Klausur zur Vorlesung

Künstliche Intelligenz

Prof. J. Fürnkranz

Technische Universität Darmstadt — Sommersemester 2007 (Klausur WS07/08)

Termin: 28. 1. 2008

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

Fachrichtung: **Wiederholer:** ☐ ja ☐ nein

☐ Diplom ☐ Master ☐ Bachelor

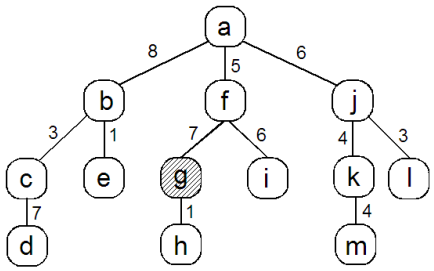
Punkte: (1) .... (2) .... (3) .... (4) .... (5) .... **Summe:**

- **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 sieben 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 Suche (22 Punkte)

Betrachten Sie folgenden Suchbaum für eine beim Knoten a beginnenden Suche



Die Zahlen neben den Kanten geben die Kosten an, die benötigt werden, um über diese Kante zu wandern.

1–a Geben Sie für die folgenden Suchverfahren jeweils den Suchrand (die *Fringe* bzw. *Open*-List) zu dem Zeitpunkt an, als g zur Expansion ausgewählt wird. Beachten Sie auch die Reihenfolge der Knoten in der Fringe!

1. Tiefensuche (ohne Berücksichtigung der Kosten)
   (2 Punkte) g, i, j
2. Breitensuche (ohne Berücksichtigung der Kosten)
   (2 Punkte) g, i, k, l, d
3. Uniform Cost Search. Geben Sie hier auch die Kosten der Knoten in der Fringe an.
   (4 Punkte) $g : 12, m : 14, d : 18$

1–b Führen Sie, beginnend beim Knoten a, eine A*-Suche aus bis der Knoten g zur Expansion selektiert wird, wobei folgende Heuristik h verwendet wird:

	a	b	c	d	e	f	g	h	i	j	k	l	m
$h(\cdot) =$	8	1	3	7	4	5	0	3	6	2	1	5	4

Geben Sie alle Zwischenschritte an, d.h. welche Knoten jeweils in der Fringe-List sind (d.h. offen sind), deren Werte für $g(n)$, $h(n)$ und $f(n)$ sowie, welcher Knoten jeweils ausgewählt wird bzw. warum die Suche abgebrochen wird.

(8 Punkte)

1.

Node	f	g	h	
a	***8***	0	8	←
b	9	8	1	
f	10	5	5	
j	***8***	6	2	←

2.

Node	f	g	h	
b	***9***	8	1	←
f	10	5	5	
k	11	10	1	
l	14	9	5	

3.

Node	f	g	h	
f	***10***	5	5	←
k	11	10	1	
l	14	9	5	
c	14	11	3	
e	13	9	4	

4.

Node	f	g	h	
k	***11***	10	1	←
l	14	9	5	
c	14	11	3	
e	13	9	4	
g	12	12	0	
i	17	11	6	

Anmerkung: *g wird hier generiert, aber noch nicht zur Expansion ausgewählt, da k eventuell noch eine billigere Lösung sein könnte!*

5.

Node	f	g	h	
l	14	9	5	
c	14	11	3	
e	13	9	4	
g	***12***	12	0	←
i	17	11	6	
m	18	14	4	

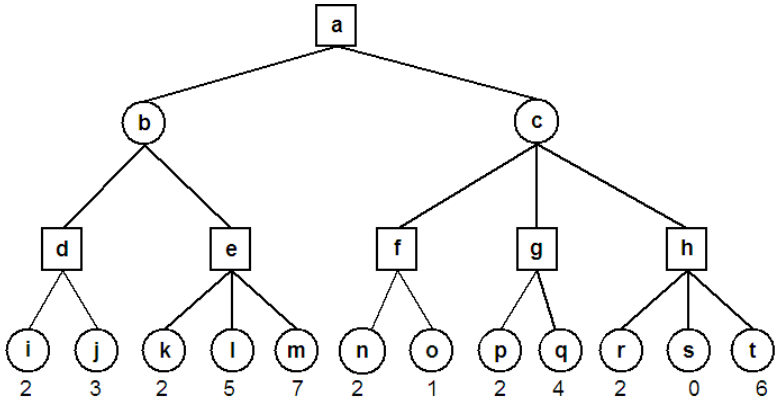
Anmerkung: *g wird hier zur Expansion ausgewählt. Da es keine günstigere Lösung geben kann, wird hier gestoppt.*

1–c Nehmen Sie eine Heuristik h_0 an, die alle Kosten mit 0 schätzt, d.h. $h_0(.) = 0$ für alle Knoten. Beantworten Sie folgende Fragen und begründen Sie jeweils Ihre Antwort.

