Klausur zur Vorlesung

Künstliche Intelligenz

Prof. J. Fürnkranz

Technische Universität Darmstadt — Sommersemester 2009

Termin: 20. 7. 2009

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

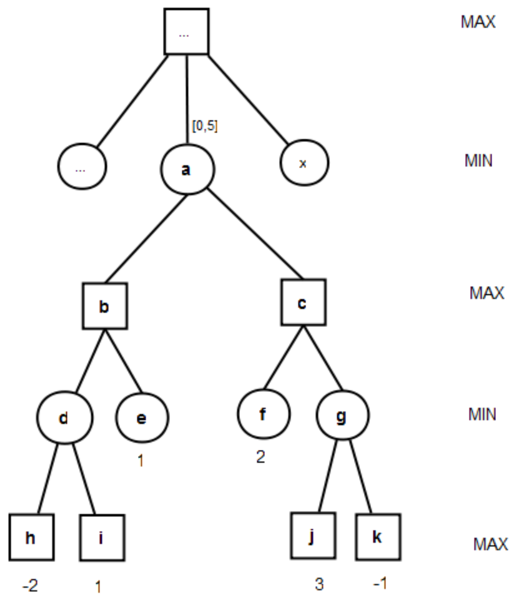
Fachrichtung:

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 Spielbaum-Suche (20 Punkte)



In nebenstehender Grafik sehen Sie einen Ausschnitt eines Spielbaums. Die Evaluierungen an den Blättern stellen die spieltheoretischen Werte aus der Sicht des MAX-Spielers dar (z.B. verliert der MAX-Spieler im Knoten h 2 Einheiten, während er im Knoten f 2 Einheiten gewinnt).

Die folgenden Aufgaben konzentrieren sich auf den mit dem Knoten a (einem MIN-Knoten) beginnenden Unterbaum.

1–a Geben Sie den MiniMax-Wert des Knotens a an.

2 Punkte
MiniMax(a) = 1

1–b Führen Sie eine Alpha-Beta Suche durch, wobei der Knoten a mit einem alpha-beta-Fenster von $[0, 5]$ aufgerufen wird. Die Nachfolger eines Knotens werden, wie in der Vorlesung, von links nach rechts durchsucht werden.

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`($State, \alpha, \beta$)).

Der Unterbaum wird also durch den Aufruf Min-Value($a, 0, 5$) betreten.

Geben Sie die Ausgaben beim Durchsuchen des Unterbaums von a an.

8 Punkte
$a, 0, 5$
$b, 0, 5$
$d, 0, 5$
$h, 0, 5$
$e, 0, 5$
$c, 0, 1$
$f, 0, 1$
Die Knoten i und g werden gepruned (j und k somit natürlich auch).

1–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*

Die Grundidee der NegaMax*-Formulierung ist, daß im Falle von Bewertungsfunktionen, die um den 0-Punkt symmetrisch sind, es gilt, daß wenn* Min *den Gewinn von* Max *minimiert, es äquivalent dazu ist, den Gewinn von* Min *zu maximieren. Die Werte müßen dabei an allen* Min*-Ebenen gespiegelt werden.*

Bei der Adaption für den $\alpha -$ beta-Algorithmus müssen wird bei einem Ebenen-Wechsel aus $[\alpha, \beta] \rightarrow [-\beta, -\alpha]$.

Konkret heißt das, daß bei den Werten von a, d, e, und f obige Transformation stattfindet.

1–d Was ist ein *Fail-High* und wie würde eine Minimal-Window-Suche (Nega-Scout) in so einem Fall verfahren?

3 *Punkte*

Ein Fail-High *zeigt an, daß die Suche ein besseres Ergebnis gebracht hat, als erwartet, das heißt die angenommene obere Schranke war vermutlich zu niedrig angesetzt. Minimal Window Search durchsucht den Baum dann nochmals mit dem Fenster*

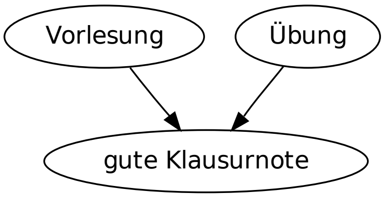
1–e Wie wird bei der UCT-Suche verhindert, daß bei der deterministischen Zugauswahl in den obersten Ebenen immer dieselbe Zugfolge ausgewählt wird?

4 *Punkte*

UCT wählt Knoten nach einer Formel aus, die zu dem momentan geschätzten Wert des Knotens noch eine Schranke zählt, die mit der Anzahl der Besuche in Nachbarknoten wächst. Das heißt, je seltener ein Knoten besucht wird, desto größer wird dieser Wert, und irgendwann kommt der Knoten dann dran.

