Meilenstein 1 - Dummy-KI

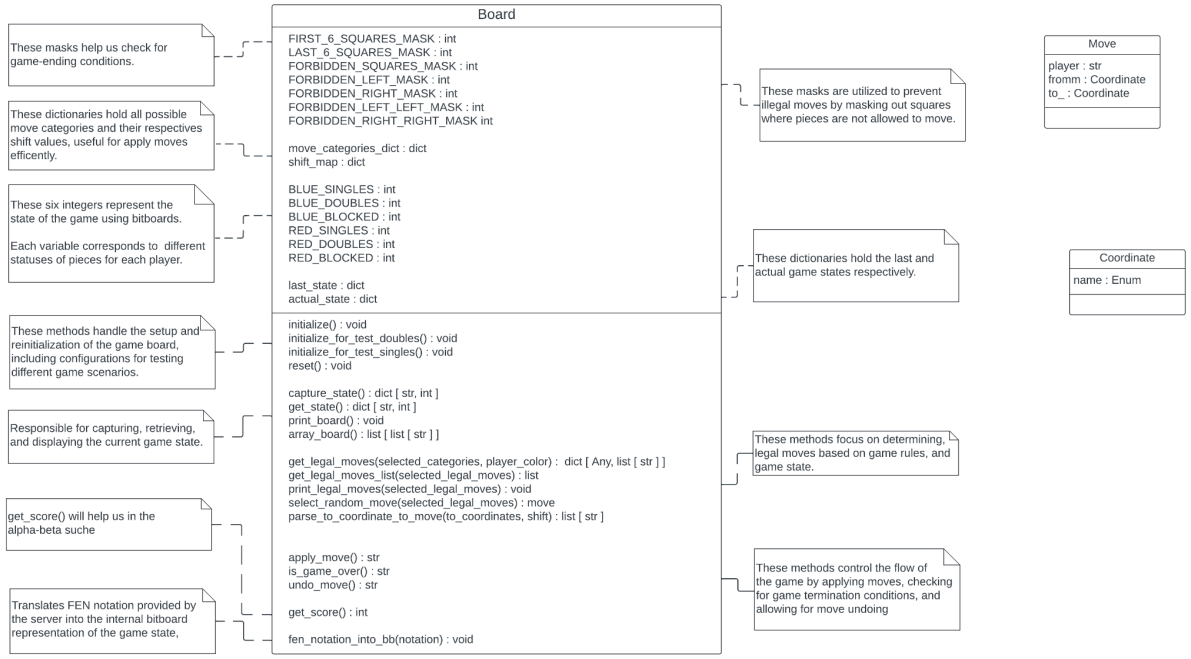
Gruppe AC
Gruppenname: AC/DC

Implementierungssprache: Python

Mitglieder (von links nach rechts):
Amar Salcinovic
Carlos Andres Sulbaran Fandino
Peter Wladislaw Urbansky



Klassendiagramm:



Benchmarks für die KI:

Prozessor:	Apple M2, 8-Core CPU 8-Core GPU		
RAM:	8GB		
Programmiersprache:	Python		
Zustand Repräsentation:	BitBoards(64 Bits)		
FEN:	b0b0b0b0b0b0/1b0b0b0b0b0b01/8/8/8/8/1r0r0r0r0r0r01/r0r0r0r0r0r0 b	b01bbb01b0/1b02b03/3bbr01b01/8/3rr1b0b01/8/2r01r01rr1/r0r0r01r01 r	2b02bb/1bb2b03/5bb2/8/1r03r02/6r01/8/r01r01rrr0 b
Wiederholungen:	100000	100000	100000
Durchschnittliche Laufzeit pro Wiederholung:	0.001165165901ms ~0.0012ms	0.001233427524ms ~0.0012ms	0.001183223724ms ~0.0012ms

Gesamtlaufzeit:	128.5009384155ms	136.0149383544ms	130.8810710906ms

Aussicht für Meilenstein 2:

Mit den bisherigen Benchmarks sind wir sehr zufrieden. Die einzige Kritik, die wir momentan als Diskussion haben, ist die Umstrukturierung und Vereinfachung des Codes. Das Programm generiert die Bitboards und interagiert erfolgreich mit dem Spielbrett.
Momentan haben wir alle möglichen Züge in folgende Kategorien unterteilt:

```
'singles_left_empty': True,
'singles_front_empty': True,
'singles_right_empty': True,
'singles_kill_left_singles': True,
'singles_kill_left_doubles': True,
'singles_kill_right_singles': True,
'singles_kill_right_doubles': True,
'singles_upgrade_left': True,
'singles_upgrade_front': True,
'singles_upgrade_right': True,
'doubles_l_l_f_empty': True,
'doubles_f_f_l_empty': True,
'doubles_f_f_r_empty': True,
'doubles_r_r_f_empty': True,
'doubles_kill_l_l_f_singles': True,
'doubles_kill_l_l_f_doubles': True,
'doubles_kill_f_f_l_singles': True,
'doubles_kill_f_f_l_doubles': True,
'doubles_kill_f_f_r_singles': True,
'doubles_kill_f_f_r_doubles': True,
'doubles_kill_r_r_f_singles': True,
'doubles_kill_r_r_f_doubles': True,
'doubles_l_l_f_singles': True,
'doubles_f_f_l_singles': True,
'doubles_f_f_r_singles': True,
'doubles_r_r_f_singles': True
```

Damit haben wir bereits einen Ansatz für unsere Bewertungsfunktion für den späteren Entscheidungsbaum. Wir wollen versuchen, im allgemeinen Bewegungen, die ein Stein

Upgrade beinhalten, vor einfachen Zügen zu priorisieren. Ebenfalls sollen Kills, die ein Upgrade des steins beinhalten, vor normalen Kills zu priorisieren.
Wir haben einige weitere Möglichkeiten in Betracht gezogen, um die Entscheidungen zu verfeinern, konnten uns jedoch bisher nicht endgültig entscheiden.

