI und das neue Spiel in der GUI, erweitert werden. Außerdem soll ein Benutzer in der Lage sein, sich mit seinem Smarthphone mit der Hardware zu verbinden, um neue, sowie die alte KI herausfordern zu können.

1.3 Struktur dieser Arbeit

Die Arbeit ist folgendermaßen aufgebaut:
Kapitel 2 befasst sich mit den Grundlagen zu Dame, sowie den verwendeten künstliche Intelligenz Algorithmen. Dabei werden zu Dame auch die Grundregeln der verwendeten Variante erklärt. Bei den KI Algorithmen handelt es sich um die Algorithmen, die in der Arbeit verwendet und miteinander verglichen werden.
In Kapitel 3 werden die Anforderungen, welche gefordert sind, vorgestellt.

2 Grundlagen

Dieses Kapitel gibt einen Überblick über die theoretischen Grundlagen, die für das Verständnis dieser Arbeit notwendig sind. Zunächst werden die Grundregeln des behandelten Brettspieles vorgestellt. Darauf folgt eine Erklärung der künstlichen Intelligenz Algorithmen, welche für folgenden Kapitel von großer Relevanz sind.

2.1 Dame

Dame ist eines der ältesten Brettspiele, wobei erste Varianten 3000 v. Chr. im irakischen Ur entdeckt wurden. In der heutigen Zeit werden verschiedene Varianten desselben Spieles weltweit gespielt. So wird in vielen englischsprachigen Ländern eine andere Version gespielt als im Rest der Welt, was auch als Internationale Dame bekannt ist. Unterschiede sind z. B. bei diesen Varianten, dass bei internationaler Dame, Damen beliebig viele Felder in alle Richtungen springen dürfen, in anderen Varianten jedoch nur 1 Feld. Da die Regeln der internationalen Dame weiter verbreitet sind und auch im Vereinssport praktiziert werden, wird sich diese Arbeit im folgenden auf diese Regeln fokussieren. [Dra]

2.1.1 Internationale Dame

In dieser Variante des Spieles wird auf einem 10x10 Brett mit Schachbrett-Muster gespielt. Siehe Abbildung 1. Die Spielsteine sind scheibenförmig und in zwei Farben vorhanden, meist schwarz und weiß und dürfen nur auf den dunklen Feldern des Schachbrettes bewegt werden. Es gibt zwei Arten von Spielsteinen. Normale Spielsteine, welche nur in Richtung des Gegners bewegt werden, aber Rückwerts schlagen dürfen, und Damen, welche in alle Richtungen beliebig viele Felder fahren und schlagen dürfen. Allgemein herrscht Schlagzwang, was bedeutet, dass falls ein Spieler die Möglichkeit hat zu schlagen, er auch schlagen muss. Ein normaler Spielstein wird zur Dame, falls er in die hinterste Reihe des Gegners kommt. Ziel des Spieles ist es, entweder alle Steine des Gegners zu schlagen, oder den Gegner in eine Situation zu zwingen, in der er keine Züge mehr machen kann. [Int]