Aufgabe 2 Bayes'sche Netze (20 Punkte)

Alice glaubt, dass für das Erlangen einer *guten Klausurnote* (K) zwei Faktoren wichtig sind. Zum einen soll der regelmäßige Besuch der *Vorlesung* (V) hilfreich sein und zum anderen ist sie davon überzeugt, dass die Bearbeitung der *Übungen* (U) eine gute Vorbereitung ist.



Die zugehörigen Wahrscheinlichkeitswerte sind :

$$
\begin{aligned}
P(V) &= 0.8\\
P(U) &= 0.5\\
P(K \mid V, U) &= 0.9\\
P(K \mid V, \neg U) &= 0.6\\
P(K \mid \neg V, U) &= 0.7\\
P(K \mid \neg V, \neg U) &= 0.3
\end{aligned}
$$

2–a Listen Sie alle möglichen Ausprägungen der Joint Probability Distribution, die durch dieses Netzwerk definiert wird.

Hinweis: Sie müssen die Wahrscheinlichkeiten nur aufzählen, nicht berechnen.

2 *Punkte*

$P(K, V, U)$
$P(K, V, \neg U)$
$P(K, \neg V, U)$
$P(K, \neg V, \neg U)$
$P(\neg K, V, U)$
$P(\neg K, V, \neg U)$
$P(\neg K, \neg V, U)$
$P(\neg K, \neg V, \neg U)$

2–b Wie wahrscheinlich ist es, dass Alice eine gute Klausurnote erhält, wenn über ihre Vorlesungs- und Übungsteilnahme nichts weiter bekannt ist?

4 *Punkte*

$$
\begin{aligned}
P(K) &= P(K \mid V, U) \cdot P(V) \cdot P(U) + P(K \mid \neg V, U) \cdot P(\neg V) \cdot P(U)\\
&+ P(K \mid V, \neg U) \cdot P(V) \cdot P(\neg U) + P(K \mid \neg V, \neg U) \cdot P(\neg V) \cdot P(\neg U)\\
&= 0.9 \cdot 0.8 \cdot 0.5 + 0.7 \cdot 0.2 \cdot 0.5 + 0.6 \cdot 0.8 \cdot 0.5 + 0.3 \cdot 0.2 \cdot 0.5 =\\
&= 0.5 \cdot (0.72 + 0.14 + 0.48 + 0.06)\\
&= 0.36 + 0.07 + 0.24 + 0.03 = \mathbf{0.7}
\end{aligned}
$$

2–c Wie wahrscheinlich ist es, dass Alice eine gute Klausurnote erhält, wenn bekannt ist, dass sie die Vorlesung regelmäßig besucht hat?

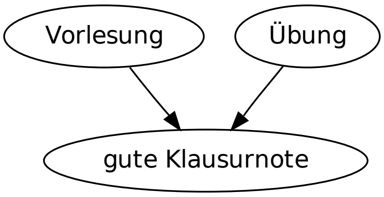
5 *Punkte*

$$\begin{aligned} P(K \mid V) &= \frac{P(K,V)}{P(V)} = \frac{1}{P(V)} \cdot (P(K \mid V,U) \cdot P(V) \cdot P(U) + P(K \mid V,\neg U) \cdot P(V) \cdot P(\neg U)) \\ &= \frac{1}{0.8}(0.9 \cdot 0.8 \cdot 0.5 + 0.6 \cdot 0.8 \cdot 0.5) \\ &= 0.9 \cdot 0.5 + 0.6 \cdot 0.5 = 0.75 \end{aligned}$$

Alternative, einfachere Berechnung:

$$\begin{aligned} P(K \mid V) &= P(K \mid V,U) \cdot P(U) + P(K \mid V,\neg U) \cdot P(\neg U) \\ &= 0.9 \cdot 0.5 + 0.6 \cdot 0.5 = 0.45 + 0.3 = \mathbf{0.75} \end{aligned}$$

Hier nochmals das Netz und die Wahrscheinlichkeitswerte:



$$
\begin{aligned}
P(V) &= 0.8 \\
P(U) &= 0.5 \\
P(K \mid V, U) &= 0.9 \\
P(K \mid V, \neg U) &= 0.6 \\
P(K \mid \neg V, U) &= 0.7 \\
P(K \mid \neg V, \neg U) &= 0.3
\end{aligned}
$$

2–d Wie wahrscheinlich ist es, dass Alice die Vorlesung regelmäßig besucht hatte, wenn bekannt ist, dass ihr Klausurergebnis schlecht ist.

