VierGewinnt - README

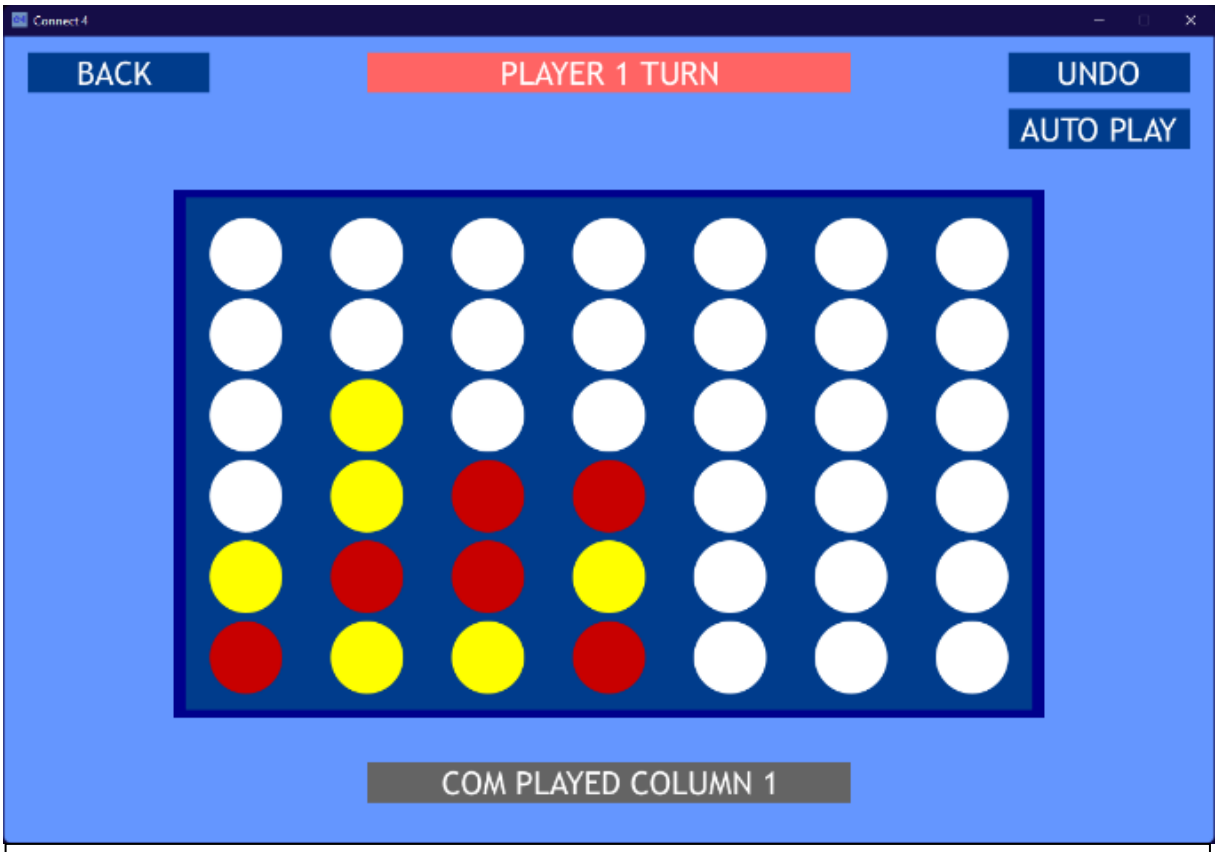
Izzet Ekicioglu, 5337587

VierGewinnt

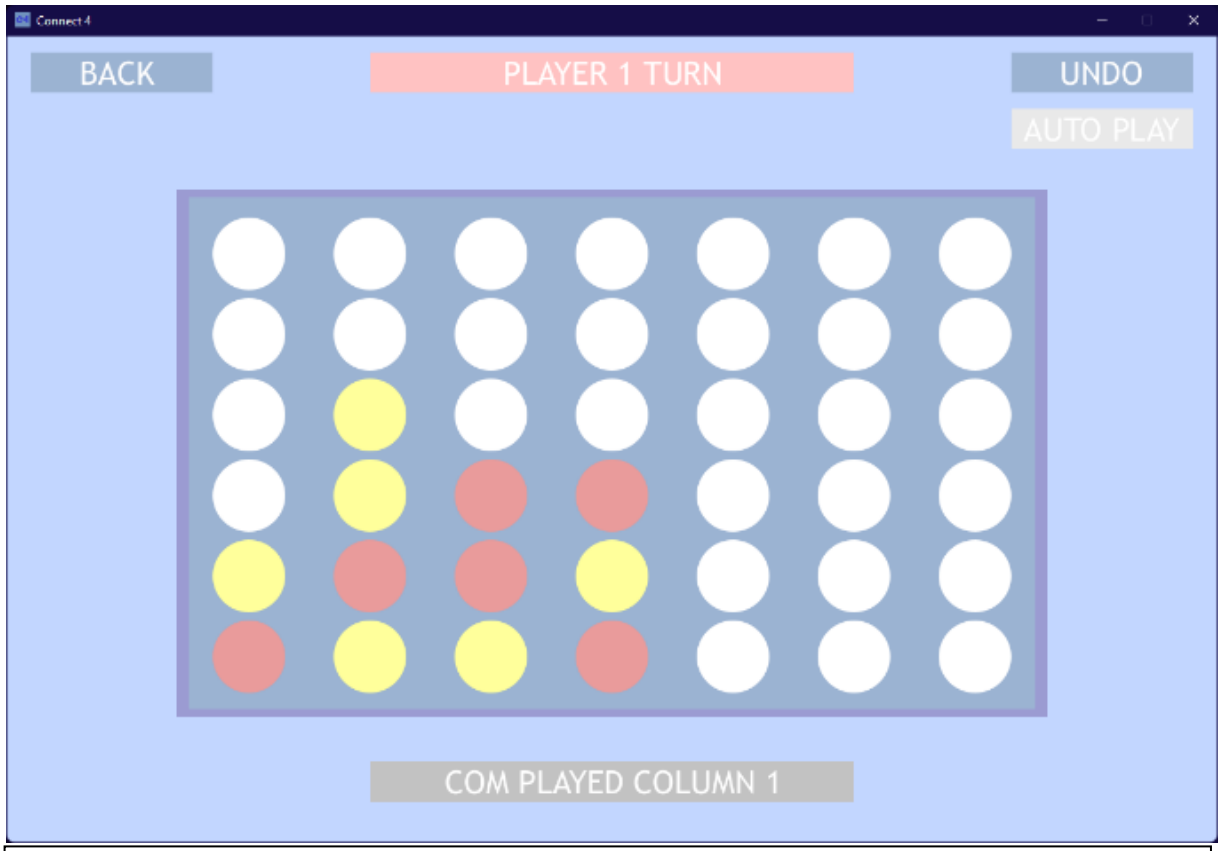
Im Rahmen der Projektarbeit für das Modul „CS1016 Programmierung interaktiver Systeme SoSe2022“ soll das Spiel „Vier gewinnt“ programmiert und anschließend ein Algorithmus implementiert werden, der den bestmöglichen Zug berechnet und somit das Spiel Mensch gegen Computer ermöglicht.

Im Spiel „Vier gewinnt“ müssen zwei Spieler abwechselnd Kügelchen in ihrer (spielerbezogenen) Farbe in ein Feld bestehend aus sieben Spalten mit je sechs Zeilen einwerfen. Dabei fallen die Kügelchen durch die vom Zugspieler gewählte Spalte, bis diese entweder auf dem Boden der Spalte (unterste Zeile) oder auf einer anderen, bereits in einem vorherigen Zug in diese Spalte eingeworfenen Kugel, landen. Gleichzeitig folgt daraus, dass in einer Spalte maximal sechs Kügelchen liegen dürfen.