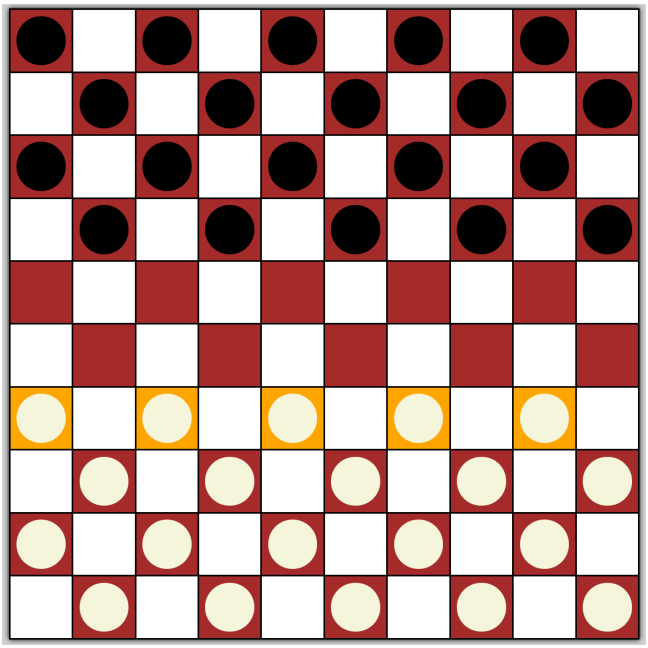

Abbildung 1: Das 10x10 Spielfeld aus der Anwendung

2.2 Minimax

Der Minimax Algorithmus wird verwendet, um einen optimalen Spielzug in Spielen mit perfekter Information zu finden. Allgemein wird eine Bewertungsfunktion verwendet, welche einen Zustand des Spieles bewertet und einen Integer Wert zurückgibt. Sehr niedrige Werte (negative) sind gut für Spieler A und sehr hohe Werte (positive) sehr gut für Spieler B. Um nun mehrere mögliche Zustände des Spieles zu berechnen und dann miteinander vergleichen zu können wird ein Suchbaum verwendet. siehe Abblidung 2. Der Algorithmus beginnt bei den untersten Blättern und geht nach oben bis zur Wurzel. Auf jeder Ebene wird versucht, je nach dem welcher Spieler an der Reihe ist, den Knoten zu maximieren oder zu minimieren. Der Wert, welcher der Wurzel zugewiesen wird, ist auch der Spielzug, der als Nächstes gespielt wird. [Rus12]

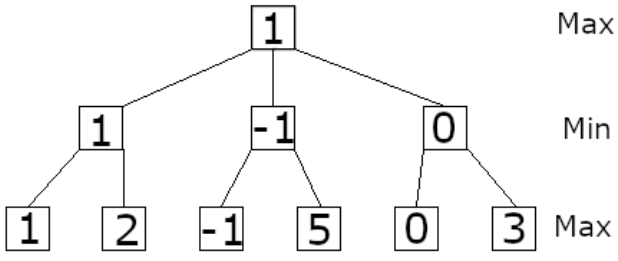

Abbildung 2: Beispiel eines Minimax Suchbaumes

2.3 Iterative Deepening

In komplexeren Spielen wie Go, Schach oder Dame ist es wegen dem Rechenaufwandt sehr schwer den Kompletten Baum von Minimax bis zu den Endzuständen aufzubauen. Deswegen versucht man schon vorher die Suche bei einer gewissen Tiefe abzubrechen um die Zustände nach einer Bewertungsfunktion zu bewerten und den Richtisten zu wählen. Dadurch, dass bei diesen Spielen oft mit Zeitlimit gearbeitet wird, ist es schwer vorrauszusagen bei welcher Tiefe man Abbrechen soll. Das Iterative Deepening fängt an bis zu einer niedrigen Tiefe an zu rechnen, speichert das Ergbnis, erhöht die Tiefe und fängt dann erneut von Anfang an. Dies wird so lange wiederholt, bis die Zeit abgelaufen ist. Das gespeicherte Ergbnis nach dem Ablaufen der Zeit ist das Ergebnis das verwendet wird. [Rus12]

2.4 Alpha-Beta Pruning

Das Alpha-Beta Pruning ist eine Optimierung zum Minimax Algorithmus. Die Idee des Algorithmus ist, dass manche Zweige des Suchbaums nicht untersucht werden müssen, da für den anderen Spieler diese Züge nicht in Frage kommen. Hierbei ist α der Wert für den Spieler, für den die niedrigen Werte besser sind und β für den anderen Spieler. Für jeden Knoten, je nach dem, ob er ein maximierender oder ein minimierender Knoten ist, wird überprüft, ob ein Kind-Knoten, welcher einen neuen Wert erhalten hat, nicht mehr vom Knoten beachtet werden muss. Z. B. ist der zu beachtende Knoten ein minimierender Knoten und hat aus seinem Linken Zweig eine 8 bekommen. Der rechte Kind-Knoten ist ein maximierender Knoten und hat von seinen Kindern eine 9 bekommen, sodass es nun egal ist, welche weiteren Werte er von seinen anderen Kindern noch bekommt. Er wird seinen Wert nicht mehr unter 8 ändern, was zur Folge hat, dass der rechte Zweig komplett ignoriert werden kann. [Rus12]