1. Ist die Heuristik h_0 zulässig?
   (1 Punkte) *Ja, da sie alle Kosten unterschätzt.*
2. Ist die Heuristik h_0 konsistent?
   (1,5 Punkte) *Ja, da jeder Kantenübergang Kosten ≥ 0 hat, und diese Kosten mit 0 geschätzt werden.*
3. Wird A* mit h_0 im allgemeinen eine optimale Lösung finden?
   (1,5 Punkte) *Ja, da h_0 zulässig ist.*
4. A* mit h_0 ist äquivalent zu einem anderen, in der Vorlesung kennengelernten Algorithmus. Zu welchem?
   (2 Punkte) *Zur Uniform Cost Search (siehe 1-b-3.)*

Aufgabe 2 α-β-Suche (22 Punkte)

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



2–a Geben Sie den MiniMax-Wert des Wurzelknotens an

(2 Punkte) *MiniMax* $= 3$.

2–b Führen Sie eine Alpha-Beta Suche durch (die Nachfolger eines Knotens werden, wie in der Vorlesung, von links nach rechts durchsucht). Nehmen Sie an, daß der Alpha-Beta-Algorithmus wie in der Vorlesung durch folgende beiden Funktionen implementiert ist:

- MIN-VALUE($State, \alpha, \beta$): berechnet und retourniert den Wert des Min-Knotens *State* mit gegebenem α und β.
- MAX-VALUE($State, \alpha, \beta$): berechnet und retourniert den Wert des Max-Knotens *State* mit gegebenem α und β.

In jeder dieser beiden Funktionen werden nun unmittelbar nach dem Funktionsaufruf die übergebenen Parameter auf dem Bildschirm ausgegeben (also z.B. durch `println`($State, \alpha, \beta$)).

Geben Sie die Ausgaben beim Durchsuchen obigen Baumes an (Aufruf: MAX-VALUE($a, -\infty, +\infty$)).

(9 Punkte)

$a, \quad -\infty, \quad +\infty$
$b, \quad -\infty, \quad +\infty$
$d, \quad -\infty, \quad +\infty$
$i, \quad -\infty, \quad +\infty$
$j, \quad 2, \quad +\infty$
$e, \quad -\infty, \quad 3$
$k, \quad -\infty, \quad 3$
$l, \quad 2, \quad 3$
$c, \quad 3, \quad \infty$
$f, \quad 3, \quad \infty$
$n, \quad 3, \quad \infty$
$o, \quad 3, \quad \infty$

Anmerkung: *Bei den Knoten m, g, und h wird geprunt.*

2–c In der Vorlesung haben wir auch die sogenannte NEGAMAX-Formulierung der Alpha-Beta Suche kennen gelernt, die die beiden Funktionen MAX-VALUE und MIN-VALUE durch eine einzige Funktion ersetzt. Erklären Sie kurz die Idee der NEGAMAX-Formulierung, und skizzieren Sie, was sich in der Ausgabe von b) ändern würde, wenn man diese Version der Alpha-Beta-Suche verwenden würde.

(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 2-b entsprechend geänderte Werte ausgegeben werden (z.B. $(c, -\infty, -3)$ oder $(l, -3, -2)$).*

2–d Bei praktischen Anwendungen der Minimax- oder AlphaBeta-Suche in vielen Karten- und Brettspielen ist es nicht möglich, den kompletten Spielbaum zu durchsuchen, da dieser zu groß ist. Nennen Sie zwei Ansätze, wie praktische Spielprogramme dieses Problem in den Griff bekommen.