Erste Ideen beinhalten, dass wir als Strategie ein paar unserer Spielsteine als “Könige” auswählen. Diese sollten um jeden Preis von den restlichen Steinen beschützt und ins Ziel gebracht werden.

Pro:

- Vorteil dieser Strategie ist, dass Züge eindeutiger werden können, da ein klares Ziel besteht.
- Strategie wird weniger Global und mehr Lokale, dadurch können Entscheidungsbäume gekürzt werden.

Con:

- Steine, die in guter Position sind selbst ins Ziel zu kommen, könnten sich aufopfern für “Königliche Steine” die sich in schlechterer position befinden, wodurch ein Sieg verschenkt werden könnte
- Durch eher lokale Strategie, könnten lange Taktiken verloren gehen

Zwischendurch gab es die Überlegung, Schwarmintelligenz zu nutzen, um das Spiel zu spielen. Jeder Stein soll dabei für sich selbst denken und seine eigene Position kennen, nicht aber die Positionen aller anderen Steine. Jeder Stein-Agent würde mit sensorischen Fähigkeiten ausgestattet werden, um lokale Gegner und Freunde wahrzunehmen. Angelehnt an den Ameisenalgorithmus, gab es Überlegungen, mit “Hormonen” zu arbeiten, wodurch jeder Stein einen Fingerabdruck auf allen Koordinaten hinterließ, auf denen sich der Stein bereits bewegt hat. Ähnliche Überlegungen hätten den Steinen Fähigkeiten gegeben, wie ein "Hilfeschrei" der nahestehenden Freunden vermitteln würde, dass Verstärkung gebraucht wird oder der Stein geschlagen wurde.
Die Steine sollen in diesem Konzept alle Wahrscheinlichkeit zum Sieg berechnen, für die Züge, die sie selbst machen können, und würden Ihre Wahrscheinlichkeiten einem Spieler-Agenten übergeben, der am Ende eine Tabelle an Wahrscheinlichkeiten vor sich sieht und nur noch die beste Ausführen muss.
In Verfeinerungen könnte der Agent natürlich selbst noch Überlegungen anstellen, und seine Entscheidung mit den Entscheidungen der Stein-Agenten abgleichen, um einen “perfekten” Spielzug zu machen.

Pro:

- Mit richtiger implementierung, könnte der Entscheidungsbaum in viele kleine Entscheidungsbäume aufgebrochen werden, die in Ihrer Summe massiv Äste abschneiden könnten und somit Speichereffizient wären
- Strategien wären weniger global und mehr lokal, wodurch speicher gespart werden könnte

Con:

- Durch eher lokale Strategie, könnten lange Taktiken verloren gehen
- Implementierung des Codes wäre komplexer und somit auch ein komplexeres Debugging beinhalten, wodurch der ganze Code fehleranfälliger wäre

Eine Variation beider vorheriger Strategien bestand daraus, dass Steine von Anfang an in Kompanien unterteilt werden sollten. Somit würden z.B. vier Steine von Anfang an versuchen zusammenzuarbeiten und sich nur auf Ihren Sieg konzentrieren. Damit hätten wir sowohl den Ansatz einer Schwarmintelligenz und der einhergehenden Speichereffizienz, als auch klare Ziele, die der Algorithmus erreichen kann, ohne sich im Detail des ganzen Spielbretts zu verlieren. Natürlich ist auch hier der Nachteil der Komplexität des Codes im Vordergrund, der schnell zum Untergang des ganzen Projekts führen kann.

Momentan tendieren wir eher dazu, die KI so einfach wie möglich zu halten, um auch effektiv daran arbeiten zu können. In den folgenden Wochen wird sich entscheiden, ob und welche Elemente wir aus unseren Überlegungen in die KI implementieren werden. Für die Bewertungsfunktion haben wir uns für folgende Kriterien entschieden, die jedoch nicht unbedingt im zweiten Meilenstein implementiert werden. Als erstes Bewertungskriterium haben wir uns für einen Wert entschieden, der sich durch die Summe aller verfügbaren Steine berechnen lässt. Dabei hat Beispielsweise ein blauer Stein, der über einen roten Stein steht und somit den roten Stein bewegungsunfähig macht, einen höheren Einfluss auf den Wert als ein normaler Stein. Ein weiteres Kriterium wäre beispielsweise die Auswirkung eines möglichen Zuges, wie einem capture oder dem Blockieren eines Steines. Wo wir noch entscheiden müssen, welche Auswirkung wie gepunktet wird. Man könnte auch schauen, dass gewisse Figuren an gewissen Positionen eine höhere Chance auf den Sieg erzielen als andere. Das könnte ebenfalls Einfluss auf die Bewertung haben. Die Anzahl der möglichen Züge ist ebenfalls ein gutes Kriterium für die Bewertung. Je mehr Züge ich habe, desto stärker bin ich positioniert. Man könnte auch schauen wie Figuren zueinander stehen, zu Mal es in diesem Game möglich ist ein Pferd zu kreieren in dem eine nebenstehende Figur auf die andere geht. Das wären so die ersten Bewertungskriterien für die Bewertungsfunktion über die wir uns Gedanken gemacht haben.

Der Termin für den zweiten Meilenstein wäre am 23.05.24.