2.5 Zugsortierung

Zugsortierung ist eine Verbesserung der Alpha-Beta Suche. Da Alpha-Beta Pruning abhängig von der Reihenfolge in der die Zustände untersucht werden ist, ist es sinnvoll die Nachfolger zu wählen, die die besten Werte erbringen. Den Besten Nachfolger findet man, in dem man eine weitere Bewertungsfunktion einbaut, welche nicht so genau wie die Bewertungsfunktion am Ende sein muss. Wenn ein Knoten also alle möglichen Nachfolgerzüge als Kinder bekommt, werden auf diese die vereinfachte Bewertungsfunktion angewandt und, je nach Ergebnis der Funktion werden die Nachfolger sortiert. Dadruch dass der Beste Zug nun sehr weit am Anfang steht, ist es sehr warscheinlich das die anderen Züge durch Alpha-Beta Pruning ignoriert werden. [Rus12]

2.6 Monte Carlo Tree Search (MCTS)

Der Monte Carlo Tree Search Algorithmus, ist ein heuristischer Algorithmus, bei welchem von einem Zustand eines Spieles zufällig endlich viele Simulationen durchgeführt werden. Die Simulation endet, wenn ein Ergebnis des simulierten Spieles feststeht. Das Wiederholen der

Simulationen aus verschiedenen Knoten hat zur Folge, dass das Ergbnis immer genauer wird. Am Ende wird der Knoten gewählt, bei dem die Simulationen die besten Ergbnisse für den momentanen Spieler gezeigt haben. Ein Vorteil des MCTS-Algorithmus gegenüber Minimax ist, dass erst am Ende eines Durchlaufs eine Bewertungsfunktion benötigt wird. Allgemin besteht der Algorithmus aus vier Schritten:

- *Selektion*: Versucht wird, einen Zustand zu finden der noch erweiterbar ist, also einen Zustand zu finden, der kein Endzustand ist und noch nicht besuchte Züge hat.
- *Expansion*: Der Spielbaum wird zufällig um einen noch nicht besuchten Zug erweitert.
- *Simulation*: Von dem gewählten Knoten aus wird nun ein Spiel zufällig bis zum Ende simuliert.
- *Backpropagation*: Das Ergebnis der Simulation wird den vorhergehenden Knoten mitgeteilt und diese werden mit diesem aktualisiert.

Da man im Normalfall nicht beliebig viel Zeit hat alle Möglichkeiten zu simulieren, versucht man die limitierte Zeit so gut wie möglich zu nutzen und die richten Knoten zum Expandieren zu wählen. Dazu wird der *upper confidence bound for trees* (UCT) verwendet. Die UCT Formal lautet:

$$w + c\sqrt{\frac{\log N}{n}} \tag{1}$$

Wobei w die prozentuale Anzahl an Gewinnen des Knoten, N die Anzahl der gesamten Expansionen und n die Expansionen nur an dem betrachteten Knoten sind. Die Aufgabe der UCT Formel ist das Erreichen von zwei im Konflikt stehenden Zielen. Das erste Ziel ist es die Knoten die bisher die höchsten Chancen auf den Gewinn haben tiefer zu simulieren, um eine bessere Genauigkeit des besten Zuges zu haben. Das zweite Ziel ist Knoten die noch nicht sehr oft besucht worden sind genauer zu untersuchen, da diese vielversprechender sein könnten als gedacht. Für die Balanzierung der beiden Ziele gibt es den Parameter c [MP19].

3 Anforderungsanalyse

In diesem Kapitel wird zuerst ein beispielhaftes Szenario gezeigt, bei welchem die Software eingesetzt werden soll. Anschließend werden anhand dieser Szenarios Anforderungen definiert, die von der Software erfüllt werden müssen.

3.1 Anwendungsszenario

Will ein Benutzer gegen den KI Client spielen, kann er über das Menü auf dem Touch Monitor den KI Algorithmus auswählen und diesen herrausfordern. Dazu kann er, falls er mit seinem Smarthphone spielen will einen QR-Code auf dem Monitor aufrufen, diesen einscannen und dann das Spiel beginnen. Andernfalls kann der Benutzer auch direkt am Monitor das

Spiel starten und auf diesen mittels Touch Züge ausführen. Die KI Algorithmen welche auf dem KI Client implementiert sind haben eine ELO Zahl hinterlegt, welche Auskunft über die Spielstärke gibt. Fühlt sich der Benutzer also über oder unterfordert kann er den geeigneten Gegner auswählen. Gibt es mehrere Benutzer welche gegen einander spielen wollen, haben diese die Möglichkeit entweder abwechselnd auf dem Monitor, oder beide mittels QR-Code über ihr Smartphone ein Spiel zu starten. Will ein Benutzer zwei KI's beim spielen beobachten, so kann er diese am Monitor auswählen und diese gegeneinander Antreten lassen.