Hinweis: Durch geschickte Verwendung der obigen Lösungen können Sie hier die Berechnung signifikant vereinfachen.

5 *Punkte*

Einfache Berechnung durch den Bayes'schen Satz:

$$
\begin{aligned}
P(V \mid \neg K) &= \frac{P(\neg K \mid V) \cdot P(V)}{P(\neg K)} = \frac{(1 - P(K \mid V)) \cdot P(V)}{1 - P(K)} \\
&= \frac{0.25 \cdot 0.8}{0.3} = \frac{0.2}{0.3} = \frac{\mathbf{2}}{\mathbf{3}}
\end{aligned}
$$

Direkte Berechnung:

$$
\begin{aligned}
P(V \mid \neg K) &= \frac{P(\neg K, V)}{P(\neg K)} = \frac{1}{1 - P(K)} \cdot (P(\neg K \mid V, U) \cdot P(V) \cdot P(U) + P(\neg K \mid V, \neg U) \cdot P(V) \cdot P(\neg U)) \\
&= \frac{1}{1 - 0.7} \cdot (0.1 \cdot 0.8 \cdot 0.5 + 0.4 \cdot 0.8 \cdot 0.5) = \frac{1}{0.3}(0.04 + 0.16) == \frac{0.2}{0.3} = \frac{2}{3}
\end{aligned}
$$

***Anmerkung**: Der Wert erscheint auf den ersten Blick hoch, er ist aber im Vergleich zur* a priori *Wahrscheinlichkeit von 0.8 für dieses Ereignis zu sehen.*

2–e Sind die Ereignisse *Vorlesung besucht* und *gute Klausurnote* konditional unabhängig gegeben das Ereignis *Übung besucht*? Wie können Sie das aus den berechneten und gegebenen Wahrscheinlichkeiten erkennen?

4 *Punkte*

Nein, weil $P(K|V) = 0.75 \neq 0.9 = P(K|V,U)$.

Aufgabe 3 Reinforcement Learning (18 Punkte)

Gegeben sei folgende deterministische 3×3 Welt:

a	b	c
d	e	f
g	h	**i**

Der Agent kann sich jeweils ein Feld nach unten, oben, links oder rechts bewegen, falls sich am Zielort ein Feld befindet. Sie können davon ausgehen, dass die Aktionen deterministisch sind.

Der Reward setzt sich aus 2 Komponenten zusammen:

- Wenn der Agent im Feld **i** landet, erhält er einen Reward von 10, auf jedem anderen Feld einen Reward von 0.
- Jede durchgeführte Aktion kostet aber eine Einheit, d.h. der Reward für das Zielfeld der Aktion vermindert sich um 1.

Im Feld **i** hat der Agent keine Aktionen zur Verfügung.

Es seien bereits einige $\hat{Q}$-Werte berechnet worden, die im folgenden graphisch angeordnet dargestellt sind:

a $\hat{Q}(a,r)=1$ $\hat{Q}(a,u)=5$	$\hat{Q}(b,l)=4$ **b** $\hat{Q}(b,r)=0$ $\hat{Q}(b,u)=5$	$\hat{Q}(c,l)=1$ **c** $\hat{Q}(c,u)=3$
$\hat{Q}(d,o)=1$ **d** $\hat{Q}(d,r)=4$ $\hat{Q}(d,u)=2$	$\hat{Q}(e,o)=1$ $\hat{Q}(e,l)=2$ **e** $\hat{Q}(e,r)=1$ $\hat{Q}(e,u)=0$	$\hat{Q}(f,o)=1$ $\hat{Q}(f,l)=2$ **f** $\hat{Q}(f,u)=5$
$\hat{Q}(g,o)=1$ **g** $\hat{Q}(g,r)=2$	$\hat{Q}(h,o)=1$ $\hat{Q}(h,l)=2$ **h** $\hat{Q}(h,r)=7$	**i**

3–a Aus der Vorlesung kennen Sie die Update-Formel für Q-Learning:

$$\hat{Q}(s,a) \leftarrow \hat{Q}(s,a) + \alpha[r(s,a) + \gamma \cdot \max_{a'} \hat{Q}(s',a') - \hat{Q}(s,a)]$$

Nehmen Sie an, die Lernrate und der Discount-Faktor seien beide gleich 1. Interpretieren Sie die resultierende Formel für obige Problemstellung.

