



Entwicklung einer künstlichen Intelligenz für das Spiel “Schach” in Python

Studienarbeit

des Studiengangs Angewandte Informatik
an der Dualen Hochschule Baden-Württemberg Mannheim

von

Lars Dittert & Pascal Schröder

29.04.2019

Bearbeitungszeitraum
Matrikelnummer, Kurs
Ausbildungsfirma
Betreuer

17.09.2018 - 29.04.2019
5388171 & 5501463, TINF16AI-BC
IBM Deutschland GmbH, Mannheim
Prof. Dr. Karl Stroetmann

Unterschrift Betreuer

Inhaltsverzeichnis

Erklärung der akademischen Aufrichtigkeit

Wir versichern hiermit, dass wir unsere Studienarbeit mit dem Thema:

Entwicklung einer künstlichen Intelligenz für das Spiel "Schach" in Python

selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt haben.

Wir versichern zudem, dass die eingereichte elektronische Fassung mit der gedruckten Fassung übereinstimmt.*

* falls beide Fassungen gefordert sind

Mannheim, 29.04.2019

Ort, Datum

Unterschrift Lars Dittert & Pascal Schröder



Abstract

1 Einleitung

1.1 Zweck und Ziel dieser Arbeit

1.2 Kriterienformulierung

2 Theoretische Hintergründe

2.1 Spieltheorie

2.2 Verwendete Methoden

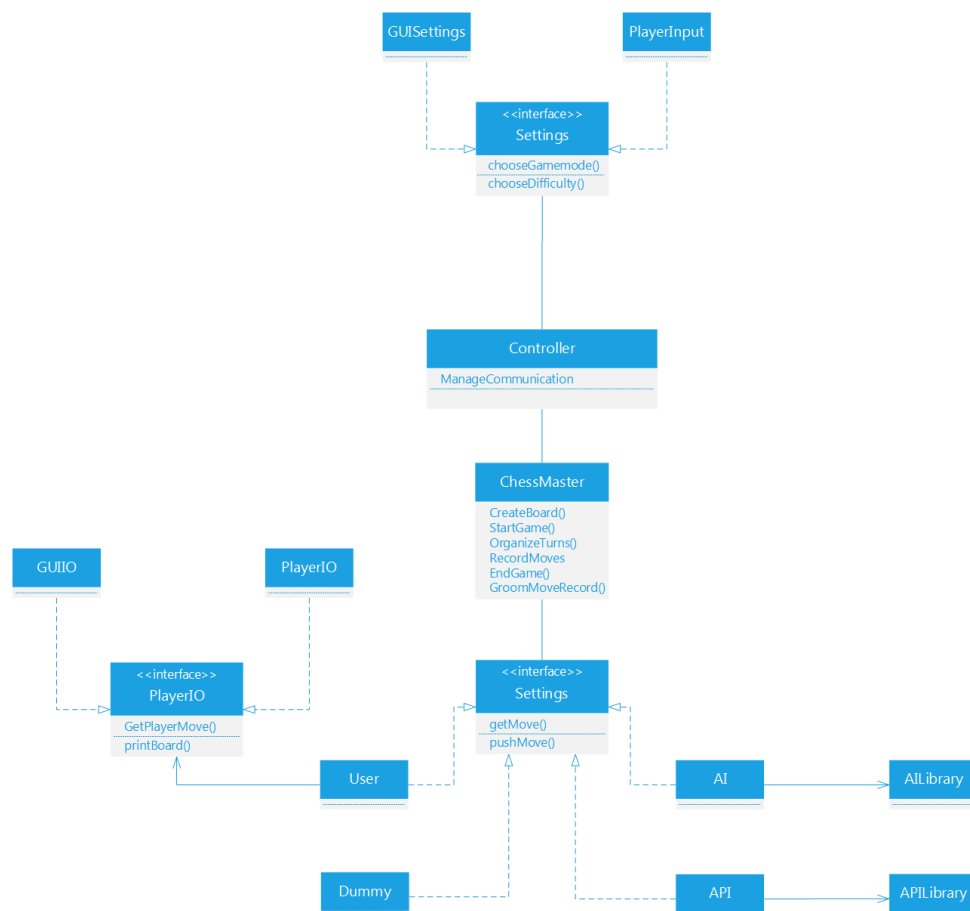
3 Technische Grundlagen

3.1 Verwendete Bibliotheken

3.2 Architektur

Um die Ziele der Erweiterbarkeit, Wartbarkeit und einfachen Verständlichkeit zu erfüllen, ist eine modulare Aufbauweise des Projektes von Nöten. Die auf diesen Prinzipien beruhende Architektur wird im folgenden Kapitel beschrieben.