(4 Punkte)

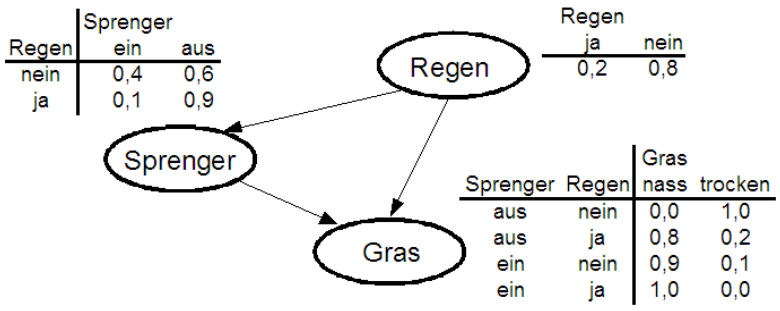
1. *Durch Begrenzung der Tiefe der Suche und Verwendung einer Evaluierungsfunktion an den Blättern.*
2. *Durch Begrenzung der Breite der Suche, indem nur manche der möglichen Pfade bis ans Ende versucht werden, und der Zug ausgewählt, der statistisch am besten abgeschnitten hat (Simulation Search).*

2–e Iterative Deepening durchsucht viele Knoten mehrfach. Dennoch wird es häufig in Spielprogrammen in Kombination mit Alpha-Beta-Suche eingesetzt, und ist dort oft sogar schneller als eine einfache Alpha-Beta-Suche. Wie kann das passieren?

(4 Punkte) Erstens werden die Kosten von Iterative Deepening von der letzten Iteration dominiert. Zweitens lassen sich gerade bei der Alpha-Beta-Suche die Informationen aus den vohergehenden Iterationen verwenden, um die Zugreihenfolge in der letzten Iteration zu optimieren, und dadurch eine insgesamt schnellere Laufzeit zu erzielen.

Aufgabe 3 Bayes'sche Netze (16 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.



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

Regen ja	Regen nein
0,2	0,8

Sprenger	Regen	Gras nass	Gras 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?

(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 Wie groß ist die Wahrscheinlichkeit, daß das Gras nass ist, wenn es regnet und der Rasensprenger aus ist?

(2 Punkte)

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

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

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

(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

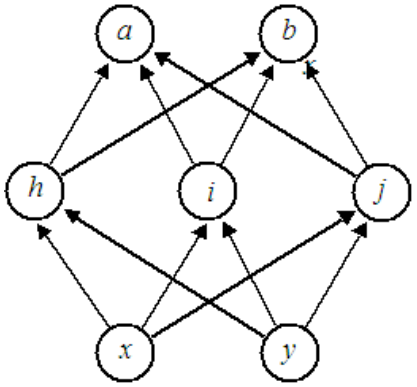
$$\begin{aligned} P(g_{nass}) &= \sum_{r\in\{ja,nein\}} \sum_{s\in\{aus,ein\}} P(g_{nass}|r, s) \times P(s|r) \times P(r) \\ &= P(g_{nass}|r_{ja}, s_{aus}) \times P(s_{aus}|r_{ja}) \times P(r_{ja}) + \\ &\quad P(g_{nass}|r_{ja}, s_{ein}) \times P(s_{ein}|r_{ja}) \times P(r_{ja}) + \\ &\quad P(g_{nass}|r_{nein},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

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

(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.

Aufgabe 4 Neuronale Netze (23 Punkte)

Gegeben sei folgendes Neuronales Netz mit einer einfachen Threshold-Aktivierungsfunktion (Output -1 für eine Eingangsaktivierung ≤ 0 und $+1$ sonst).



$W_{x,h} = 0.5$ $W_{h,a} = -0.4$
$W_{x,i} = 0.3$ $W_{i,a} = \ldots$
$W_{x,j} = -0.4$ $W_{j,a} = -0.3$
$W_{y,h} = 0.8$ $W_{h,b} = -0.5$
$W_{y,i} = 0.2$ $W_{i,b} = \ldots$
$W_{y,j} = 0.5$ $W_{j,b} = 0.3$

4–a Berechnen Sie die Outputs für alle Knoten im Hidden Layer für die Eingabe $x = 1$ und $y = -1$.

(3 Punkte)

$$\begin{aligned} in_h &= W_{x,h} \cdot x + W_{y,h} \cdot y = 0.5 - 0.8 = -0.3 \\ in_i &= 0.3 - 0.2 = 0.1 \\ in_j &= -0.4 - 0.5 = -0.9 \end{aligned}$$

Die Threshold-Funktion rundet das auf die Outputs $h = -1$, $i = 1$ und $j = -1$.

4–b Geben Sie grösstmögliche Wertebereiche für die fehlenden Gewichtswerte $W_{i,a}$ und $W_{i,b}$ an, sodaß der Input $(x, y) = (1, -1)$ auf den Output $(a, b) = (-1, 1)$ abgebildet wird.