3 Punkte

$\alpha = 1$ *und* $\gamma = 1 \Rightarrow \hat{Q}(s,a) \leftarrow r(s,a) + \max_{a'} \hat{Q}(s',a')$

Der neue Wert entspricht immmer dem um 1 verminderten Bewertung der besten Aktion im Nachfolgezustand. Er ist unabhängig vom momentanen Wert.

3–b Die momentane (suboptimale) Policy des Agenten führt ihn auf den Pfad a → b → e → f → **i**. Wie sehen die neuen $\hat{Q}$-Werte aus, wenn der Agent Q-Learning mit oben angegebenen Parametern zum Update der Werte verwendet?

6 Punkte

$$\begin{aligned}
\hat{Q}(a,r) &= -1 + 5 = 4 \\
\hat{Q}(b,u) &= -1 + 2 = 1 \\
\hat{Q}(e,r) &= -1 + 5 = 4 \\
\hat{Q}(f,u) &= (10 - 1) + 0 = 9
\end{aligned}$$

3–c Sie wissen aus der Vorlesung, dass der Wert eines Zustands das Maximum aller Q-Werte in diesem Zustand ist, i.e. $V^*(s) = \max_a Q(s, a)$.

Geben Sie für alle Zustände in obigem Problem die Werte an, die sich aus den Q-Werten nach Konvergenz von Q-Learning mit obigen Parametern ergeben. Der Wert des Felds **i** sei 10.

4 *Punkte*

Bei jedem Schritt verliert man eine Einheit. Das heißt, der Wert eines Zustands ist der Wert des Zielzustands vermindert um den kürzesten Weg zum Zielzustand, i.e. $V^*(s) = 10 - Ham(s, \boldsymbol{i})$*, wobei* $Ham(s_1, s_2)$ *die Hamming-Distanz (Manhattan-Distanz) zwischen den Zuständen* s_1 *und* s_2 *ist.*

6	*7*	*8*
7	*8*	*9*
8	*9*	*10*

3–d Welche der Updates, die Sie in Aufgabe b durchgeführt haben, hätten in unterschiedlichen $\hat{Q}$-Werten resultiert, wenn der Agent statt Q-Learning SARSA verwendet hätte? Warum?

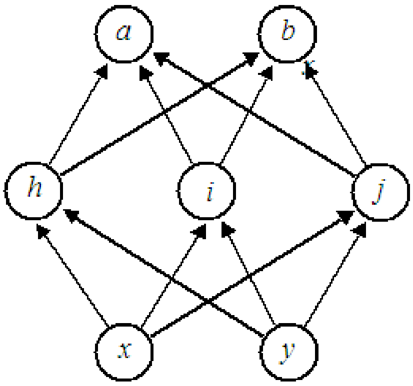
5 *Punkte*

SARSA führt ein on-policy update durch, d.h. statt des Maximums wird der Q-Wert genommen, der der nächsten Aktion entspricht. Auf dem Pfad $a \to b \to e \to f \to \boldsymbol{i}$ *ist e der einzige Zustand, in dem die Policy nicht die optimale Aktion ausgewählt hat. Daher würde der Update für* $\hat{Q}(b, u)$ *wie folgt aussehen:*

$$\hat{Q}(b, u) = r(s, u) + \hat{Q}(e, r) = -1 + 1 = 0$$

Aufgabe 4 Neuronale Netze (18 Punkte)

Gegeben sei folgendes Neuronales Netz mit der Identität als Aktivierungsfunktion, d.h. $g(x) = x$.



$W_{x,h} = 0.5$ $\quad W_{h,a} = -0.4$
$W_{x,i} = 0.3$ $\quad W_{i,a} = -0.9$
$W_{x,j} = -0.4$ $\quad W_{j,a} = -0.3$
$W_{y,h} = 0.8$ $\quad W_{h,b} = -0.5$
$W_{y,i} = 0.2$ $\quad W_{i,b} = 0.2$
$W_{y,j} = 0.5$ $\quad W_{j,b} = 0.3$

4–a Berechnen Sie die Outputs (a, b) für die Eingabe $x = 1$ und $y = -1$. Geben Sie auch alle relevanten Zwischenresultate an (z.B. die Aktivierung der Zwischenknoten).