Zunächst wurde das Programm der Aufgaben entsprechend in einzelne Klassen unterteilt, die verschiedene Aufgaben zugewiesen bekommen haben. Diese Aufteilung kann Figur 3.2.1 entnommen werden.



Figur 3.2.1 Klassendiagramm Architektur ChessAI

Als zentrales, Verbindungselement steht zunächst der “Controller”. Dieser übernimmt die Abfrage aller Optionen, falls diese nicht beim Start des Programms mit übermittelt werden. Zudem initialisiert der Controller die Spielteilnehmer sowie den “ChessMaster”. Bei letzterem übergibt der Controller diesem die initialisierten Spielteilnehmer. Die Aufgaben des “ChessMaster” wird später genauer beschrieben.

Zur Abfrage der Optionen steht dem Controller eine ‘Settings’ Schnittstelle zur Verfügung. Diese wird für Eingaben des Nutzers einerseits über eine Konsole als auch optional über eine grafische Benutzeroberfläche implementiert. Dabei wird abgefragt, welchen Typ die jeweiligen Spielteilnehmer annehmen sollen und es können zudem zusätzliche Optionen für die einzelnen Spieler definiert werden. Als Beispiel kann für Spieler 1 der Typ “User” gewählt werden und für Spieler 2 der Typ “AI”. Dies ermöglicht ein Spiel des Nutzers gegen die im Rahmen dieser Arbeit entwickelte KI. Für die KI wird darauf folgend noch der Schwierigkeitsgrad abgefragt. Zusätzlich kann für jeden Spieler ein Name festgelegt werden.

Die Aufgaben des “ChessMaster” erstrecken sich über die Verwaltung des Schachspiels an sich, das Ansprechen der jeweiligen Spieler zum Ermitteln ihrer Züge sowie dem Durchführen des gewählten Spielzugs auf dem Schachbrett. Zusätzlich speichert es jedes Schachbrett, das sich im Laufe des Spiels ergibt, und fügt diese zur “board_history.” Datei hinzu. Diese speichert alle Schachbretter gemeinsam mit einem numerischen Wert. Dieser gibt Aufschluss über Erfolgsaussichten der jeweiligen Akteure des Spiels. Dazu wird nach jedem Spiel zu dem entsprechenden Eintrag in der Datei eine eins zu dem alten Wert aufaddiert, wenn der Spieler der weißen Figuren das Spiel gewonnen hat und eine eins subtrahiert, wenn der Spieler der schwarzen Figuren das Spiel gewonnen hat. Bei einem Unentschieden bleibt der alte Wert bestehen. Dies hilft der KI bei der Bewertung eines Schachbretts unter zur Hilfenahme von statistischen Werten.

Dieser spricht die vom Controller erstellten Spieler an. Diese können, wie bereits angedeutet, von verschiedenen Typen sein. Zur Auswahl stehen

- User - Ein menschlicher Akteur kann Züge über eine Nutzerschnittstelle eingeben
- AI - Die künstliche Intelligenz versucht den best möglichen Zug zu berechnen
- Dummy - Ein zufälliger Zug wird ausgewählt
- (Optional) API - Ein Zug wird über eine Schnittstelle zu einer Online Schachplattform, auf der menschliche sowie künstliche Spieler teilnehmen dürfen, bestimmt

