Übungen zur Vorlesung
Grundlagen der Künstlichen Intelligenz
Sommersemester 2024

Institut für Informatik 4
Priv.-Doz. Dr. V. Steinhage
Friedrich-Hirzebruch-Allee 8
53115 Bonn
Email: steinhage@cs.uni-bonn.de
WWW: http://net.cs.uni-bonn.de/ivs/

Blatt 3 (10 Punkte)

Abgabe durch Hochladen (nur PDF-Format bzw. Python-Code) auf der eCampus-Seite bis **Sonntag, 28.04.2024, 12:00 Uhr**, in Gruppen von 3 Personen.

Beachten Sie, dass von Aufgaben 3.1, 3.2 und 3.3 entweder einerseits nur 3.1 oder andererseits 3.2 und 3.3 für die Abgabe auszusuchen sind. Es werden dementsprechend entweder einerseits nur 3.1 oder andererseits 3.2 und 3.3 bepunktet korrigiert. Bei Abgabe aller Aufgaben bitte kenntlich machen, ob entweder einerseits nur 3.1 oder andererseits 3.2 und 3.3 bewertet werden sollen. Erfolgt keine solche Kennzeichnung, werden nur Aufg. 3.2 und 3.3 bewertet. Nun aber genug bzgl. des einerseits und andererseits!

Aufgabe 3.1: Programmieraufgabe: Streichhölzer ziehen (Entweder hier: 4)

Vorbereitung. Laden Sie bitte das ZIP-Archiv spiele.zip herunter von unserer eCampus-Seite unter Kursunterlagen » Python und AIMA Python » AIMA-Py Spiele. Das ZIP-Archiv enthält den Ordner *spiele* mit den Skripten *matchesrules.py* und *minimax.py*.

Das Spiel beginnt mit 4 Streichhölzern. Abwechselnd mit ihrem Gegner können Sie bis zu 2 Streichhölzer wegnehmen oder den Gegner den nächsten Zug überlassen. Falls ein Spieler zwei Streichhölzer nimmt, muss danach mindestens 1 Streichholz übrig bleiben. Falls sich Spieler A entscheidet kein Streichholz zu nehmen, muss Spieler B mindestens ein Streichholz nehmen und Spieler A muss in der nächsten Runde auch mindestens ein Streichholz nehmen. Es gewinnt der Spieler, der als letztes ein einzelnes Streichholz nimmt. Der Benutzer bzw. die Benutzerin beginnt das Spiel als MIN.

Aufgabe: Implementieren Sie die Alpha-Beta-Suche und wenden Sie diese auf das Beispiel des Ziehens von Streichhölzern an. Gehen Sie dabei vom Skript minimax.py (Minimax-Suche) aus. Kopieren Sie dies in ein neues Skript alphabeta.py und modifizieren Sie die gegebene Minimax-Suche nun geeignet zur Alpha-Beta-Suche. Testen Sie das neue Skript alphabeta.py auf den folgenden Eingabefolgen von User. Folge 1: Take 2 - Take 1; Folge 2: Take 1 - Take 0 - Take 1; Folge 3: Take 1 - Take 2. Zum Vergleich können Sie die entspr. Minimax-Ergebnisse heranziehen.

Aufgabe 3.2: Minimax: Tic-Tac-Toe **(oder hier: 2)**

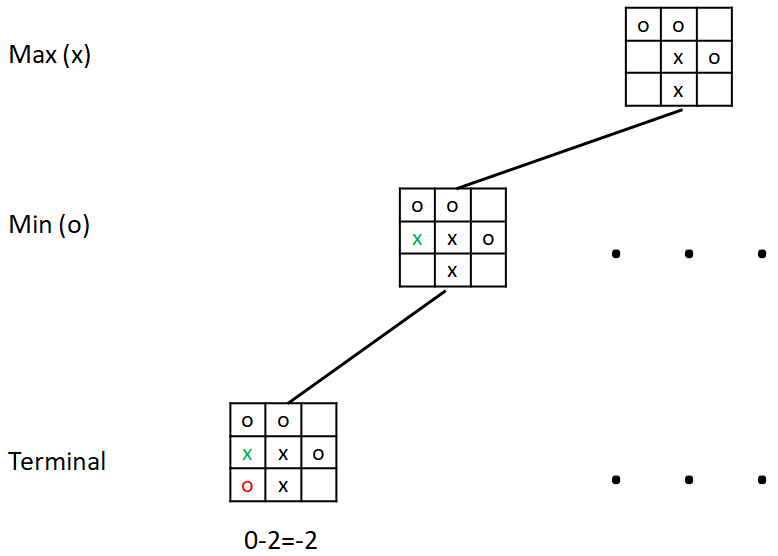
Für das Spiel Tic-Tac-Toe haben Sie ein Minimax-Programm geschrieben. Aufgrund beschränkter Ressourcen kann Ihr Minimax-Programm nur zwei Halbzüge vorausschauen. Wenn Ihr Programm also als MAX am (Halb)Zug ist, kann das Minimax-Programm nur die direkten möglichen MAX-Halbzüge sowie die direkt darauf folgenden möglichen MIN-Halbzüge berücksichten. Letztere stellen also die Blattknoten des Spielbaums dar.
Die Bewertungsfunktion von Minimax für diese Blattknoten lautet wie folgt:

