Klausur zur Vorlesung

Künstliche Intelligenz

Prof. J. Fürnkranz

Technische Universität Darmstadt — Wintersemester 2017/18

Termin: 20. 2. 2018

Name: **Vorname:** **Matrikelnummer:**

Fachrichtung:

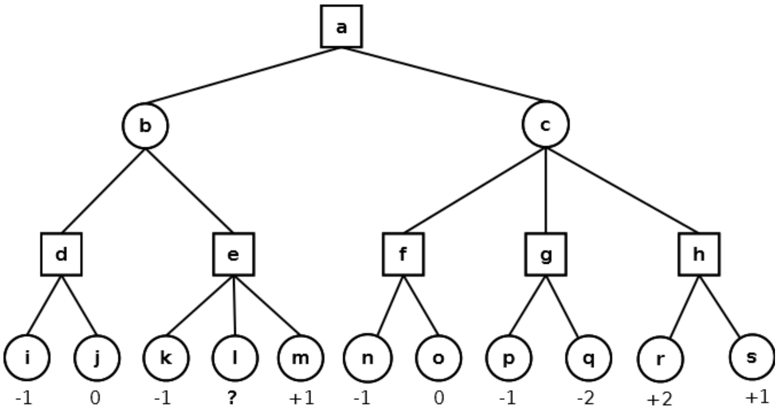
Punkte:	(1)	(2)	(3)	(4)	(5)	**Summe:**

- **Aufgaben:** Diese Klausur enthält auf den folgenden Seiten 5 Aufgaben zu insgesamt 100 Punkten. Jede Aufgabe steht auf einem eigenen Blatt. Kontrollieren Sie *sofort*, ob Sie alle sechs Blätter erhalten haben!
- **Zeiteinteilung:** Die Zeit ist knapp bemessen. Wir empfehlen Ihnen, sich zuerst einen kurzen Überblick über die Aufgabenstellungen zu verschaffen, und dann mit den Aufgaben zu beginnen, die Ihnen am besten liegen.
- **Papier:** Verwenden Sie nur Papier, das Sie von uns ausgeteilt bekommen. Bitte lösen Sie die Aufgaben auf den dafür vorgesehenen Seiten. Falls der Platz nicht ausreicht, vermerken sie dies bitte und setzen die Lösung auf der letzten Seite fort. Brauchen Sie zusätzlich Papier (auch Schmierpaper), bitte melden.
- **Fragen:** Sollten Sie Teile der Aufgabenstellung nicht verstehen, bitte fragen Sie!
- **Abschreiben:** Sollte es sich herausstellen, dass Ihre Lösung und die eines Kommilitonen über das zu erwartende Maß hinaus übereinstimmen, werden beide Arbeiten negativ beurteilt (ganz egal wer von wem in welchem Umfang abgeschrieben hat).
- **Ausweis:** Legen Sie Ihren *Studentenausweis* und *Lichtbildausweis* sichtbar auf Ihren Platz. Füllen Sie das Deckblatt sofort aus!
- **Hilfsmittel:** Zur Lösung der Aufgaben ist ein von Ihnen selbst handschriftlich beschriebenes DIN-A4-Blatt erlaubt. Gedruckte Wörterbücher sind für ausländische Studenten erlaubt, elektronische Hilfsmittel (Taschenrechner, elektronische Wörterbücher, Handy, etc.) sind verboten! Sollten Sie etwas verwenden wollen, was nicht in diese Kategorien fällt, bitte klären Sie das *bevor* Sie zu arbeiten beginnen.
- **Aufräumen:** Sonst darf außer Schreibgerät, Essbarem, von uns ausgeteiltem Papier und eventuell Wörterbüchern nichts auf Ihrem Platz liegen. Taschen bitte unter den Tisch!

Gutes Gelingen!

Aufgabe 1 α-β-Suche (2/2/8/3/3/3 = 21 Punkte)

Gegeben sei der folgende Suchbaum (im Knoten **a** ist MAX am Zug):



1–a Im Blatt-Knoten l fehlt ein Wert. Können Sie dennoch den Minimax-Wert des Baums eindeutig bestimmen? Wenn ja, was ist der MiniMax-Wert? Wenn nein, warum nicht?

1–b Können Sie eine eindeutige Hauptvariante angeben? Wenn ja, welche? Wenn nein, warum nicht?