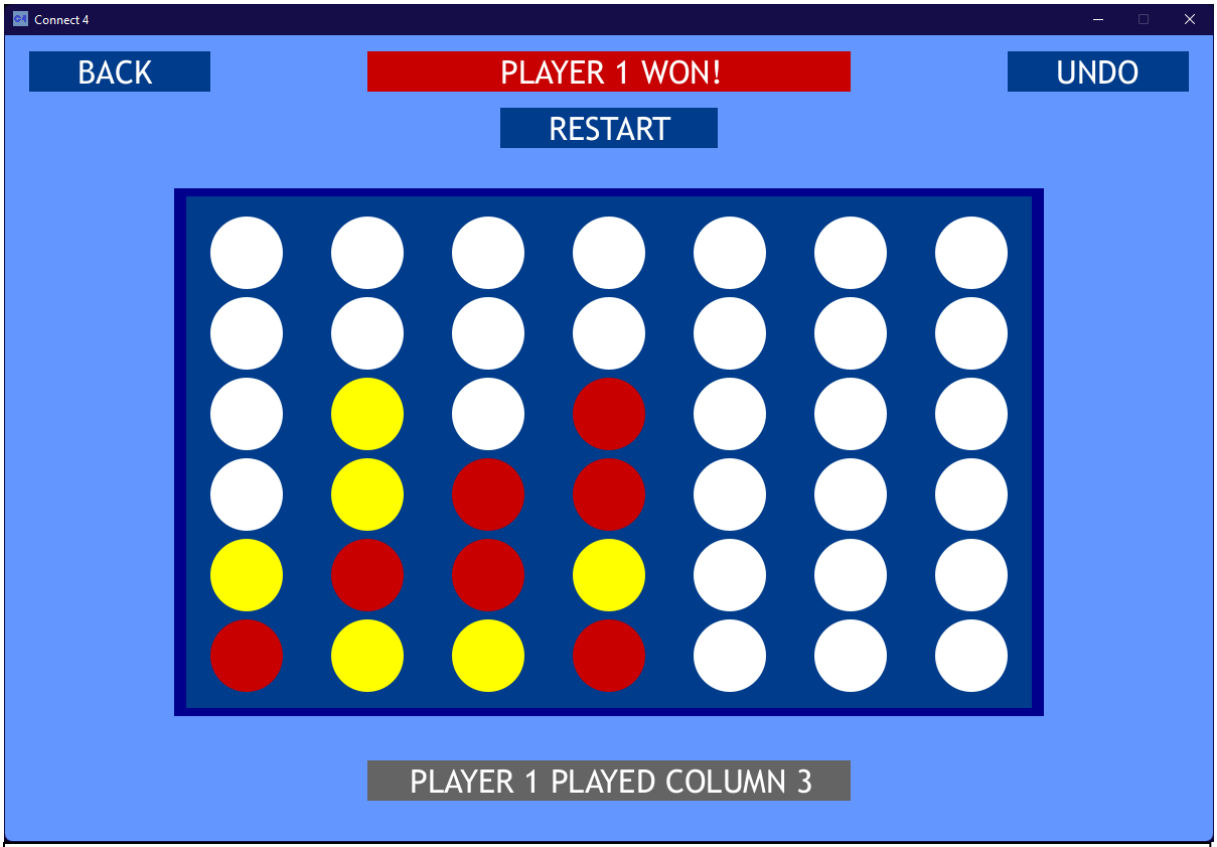
Das Spiel ist zu Ende, wenn ein Spieler gewinnt oder das Spielfeld voll ist (in allen Spalten liegen sechs Kügelchen), sodass keine weiteren Züge mehr möglich sind. Gewonnen hat der Spieler, der es als erster schafft, dass vier Kügelchen seiner Farbe im Feld entweder vertikal, horizontal oder diagonal nebeneinander liegen. Ist das Spielfeld voll und hat kein Spieler die Gewinnsituation erreicht, so endet das Spiel mit einem unentschieden.



[Spielfeld während des Spielverlaufs]



[Bild während Computer Berechnungen durchführt]



[Gewinnsituation nach der Berechnung des Computers]

Funktionsweise des Algorithmus

Um die Berechnung des bestmöglichen Spielzugs ermöglichen zu können, greift das hier vorgestellte Programm auf eine Kombination aus dem MinMax- und dem MonteCarlo-Algorithmus zurück.

Hierbei wird zunächst für alle möglichen Züge (maximal sieben Züge für sieben Spalten) aus einer Spielsituation heraus der MinMax-Algorithmus aufgerufen. Dieser berechnet hier für die nächsten drei Züge alle möglichen Spielverläufe.

In der Methode bestMove() wird in einem Stream das aktuelle Feld (parallel, um die Effizienz zu erhöhen) mit allen aktuell möglichen Spielzügen bespielt. Die bespielten Felder werden anschließend in die min()-Methode übergeben, der die möglichen Spielzüge des Gegners in der neuen Spielsituation simuliert. Wiederrum werden für die in der min()-Methode erstellten möglichen Spielzüge jeweils die max()-Methode aufgerufen, um entsprechend die Züge des aktuellen Spielers zu simulieren. Dieser rekursive Aufruf der beiden Methoden geht soweit, bis entweder eine Gewinnposition für einen der beiden Spieler ein Spielausgang mit unentschieden (Spielfeld voll, aber keine Gewinnsituation) gefunden wurde oder die gewünschte Rekursionstiefe, hier festgelegt auf 2, erreicht wurde.