- Blattknoten mit Gewinn für MAX erhalten die Bewertung 10
- Blattknoten mit Gewinn für MIN erhalten die Bewertung -10
- sonstige Blattknoten erhalten als Bewertung: Anzahl der noch offenen Gewinndreier für MAX - Anzahl der noch offenen Gewinndreier für MIN (damit sind auch finale Pattstellungen mit 0 - 0 = 0 erfasst)

a) Starten Sie mit dem Aufruf im abgebildeten Spielstand, zu dem es kam, indem MIN den ersten Halbzug hatte und nach 2,5 Halbzügen bereits 3 Felder belegt hat (mit Kreisen) und Ihr Minimax-Programm zwei Felder belegt hat (mit Kreuzen). MAX ist also dran und muss zwischen vier Feldern entscheiden.

o	o	
	x	o
	x	

Geben Sie den resultierenden Spielbaum explizit wieder: mit **allen** Nachfolgezuständen, den Bewertungen **aller** Blattknoten und **allen** hochgereichten MIN- und MAX-Werten. Die folg. Abbildung zeigt einen Pfad in diesem Suchbaum.



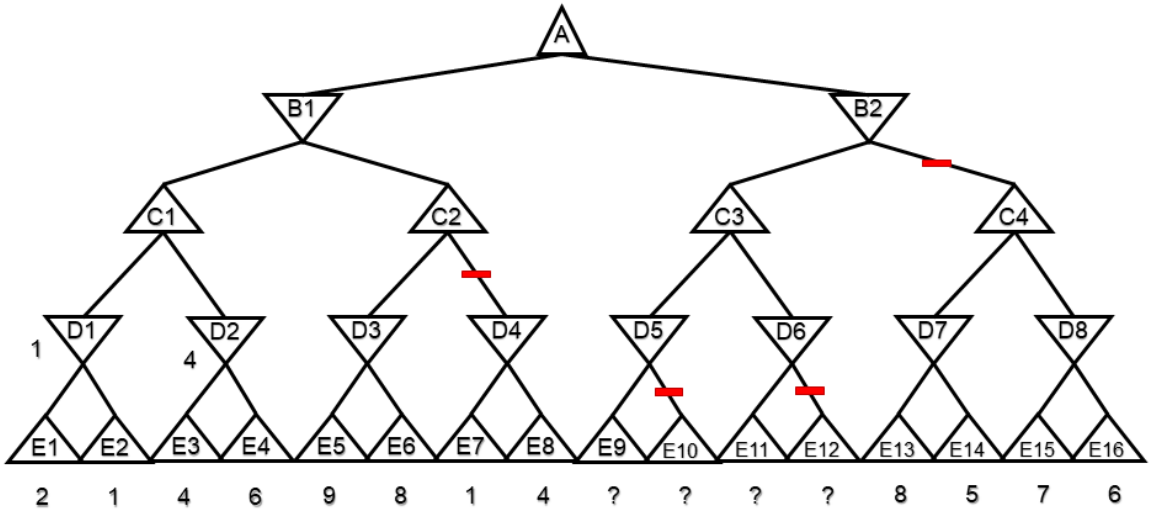
Welchen (Halb)Zug hat MAX zu wählen? Umkreisen Sie den entspr. Zustand im Spielbaum eindeutig.

b) Aus dem Baum von Aufgabenteil a) ergibt sich eindeutig der umzusetzende (Halb)Zug für MAX (auch durch anschauliche Betrachtung). Gehen Sie davon aus, dass MIN auf diesen Zug mit dem für MAX schlechtesten (Halb)Zug reagiert. Danach sind nur noch 2 Felder zu belegen und MAX hat den ersten Halb(Zug).

Starten Sie also erneut Minimax auf diesem Spielzustand. Geben Sie auch diesen resultierenden Spielbaum explizit wieder mit **allen** Nachfolgezuständen, den Bewertungen **aller** Blattknoten und **allen** hochgereichten MIN- und MAX-Werten. Welchen (Halb)Zug hat MAX zu wählen? Umkreisen Sie den entspr. Zustand im Spielbaum eindeutig.