1–c Geben Sie jeweils den Alpha- und Beta-Wert an, mit denen eine Alpha-Beta-Suche die Knoten c, e, j, l betreten würde, wenn Sie annehmen, daß der Knoten a mit $\alpha = -\infty$ und $\beta = +\infty$ betreten wird, und die Nachfolgeknoten jedes Knotens jeweils in der angegebenen Reihenfolge (von links nach rechts) durchsucht werden.

1–d Welche Werte aus dem Intervall $[-\infty, +\infty]$ können dem Blatt-Knoten l zugeordnet werden, sodaß eine Alpha-Beta-Suche den Knoten m prunen kann?

1–e In welcher Reihenfolge sollte der Knoten c seine Nachfolge-Knoten f, g und h durchsuchen, damit möglichst wenige Knoten durchsucht werden müssen und möglichst viel geprunt werden kann?

1–f In der Vorlesung haben wir auch die sogenannte NegaMax-Formulierung der Alpha-Beta Suche kennen gelernt. Erklären Sie kurz die Idee der NegaMax-Formulierung, und skizzieren Sie, was sich in der Ausgabe von c) ändern würde, wenn man diese Version der Alpha-Beta-Suche verwenden würde.

Aufgabe 2 Constraint Satisfaction Problems (5/4/5/3 = 17 Punkte)

Sie sollen ein 3×3 magisches Quadrat erstellen, d.h. ein Quadrat, in dem jedes Feld mit einer der Zahlen 1, 2, oder 3 gefüllt wird, sodaß jede Zeile die Summe 6 ergibt, und jede Spalte die Summe 6 ergibt.

1	A	B
D	2	C
E	F	3

Für drei Felder sind Werte vorgegeben, die freien Felder A, B, C, D, E, F sollen gefüllt werden.

2–a Geben Sie alle binären Constraints für die Variablen A, B, C, D, E, F an und zeichnen Sie den Constraint Graph.

2–b Zusätzlich sei noch verlangt, daß die Summe von E und B gleich t sein muß. Welche Werte aus $\{1, 2, 3, 4, 5, 6\}$ können Sie für t wählen, sodaß E und B mit dem Wertebereich $\{1, 2, 3\}$ arc-consistent bezüglich dieses Constraints sind?

2–c Sie führen auf den Constraints, die sie in a) definiert haben, Constraint Propagation mittels des Arc-Consistency Algorithmus (AC-3) aus, der mittels Forward Checking Einschränkungen von Variablen-Bereichen an die jeweiligen Nachbarn im Constraint Graph weiterpropagiert.

Die Queue soll mit dem einzigen Constraint $A + B = 5$ initialisiert werden.

Welche Wertebereiche für die Variablen sind das Ergebnis der Abarbeitung?

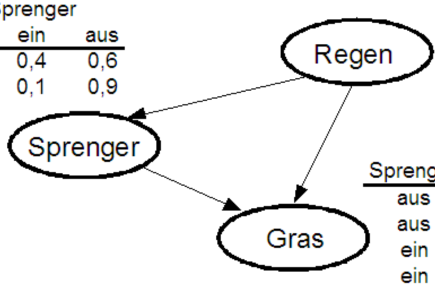
Hinweis: Das Constraint aus Aufgabe b) ist hier nicht mehr gültig.

2–d Wenn ein Problem mittels Forward-Checking und Constraint Propagation nicht direkt lösbar ist, wie können diese Verfahren erweitert werden, sodaß eine Lösung gefunden werden kann?

Aufgabe 3 Bayes'sche Netze (4/4/6/4 = 18 Punkte)

Gegeben sei folgendes Bayes'sches Netz, das die Ereignisse Regen (ja/nein), (Rasen-)Sprenger (ein/aus) und Gras (nass/trocken) und deren Zusammenhänge repräsentiert.

	Sprenger	
Regen	ein	aus
nein	0,4	0,6
ja	0,1	0,9



Regen	
ja	nein
0,2	0,8

		Gras	
Sprenger	Regen	nass	trocken
aus	nein	0,0	1,0
aus	ja	0,8	0,2
ein	nein	0,9	0,1
ein	ja	1,0	0,0

Beantworten Sie folgende Fragen, und geben Sie jeweils eine kurze Begründung bzw. eine entsprechende Berechnung des Ergebnisses an:

3–a Wie groß ist die Wahrscheinlichkeit, daß das Gras nass ist, es regnet und der Rasensprenger aus ist?