Die Rückgabewerte der min()- und der max()-Methoden sind entsprechend der Spielausgänge, und ermöglichen so, den bestmöglichen Spielzug zu bestimmen. Falls die Bewertung ein Gewinnzug für den aktuellen Spieler ermittelt, so wird der Wert 100 zurückgegeben, für den Gegner hingegen der Wert 0. Liegt ein unentschieden vor, so wird der Wert 50 übergeben. Aufgrund des gegenseitig rekursiven Aufrufs der min()- und der max()-Methoden, werden Spielzüge, die unerreichbar wären, sofern beide Spieler die für sie selbst bestmöglichen Züge spielen, aussortiert. So wird bei der max()-Methode nach der maximalen Bewertung gesucht und diese wiederrum an die min()-Methode übergeben, die hingegen nach dem kleinstmöglichen Wert sucht. So wird ist der Rückgabewert der min()-Methode entsprechend der schlecht-möglichste Zug aus der max()-Methode. Sollte also z.B. für einen Spielzug aus der min()-Methode die max()-Methode den Wert 100 (Gewinn für aktuellen Spieler) zurückgeben und für einen weiteren den Wert 50 (unentschieden), so übergibt die Methode den kleineren Wert zurück, da der simulierte Gegner einen Gewinn des aktuellen Spielers vermeiden möchte.

Ist die gewünschte Rekursionstiefe erreicht und es liegt weder eine Gewinnsituation noch ein unentschieden vor, so wird die Berechnung des bestmöglichen Zuges anhand des MonteCarlo-Algorithmus berechnet. Der Sinn hinter diesem Algorithmus ist es, anhand vieler, bis zum Ende durchgespielter Spielverläufe, statistisch zu entscheiden, welcher Zug die höchste Gewinnwahrscheinlichkeit besitzt.

Aus der Bewertungs-Methode des MinMax-Algorithmus wird die monteCarlo()-Methode aufgerufen, die entsprechend der vorher festgelegten Anzahl der zu berechnenden Spielverläufe (festgelegt durch den Schwierigkeitsgrad), parallel zueinander die ergänzende Methode monteCarloCalc() aufruft. Diese Methode ruft sich bis zu einem beliebigen Spielende rekursiv, mit veränderten Parametern (abwechselnde Spieler und stets neu ergänztes Spielfeld) mit zufälligen Zügen selbst auf. Sobald ein Spielende vorliegt, wird an die ursprüngliche Methode monteCarlo() entsprechend des Spielendes ein boolescher Wert zurückgegeben. Sofern der aktuelle Spieler für einen Spielverlauf gewonnen hat, wird true zurückgegeben, andernfalls false. Die monteCarlo()-Methode

hingegen inkrementiert für jeden erhaltenen true-Wert die entsprechende Position in einem vorher definiertem Array, welches die möglichen Ausgangszüge repräsentiert.

Nach erfolgreichem Durchlauf der Spielverläufe liegt ein Array vor, welches für die jeweiligen Ausgangszüge die Anzahl der resultierten Gewinne enthält. Um eine Bewertung wieder an den MinMax-Algorithmus zu übergeben, wird die gesamte Gewinnwahrscheinlichkeit berechnet. Hierfür wird lediglich die Summe der Gewinne durch die Anzahl der gesamt berechneten Spielverläufe dividiert und anschließend mit 100 multipliziert.

Ähnlich verläuft es in der Ausgangsmethode bestMove(). Dieser speichert ebenfalls die Rückgabewerte des MinMax-Algorithmus in einem Array, welches die möglichen Spielzüge darstellt. Am Ende wird hierbei der Index des größten Wertes zu spielende Spalte im VierGewinnt-Spiel ausgewählt.

Quellen

- Tobias Nopper, „Prinzip MinMax-Algorithmus am Beispiel von Tic Tac Toe | Wie spielen Computer?“ (YouTube)
    - https://www.youtube.com/watch?v=3ufcmCpKb6w (Stand: 14.07.2022)
- Prof. Dr. Andreas Gogol-Döring, Algorithmen und Datenstrukturen, „Minimax Varianten“ (YouTube)
    - https://www.youtube.com/watch?v=FH29YqB7580 (Stand: 14.07.2022)
- Prof. Dr.-Ing. Dominikus Herzberg, Vorlesungsmaterialien PiS SoSe2022
    - Folien und Podcast Vorlesungen