Aufgabe 3.3: Alpha-Beta-Kürzung (und oder hier: 2)

Die Abbildung zeigt einen Vier-Schichten-Spielbaum mit dem Max-Knoten A als Wurzelknoten in der obersten Schicht und darauf folgend Min- und Maxknoten Knoten bis zu Terminalknoten in der untersten Schicht. Die roten Kanten zeigen jeweils eine Pruning-Stelle.



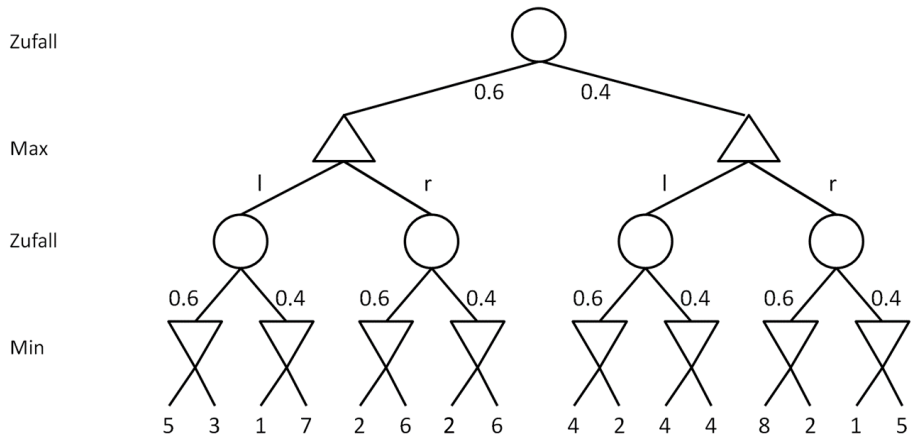
a) Tragen Sie in der folg. Tabelle die α- und β-Werte in den Zeilen 2 bis 5 ein. Begründen Sie anhand der Werte, warum es in Zeile 5 zu einem Pruning unterhalb von C2 kommt.

Zeile	Knoten	nach Besuch von Knoten	α	β
1	A	bei Start	$-\infty$	$+\infty$
2	D1	E1		
3	C1	D1		
4	C1	D2		
5	C2	D3		

b) Belegen Sie die Werte der Terminalknoten E9, E10, E11 und E12 mit positiven Ganzzahlen derart, dass die drei Prunings unterhalb der Knoten D5, D6 und B2 zustande kommen. Begründen Sie dann die Prunings an den entspr. Stellen.

Aufgabe 3.4: Expectimax **(2)**

Betrachten Sie ein Spiel, bei dem MAX eine unfaire Münze wirft und dann nach links (l) oder rechts (r) ziehen kann, danach macht MIN das Gleiche. Die Münze hat die Ergebnisse Zahl und Wappen, die mit den Wahrscheinlichkeiten P(Zahl) = 0.6 bzw. P(Wappen) = 0.4 eintreten. Für die Teilaufgaben b) und c) soll noch offen sein, wie der Münzwurf für A ausging. Daher wird der Baum hier mit dem entspr. Zufallsknoten gestartet. Die Nutzenwerte der Endzustände werden in folgendem Spielbaum dargestellt:



a) Schreiben Sie die Expectimax-Werte (vgl. Vorlesung 5, Folie 29) an die Knoten.

b) Welchen Zug sollte MAX wählen, wenn Zahl bzw. wenn Wappen geworfen wird?

c) Wie ist der erwartete Nutzen des Spiels für MAX bei offenen Münzwurfergebnis (im Wurzelknoten abzulesen)?

Aufgabe 3.5: KNF **(2)**

Wandeln Sie die aussagenlogische Formel $f = ((A \Rightarrow B) \wedge (C \Leftrightarrow (A \wedge \neg B)))$ mit Hilfe der KNF-Transformationsregeln (Vorlesung 6, Folie 28) in KNF um.

Aufgabe 3.6: AL-Resolution **$(\mathbf{0}, \mathbf{5} + \mathbf{1}, \mathbf{5} = \mathbf{2})$**

a) Geben Sie alle möglichen Resolventen an, die aus den Klauseln der in KNF gegebenen Wissensbasis

$$\Delta_1 = \{[\neg a, b, \neg c, \neg d], [\neg a, \neg b, \neg c, d]\}$$

ableitbar sind.

b) Gegeben sei folgende Wissensbasis in KNF:

$$\Delta_2 = \{[a, \neg b], [b, b, d], [b, \neg d], [\neg a, \neg b, \neg c]\}$$

Prüfen Sie mittels Resolution über einen Widerspruchsbeweis: $KB \vdash_R \ a$?

Hinweis zum Formalismus: Eckige Klammern fassen disjunktiv verknüpfte Literale als Klauseln zusammen, die wiederum konjunktiv untereinander verknüpft sind.