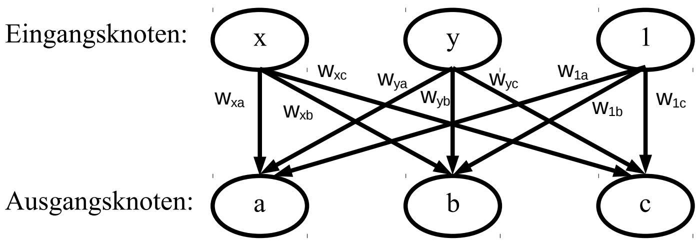
3–b Wie groß ist die Wahrscheinlichkeit, daß das Gras nass ist, gegeben daß es regnet und der Rasensprenger aus ist?

3–c Wie groß ist die Wahrscheinlichkeit, daß es regnet und der Rasensprenger aus ist, gegeben daß das Gras nass ist?

3–d Erklären Sie kurz die Grundidee von Rejection Sampling zur Bestimmung einer approximativen Wahrscheinlichkeit für Queries an ein Bayes'sches Netz.

Aufgabe 4 Neuronale Netze (7/4/3/4 = 18 Punkte)

Gegeben sei folgende Struktur eines neuronalen Netzes mit den Eingangsknoten x, y und *1*, wobei an *1* konstant 1 anliegt, und den Ausgangsknoten a, b und c.



Das Netzwerk erlaubt als Eingabe- und Ausgabesignale jeweils nur 0 und 1, wobei 0 für logisch *false* und 1 für logisch *true* steht. Die verwendete Aktivierungsfunktion ist

$$g(x) = \begin{cases} 1 & \text{falls } x > 0 \\ 0 & \text{sonst.} \end{cases}$$

4–a Legen Sie die Gewichte $w_{xa}, w_{xb}, w_{xc}, w_{ya}, w_{yb}, w_{yc}, w_{1a}, w_{1b}, w_{1c}$ des neuronalen Netzes so fest, dass der Ausgang

- out_a=1 ist, falls die Anzahl der "1" in den Eingängen x und y **kleiner oder gleich 0** ist, sonst $out_a = 0$,
- out_b=1 ist, falls die Anzahl der "1" in den Eingängen x und y **kleiner oder gleich 1** ist, sonst $out_b = 0$,
- out_c=1 ist, falls die Anzahl der "1" in den Eingängen x und y **kleiner oder gleich 2** ist, sonst $out_c = 0$.

Wählen Sie **ganzzahlige** Gewichte und tragen Sie Ihre Werte in folgende Tabelle ein:

w von \ nach	a	b	c
x			
y			
1			

Vervollständigen Sie außerdem folgende Tabelle um die Eingangssignale in den Knoten a, b, c. Benutzen Sie hierbei für die Berechnung die oben eingetragenen Kantengewichte.

		Eingangssignale			Ausgangssignale		
x	y	in_a	in_b	in_c	out_a	out_b	out_c
0	0				1	1	1
0	1				0	1	1
1	0				0	1	1
1	1				0	0	1

4–b Sie möchten nun an einen der Ausgangsknoten die Funktion "*genau ein signal ist 1*" berechnen lassen. Lassen sich für die obige Netzwerkstruktur Gewichte wählen, sodass eine Umsetzung der verlangten Funktion möglich ist? Begründen Sie Ihre Antwort bzw. zeigen Sie ggf. eine mögliche Lösung auf.

4–c Nehmen Sie für das ursprüngliche Netzwerk die Aktivierungsfunktion

$$g(x) = 2x$$

und folgende Gewichte an:

w von \ nach	a	b	c
x	-1	1	2
y	-2	1	0
1	0	0	0

Berechnen Sie mittels Forward-Propagation das Ausgangssignal out_a für das Eingangssignal $(x, y) = (1, 2)$.

4–d Nehmen Sie nun ferner an, dass das Netzwerk aus Aufgabe c) für das Eingangssignal $(x, y) = (1, 2)$ die Ausgabe $(a', b', c') = (5, -2, -2)$ liefern soll.

Bestimmen Sie den Fehlerterm Δ_a und berechnen Sie die Gewichtsänderung für das Gewicht w_{xa} für eine Lernrate $\alpha = \frac{1}{2}$.