3.2 Anforderungen an die Software

Aus dem oben beschriebenen Anwendungsszenario lassen sich konkrete Anforderungen ableiten, die für die Software von Relevanz sind. Hierbei wird zwischen funktionalen und nicht funktionalen Anforderungen unterschieden [Bal09].

3.2.1 Funktionale Anforderungen

- **/F10/** *Menü zum Auswählen des Spieles:* Die Graphische Oberfläche soll dem Benutzer die Möglichkeit geben das ein Spiel auszuwählen. Dazu soll es ein Menü geben welches die möglichen Spiele, wie z.B. Reversi oder Dame zur Auswahl stellt.
- **/F11/** *Auswählbares Menü für verschiedene KI Algorithmen:* Die Applikation muss ein leicht bedienbares Graphisches Interface bieten, bei welchem verschiedene künstliche Intelligenz Algorithmuen ausgewählt und herrausgefordert und werden können.
- **/F12/** *QR-Code fürs verbinden mit dem Smartphone:* Um einen unkomplizierten Verbindungsaufbau vom Raspberry mit dem Smarthphone zu gewährleisten, soll es eine Menü-Option geben, bei der ein QR-Code angezeigt wird. Nach dem scannen des QR-Codes soll eine Verbindung Aufgebaut werden, welche bis zum Beenden bestehen bleibt.
- **/F13/** *Varaible Zeiteinstellung im Menü:* Dem Benutzer soll es möglich sein über das Menü eine Zeit einstellen zu können, welche jeder Spieler im Spiel zur Verfügung für seine Züge hat. Als Spieler können entweder Benutzer oder KI Clients agieren.
- **/F20/** *Benutzer soll Züge ausführen können:* Der Benutzer soll in der Lage sein, Züge gegen die KI spielen zu können. Dazu soll er entweder direkt über den Touch-Monitor oder über das Smartphone eine Eingabemöglichkeit haben. Diese soll den Momentanzustand des Spielbrettes zeigen, wodurch der Benutzer eine Entscheidung für seinen nächsten Zug treffen und diese über eine Touch-Berührung ausführen kann.
- **/F21/** *Benutzer sollen gegen andere Benutzer spielen können:* Für mehrere Benutzer soll es möglich sein, gegeneinander spielen zu können. Dazu sollen sie entweder den Touch-Monitor verwenden indem sie abwechselnd Züge ausführen, oder beide jeweils ein Smarthphone.
- **/F22/** *Der Benutzer kann KI's gegeneinander spielen lassen:* Der Benutzer soll in der

Lage sein zwei KI-Algorithemen auswählen zu können um diese gegeneinander spielen zu lassen. Damit man dieses Spiel sehen zu kann, sollen alle Züge die von beiden getätigt werden auf dem Spielbrett des Touch-Monitors angezeigt werden.

- **/F30/** *Eine ELO Zahl soll die Spielstärke der Algorithmen angeben:* Damit der Benutzer eine für sich angemessene Herrausforderung findet, sollen schwächere und stärkere KI-Algorithmen in der GUI gekennzeichnet werden. Um eine genaue Kennzahl für die Stärke zu erhalten, werden die ELO Zahlen durch Simulationen berechnet. Diese Simulationen sind Spiele der Algorithemen untereinander.

3.2.2 Nichtfunktionale Anforderungen

- **/Q10/** *Robustheit der Smarthphone Verbindung:* Nach der Verbindung mit dem Smarthphone (mittels QR-Code /F12/) muss sichergestellt sein, dass die Verbindung nicht ohne Grund abbricht, sondern erst wenn z.B. die Distanz zwischen Smartphone und Pi zu groß ist. Desweitern soll nach einem Verbindungsabbruch, das Spiel nicht abbgebrochen werden, sondern es soll eine Möglichkeit zum wiederverbinden bestehen.
- **/Q20/** *ELO Zahlen sollen Stärke wiederspiegeln:* Die durch die Simulationen errechnete ELO Zahl von /F30/ soll auch in etwa dem Stärkegrad der Algorithmen entsprechen. Hat ein Algorithmus die sehr viel mehr ELO muss er auch dementsprechend stärker sein.
- **/Q30/** *Zeiteinstellung soll von der KI Berücksichtigt werden:* Die Zeiteinstellung von /F13/ soll vom KI Client als Berechnungsdauer genutzt werden. Dieser soll dabei seine Rechenzeit so gut wie möglich an die Zeiteinstellung anpassen.
- **/Q40/** *Reaktionszeit des Touch-Interfaces:* Das Berühren des Touch-Monitors soll zur sofortigen Ausführung des Befehls der Software führen.
- **/Q50/** *Mobile Responsiveness:* Das Spielfeld soll auf egal welchem verbundenen Smarthphone gleich skaliert aussehen. Das verwenden von Tablets, oder das Drehen des Gerätes soll keinen Einfluss auf die Darstellung des Spielbrettes haben.