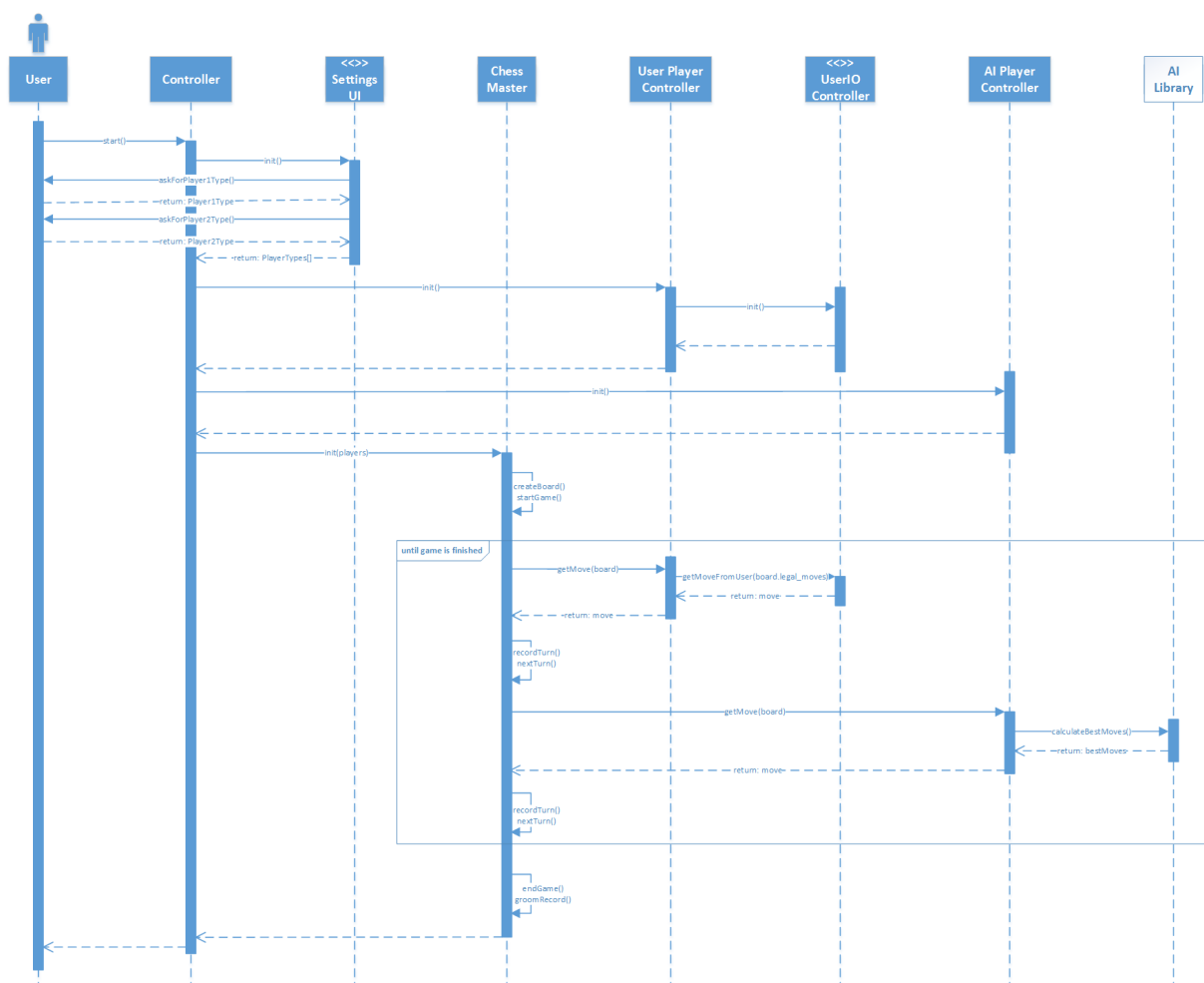
Dazu existiert eine “Player” Schnittstelle, die für jeden dieser Spieler implementiert werden muss.

Die “User” Implementation greift dabei zur Ermittlung des Zuges auf eine “PlayerIO” Schnittstelle zurück. Diese wiederum kann - ähnlich zur “SettingsUI” Schnittstelle - sowohl für Eingaben über eine Konsole als auch für Eingaben über eine grafische Benutzeroberfläche implementiert werden.

Die “AI” sowie die “API” Implementation greifen für die Ermittlungen ihrer Züge nochmals auf eigens erstellte Bibliotheken zurück, die elementare Funktionen erhalten.

Der genaue Ablauf der Ermittlung der Züge sowie der andere Operationen des Programmes werden in Kapitel XY.ZY näher erläutert.

In Figur 3.2.2 ist der sequentielle Ablauf der Funktionsaufrufe erkennbar.



Figur 3.2.2 Sequenzdiagramm Architektur ChessAI

Dabei wird zunächst über die “Settings” Schnittstelle nach den Spielertypen sowie Namen gefragt. initialisiert der “Controller” die Spieler, in diesem Fall einerseits den “User”, der wiederum die “UserIO” Schnittstelle implementiert, und außerdem den “AI” Player.

Darauffolgend spricht der “Controller” den “ChessMaster” an. Dieser startet nun das Spiel und fragt solange wie das Spiel nicht vorbei ist immer abwechselnd erst zu Spieler 1 - dem

Nutzer - und dann zu Spieler 2 - der KI - nach dem nächste Zug. Dabei übergibt der “ChessMaster” stets das aktuelle Schachbrett. Der Nutzer ermittelt den durchzuführenden Zug über eine Abfrage an den Nutzer über die entsprechende Schnittstelle, der “AI” Spieler, indem dieser Funktionen aus der entsprechenden Bibliothek zur Hilfe nimmt.

Nach jedem Zug fügt der “ChessMaster” diesen zum Schachbrett hinzu und speichert dieses in einen Spielverlauf. Nach Ende des Spiels wird dieser in das “board_history.csv” eingepflegt wie weiter oben bei der Beschreibung der Klasse bereits erläutert.

Abschließend beendet der “ChessMaster” das Spiel, wenn keine Revanche gewünscht ist, und so wird auch das Programm im Anschluss daran geschlossen.

4 Implementation

5 Evaluation

5.1 Kriterienerfüllung

5.2 Einordnung Intelligenz

Abkürzungsverzeichnis

Literatur