Aufgabe 5 A^*, Planning, Reinforcement Learning (4/4/4/4/5/5 = 26 Punkte)

5–a Angenommen, Sie haben mehrere unterschiedliche zulässige Heuristiken h_1, h_2, $\ldots h_N$ für ein Problem. Wie können Sie das beste Verhalten erzielen? (eine richtige Antwort)

☐ Auswahl der Heuristik, die die tatsächlichen Kosten am wenigsten unterschätzt

☐ Auswahl der Heuristik, die am nähesten an den tatsächlichen Kosten liegt

☐ Kombination aller Heuristiken, indem in jedem Knoten das Maximum über alle Heuristiken gebildet wird.

Begründen Sie Ihre Antwort.

5–b Sie haben zwei Heuristiken für ein Problem: h_0 schätzt die verbleibenden Kosten in jedem Zustand immer mit 0 ab, h_1 errät die verbleibenden Kosten immer genau.

Findet A^* mit beiden Heuristiken immer dieselbe Lösung? Was ist der Vorteil von h_1 über h_0?

5–c Erklären Sie den Unterschied zwischen Progression Planning und Regression Planning in STRIPS.

5–d Erklären Sie genau, wie ein STRIPS-Operator bei Regression Planning eingesetzt werden kann. Wann ist ein Operator anwendbar? Wie ändern sich die Zustandsbeschreibung und die Liste der Teilziele?

5–e In der Vorlesung haben wir das System MENACE kennengelernt, dass mit Hilfe von verschiedenfarbigen Perlen in Zündholzschachteln lernt, Tic-Tac-Toe zu spielen. Wie wird in diesem Algorithmus der Exploration/Exploitation Trade-off realisiert? Wie wird hier die Funktion $Q(s, a)$ repräsentiert?

5–f Einige Reinforcement Learning Verfahren setzen voraus, daß man die Zustandsübergangsfunktion δ kennt. Wie nennt man diese Eigenschaft und mit welchen Verfahren kann man auch lernen, ohne diese Funktion zu kennen? Was ist die Grundidee dieser Verfahren?

Lösung 1

1–a 2 *Punkte* Ja. MiniMax = 0.

1–b 2 *Punkte* Ja. $a \rightarrow b \rightarrow d \rightarrow j$.

1–c 8 *Punkte* (je Wert 1 Punkte)

$$\alpha(c) = 0, \beta(c) = +\infty$$
$$\alpha(e) = -\infty, \beta(e) = 0$$
$$\alpha(j) = -1, \beta(j) = +\infty$$
$$\alpha(l) = -1, \beta(l) = 0$$

1–d 3 *Punkte* Alle Werte aus dem Intervall $[0, \infty]$ führen zu einem Cutoff, da sie größer als der β-Wert sind, und somit in Teilbaum d eine für den Min-Spieler günstigere Variante zu finden ist.

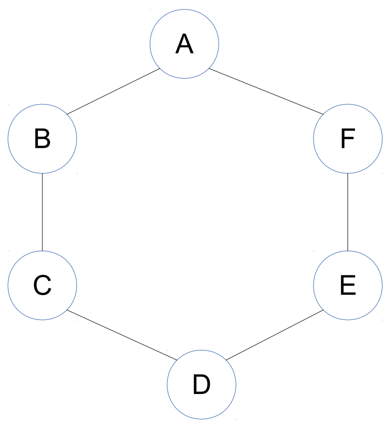
1–e 3 *Punkte* Knoten g oder f muß zuerst durchsucht werden, die anderen beiden Knoten können dann geprunt werden.

1–f 3 *Punkte* NegaMax ersetzt die Minimierung an den Min-Knoten durch eine Maximierung der negierten Werte. Dadurch müssen auch die α und β-Werte angepaßt werden ($\alpha \leftarrow -\beta$ und $\beta \leftarrow -\alpha$).
D.h., daß bei allen Min-Knoten in c) entsprechend geänderte Werte ausgegeben werden (z.B. $(l, 0, +1)$ statt $(l, -1, 0)$).

Lösung 2

2–a 5 *Punkte*

$$A + B = 5$$
$$D + C = 4$$
$$E + F = 3$$
$$D + E = 5$$
$$A + F = 4$$
$$B + C = 3$$



2–b 4 *Punkte*

Nur 4 kann gewählt werden, da es z.B. bei $t = 5$ keinen gültigen Wert von B für den Wert $E = 1$ gibt, oder bei $t = 3$ keinen gültigen Wert für $E = 3$.

2–c 5 *Punkte*

Alle Domänen werden auf einen Wert reduziert, d.h. das Problem wird direkt gelöst.

$$\begin{aligned}
dom(A) &= \{3\} \\
dom(B) &= \{2\} \\
dom(C) &= \{1\} \\
dom(D) &= \{3\} \\
dom(E) &= \{2\} \\
dom(F) &= \{1\}
\end{aligned}$$

2–d 3 *Punkte* Mit Backtracking-Suche, wie in den Übungen.