(4 Punkte)

$a = -1$, d.h. $in_a \leq 0$.

$$\begin{aligned} 0 &\geq W_{h,a} \cdot h + W_{i,a} \cdot i + W_{j,a} \cdot j \\ 0 &\geq 0.4 + W_{i,a} + 0.3 \\ Wi,a &\leq -0.7 \end{aligned}$$

$b = 1$, d.h. $in_b > 0$.

$$\begin{aligned} 0 &< W_{h,b} \cdot h + W_{i,b} \cdot i + W_{j,b} \cdot j \\ 0 &< 0.5 + W_{i,b} - 0.3 \\ Wi,b &> -0.2 \end{aligned}$$

4–c Angenommen, dass das Netz den Input $(x, y) = (1, -1)$ auf den Output $(a, b) = (-1, 1)$ abbildet. Sie erhalten ein Trainingsbeispiel $(x, y, a, b) = (1, -1, -1, 1)$. Wie sehr werden sich die Gewichte $W_{h,a}$ und $W_{j,b}$ ändern?

(4 Punkte) Gar nicht, da der Zielwert mit dem vorhergesagten Wert übereinstimmt, und dadurch keine Anpassung notwendig ist.

4–d Warum wird die einfache Threshold-Funktion in praktischen Anwendungen oft durch eine sigmoide Funktion ersetzt?

(4 Punkte) Zur Bestimmung der Größ e der Fehlerkorrektur wird die Ableitung der Aktivierungsfunktion benötigt. Die Threshold-Funktion ist nicht differenzierbar (nicht einmal stetig), die Approximation durch die sigmoide Funktion dagegen schon.

4–e Warum kann ein einfaches Perceptron keine XOR-Funktion lernen?

(3 Punkte) Ein Perceptron kodiert eine Linearkombination seiner Inputs, kann also nur eine lineare Trennebene durch den Beispielraum legen. Um die positiven und negativen Beispiele der XOR-Funktion zu trennen benötigt man jedoch eine nichtlineare Funktion.

4–f Geben Sie ein Multilayer Perceptron mit zwei Knoten in einem Hidden Layer an, das eine XOR-Funktion kodiert.

(5 Punkte)

Aufgabe 5 Verschiedenes (17 Punkte)

Beantworten Sie folgende Fragen:

5–a Was ist das sogenannte Frame-Problem, und wie wird es mit dem in der Vorlesung kennen gelernten Repräsentationsformalismus aus dem STRIPS-Planner gelöst?

(4 Punkte) Das Frame-Problem ist die Tatsache, daß man in einer logischen Repräsentation der Welt im Situationskalkül auch Regeln dafür definieren muß, daß die meisten Aktionen an den meisten Zuständen nichts ändern. Bei STRIPS ist das nicht mehr notwendig, da immer nur ein konkreter Weltzustand betrachtet wird, und nur mehr die durch Aktionen hervorgerufenen Änderungen (durch die Add- und Delete-Listen) repräsentiert werden müssen.

5–b Was ist der Unterschied zwischen *State-Space Planning* und *Plan-Space Planning*?

(3 Punkte)

State-Space Planning *sucht im Raum der Weltzustände nach einem Plan, der vom Startzustand in den Endzustand führt.*

Plan-Space Planning *sucht im Raum aller möglichen Pläne, d.h., ein Zustandsübergang führt nicht von einem Weltzustand zum nächsten (wie beim SSP), sondern von einem (unfertigen) Plan zum nächsten. Der Zielzustand ist ein fertiger Plan.*

5–c Erklären Sie kurz die Idee des Temporal-Difference Learnings ohne dabei Formeln zu verwenden.

(4 Punkte) Die Grundidee des Temporal Difference Learning ist, daß man in jedem Lernschritt versucht, die Bewertung der momentanen Situation an die Bewertung der Situation einen (oder mehrere) Schritte später anzugleichen.

5–d Erklären Sie den Zusammenhang zwischen dem Chinese Room Argument und dem Turing Test. Wie, glauben Sie, beurteilt Searle den Turing Test?

(6 Punkte) Das Szenario des Chinesischen Raumes ist dem Turing-Test nachempfunden. Searle möchte damit zeigen, daß, selbst wenn aus dem Raum perfekte Antworten kommen, d.h. das System den Turing-Test bestehen würe, es klar ist, daß nichts in dem Raum chinesisch versteht, d.h. daß hier keine "Intentionalität" vorhanden sein kann, die seiner Ansicht nach Intelligenz bedingt. Der Turing-Test ist daher, laut Searle, nicht zuläßig.