3.2.3 Zusammenfassung der Anforderungen

Die Identifizierten Funktionalen und Nichtfunktionalen Anforderungen werden in der Tabelle 1 zusammengefasst. Die Kürzel sind für die folgenden Kapietel von Relevanz da sie in diesen Referenziert werden.

ID	**Funktionale Anforderung**
/F10/	Menü zum Auswählen des Spieles
/F11/	Auswählbares Menü für verschiedene KI Algorithmen
/F12/	QR-Code fürs verbinden mit dem Smartphone
/F13/	Varaible Zeiteinstellung im Menü
/F20/	Benutzer soll Züge ausführen können
/F21/	Benutzer sollen gegen andere Benutzer spielen können
/F22/	Der Benutzer kann KI's gegeneinander spielen lassen
/F30/	Eine ELO Zahl soll die Spielstärke der Algorithmen angeben
ID	**Nichtfunktionale Anforderung**
/Q10/	Robustheit der Smarthphone Verbindung
/Q20/	ELO Zahlen sollen Stärke wiederspiegeln
/Q30/	Zeiteinstellung soll von der KI Berücksichtigt werden
/Q40/	Reaktionszeit des Touch-Interfaces
/Q50/	Mobile Responsiveness

Tabelle 1: Anforderungstabelle

4 Architektur der Software

4.1 Überblick

4.2 Gameserver

4.3 ReversiXT GUI

4.4 KI Client

5 Hardware

Dieses Kapitel handelt von der verwendeten Hardware auf der die Software zum läuft. Die Software wird auf einem Raspberry Pi, welcher an einem Touch Monitor anschlossen ist ausgeführt.

5.1 Raspberry Pi

5.2 Touch Monitor

6 Imlementierung

6.1 Eingesetzte Sofwarekomponenten

6.1.1 Programmiersprachen und Frameworks

6.1.2 Datentransferprotokolle

6.2 Gameserver

6.2.1 Netzwerkspezifikation des Gameservers

6.3 Graphische Oberfläche

6.3.1 Gegebene React Anwendung

6.3.2 Erweiterungen

6.4 KI Client

6.4.1 Vergleich der KI Algorithmen

7 Testing

7.1 Integrationstest

7.2 Ergebnisse

8 Fazit und Ausblick

Literaturverzeichnis

[Bal09] Helmut Balzert. *Lehrbuch der Softwaretechnik: Basiskonzepte und Requirements Engineering (German Edition)*. Spektrum Akademischer Verlag, 2009.

[MP19] Kevin Ferguson Max Pumperla. *Deep Learning and the Game of Go*. Manning, 2019.

[Rus12] Stuart Russell. *Künstliche Intelligenz ein moderner Ansatz*. Pearson, 2012.

Quellenverzeichnis

[Dra] *Mindsports*. Die Geschichte von Dame. URL: http://www.mindsports.nl/index.php/arena/draughts/478-the-history-of-draughts.

[Int] *World Draughts Federation*. official FMJD rules for international draughts 100. URL: http://www.fmjd.org/?p=v-100.

Anhang

Inhalt des beigefügten Datenträgers:

- ...
- ...

A Domändenmodell

Ein toller Anhang, der nicht nur als „*Müllhalde*“ genutzt wird, sondern in dem Bilder und Inhalte auch mit eigenen Worten erklärt werden und den man auch für sich alleine lesen kann. Es sollten auch Referenzen auf die zugehörige ausführliche Behandlung im Hauptteil inklusive Seitenangabe mit `\pageref` gegeben werden.

Screenshot

Unterkategorie, die nicht im Inhaltsverzeichnis auftaucht.