Lösung 3

3–a 4 *Punkte*

$$\begin{aligned}
P(g_{nass}, r_{ja}, s_{aus}) &= P(g_{nass}|s_{aus}, r_{ja}) \times P(s_{aus}|r_{ja}) \times P(r_{ja}) = \\
&= 0.8 \times 0.9 \times 0.2 = 0.144
\end{aligned}$$

3–b 2 *Punkte*

$$P(g_{nass}|r_{ja}, s_{aus}) = 0.8$$

Anmerkung: Diesen Wert kann man direkt aus der Tabelle ablesen.

3–c 6 *Punkte*

$$P(r_{ja}, s_{aus}|g_{nass}) = \frac{P(g_{nass}, r_{ja}, s_{aus})}{P(g_{nass})} = \frac{0.144}{0.452} = \frac{36}{113} \approx 0.32$$

da

$$\begin{aligned}
P(g_{nass}) &= \sum_{r \in \{ja,nein\}} \sum_{s \in \{aus,ein\}} P(g_{nass}|r, s) \times P(s|r) \times P(r) \\
&= P(g_{nass}|r_{ja}, s_{aus}) \times P(s_{aus}|r_{ja}) \times P(r_{ja}) + \\
&\quad P(g_{nass}|r_{ja}, s_{ein}) \times P(s_{ein}|r_{ja}) \times P(r_{ja}) + \\
&\quad P(g_{nass}|r_{nein}, s_{aus}) \times P(s_{aus}|r_{nein}) \times P(r_{nein}) + \\
&\quad P(g_{nass}|r_{nein}, s_{ein}) \times P(s_{ein}|r_{nein}) \times P(r_{nein}) \\
&= 0.8 \times 0.9 \times 0.2 + \\
&\quad 1.0 \times 0.1 \times 0.2 + \\
&\quad 0.0 \times 0.6 \times 0.8 + \\
&\quad 0.9 \times 0.4 \times 0.8 \\
&= 0.144 + 0.020 + 0 + 0.288 = 0.452
\end{aligned}$$

3–d 4 *Punkte* Beim Rejection Sampling werden zufällig Instanzen für eine Query $P(X|e)$ generiert. Wenn eine Instanz nicht mit den Evidenzwerten e übereinstimmt, wird sie verworfen, ansonsten wird der Zähler für die beobachtete Ausprägung der X-Variablen hochgezählt. Nach vielen Iterationen konvergieren die aus der Normalisierung dieser Zähler geschätzten Wahrscheinlichkeiten. Das Verfahren ist aber in der Praxis uninteressant, da zu viele Instanzen verworfen werden müssen.

Lösung 4

4–a

w von \ nach	a	b	c
x	-1	-1	-1
y	-1	-1	-1
1	1	2	3 oder > 3

$\epsilon = 1$

		Eingangssignale			Ausgangssignale		
x	y	a	b	c	a	b	c
0	0	$0+\epsilon$	$-1+\epsilon$	$0+\epsilon$	1	1	1
0	1	$-1+\epsilon$	$0+\epsilon$	$1+\epsilon$	0	1	1
1	0	$-1+\epsilon$	$0+\epsilon$	$1+\epsilon$	0	1	1
1	1	$-2+\epsilon$	$1+\epsilon$	$2+\epsilon$	0	0	1

4–b Nein, das Netzwerk kann man in dieser Form nicht verwenden werden. Es ist nicht mg̈lich, diese Funktionen mit diesem einfachen Netzwerk zu lernen, da es keine linearen Funktionen sind. Es wird ein weiterer Layer gebraucht. Der kann z.B. so aussehen, dass er direkt an den Ausgängen des Netzwerkes aus A1 anknüpft. $= 1$ würde dann z.B. als ≤ 1 AND NOT ≤ 0 umgesetzt.

4–c $in_a = 1 \cdot -1 + 2 \cdot (-2) = -5 \;\; out_a = -10$

$g'(x) = 2$

$\Delta_a = Err_a \cdot g'(in_a) = (5 - (-10)) \cdot 2 = 30$

4–d $w_{xa} \leftarrow w_{xa} + \alpha \cdot \Delta_a \cdot out_x = -1 + \frac{1}{2} \cdot 30 \cdot 1 = 14$