5 *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 identische Aktivierungsfunktion läßt gibt die Aktivierungswerte unverändert weiter, d.h. $out_x = in_x$.

$$\begin{aligned} in_a &= W_{h,a} \cdot out_h + W_{i,a} \cdot out_i + W_{j,a} \cdot out_j = (-0.4) \cdot (-0.3) - 0.9 \cdot 0.1 + (-0.3) \cdot (-0.9) = 0.3 \\ in_b &= W_{h,b} \cdot out_h + W_{i,b} \cdot out_i + W_{j,b} \cdot out_j = (-0.5) \cdot (-0.3) + 0.2 \cdot 0.1 + 0.3 \cdot (-0.9) = -0.1 \end{aligned}$$

Die Ausgabewerte bleiben wiederum unverändert.

4–b Nehmen Sie nun an, dass das Netzwerk für obigen Input $(x, y) = (1, -1)$ die Ausgabe $(a, b) = (0.5, 0.9)$ liefern soll. Die Lernrate sei $\alpha = 0.5$.

1. Berechnen Sie die Fehlerterme Δ_a und Δ_b

   3 *Punkte* $g'(x) = 1$
   $\Delta_a = Err_a \cdot g'(in_a) = (0.5 - 0.3) \cdot 1 = 0.2$
   $\Delta_b = Err_b \cdot g'(in_b) = (0.9 - (-0.1)) \cdot 1 = 1$

2. Berechnen Sie die Fehlerrate Δ_h

   3 *Punkte*
   $\Delta_h = W_{h,a} \cdot \Delta_a \cdot g'(in_h) + W_{h,b} \cdot \Delta_b \cdot g'(in_h) = (-0.4) \cdot 0.2 + (-0.5) \cdot 1 = -0.08 - 0.5 = -0.58$

3. Berechnen Sie die Gewichtsänderung für das Gewicht $W_{h,a}$

   3 *Punkte*
   $W_{h,a} \leftarrow W_{h,a} + \alpha \cdot \Delta_a \cdot out_h == -0.4 + 0.5 \cdot 0.2 \cdot (-0.3) = -0.43$

4–c Angenommen, Sie können den Hidden Layer dieses Netzes beliebig vergrößern. Welche Art von Funktionen könnten Sie dann in den Outputs a und b zumindest lernen? Können Sie insbesondere eine XOR-Funktion lernen?

4 *Punkte*

Alle stetigen Funktionen. Für XOR wurde es in der Vorlesung gezeigt, daß es hier eine Lösung mit einem Hidden Layer gibt.

Anmerkung: *Genau genommen setzt obige Antwort auch voraus, daß man beliebige Aktivierungsfunktionen wählen kann, was in der Aufgabenstellung nicht angegeben war. Unter der Voraussetzung, daß die Aktivierungsfunktion lineaer bleiben muß, dann können nur* lineare Funktionen *modelliert werden.*

Aufgabe 5 Verschiedenes (24 Punkte)

5–a Sind die folgenden Aussage wahr oder falsch? Begründen Sie Ihre Antworten kurz.

1. Jede zulässige Heuristik ist auch konsistent.

   3 *Punkte*

   Nein. Aus Konsistenz folgt Zulässigkeit, aber nicht umgekehrt.

2. A^* ist vollständig.

   3 *Punkte*

   Ja, weil A^* findet eine existierende Lösung auf jeden Fall.

3. Wenn der A^*-Algorithmus den Zielzustand erreicht hat, bricht er sofort ab.

   3 *Punkte*

   Falsch. Es kann ja sein, dass meine Heuristik sich maßlos verschätzt hat und die realen Kosten für den aktuellen Knoten extrem hoch sind.

5–b Beantworten Sie folgende Fragen zu Graphplan:

1. Warum ist die Anzahl der Zustands-Literale von Schicht zu Schicht monoton steigend?

   3 *Punkte*

   Weil jedes einmal im Graphen vorhandene Literal durch die Persistenz-Aktionen in den folgenden Schichten erhalten bleibt.

2. Warum ist die Anzahl der Mutex-Verbindungen zwischen diesen Literalen fallend?

   4 *Punkte*

   Je mehr Literale vorhanden sind, desto mehr Aktionen werden möglich, und desto eher ist es möglich ein Paar von Literalen, die bisher einen Mutex-Link hatten, durch ein Aktionen-Paar zu erreichen, daß sich nicht gegenseitig ausschließt.

3. Was folgt aus diesen beiden Sachverhalten?

   3 *Punkte*

   Dass der Planungsgraph endlich ist, und somit sein Aufbau konvergiert.

5–c Halten Sie das Chinese Room Argument für ein sinnvolles Gegenargument für den Turing Test oder nicht? Diskutieren Sie Ihren Standpunkt in einigen Sätzen.

Hinweis: Sie müssen weder den Turing Test noch das Chinese Room Argument erklären, für diese Erklärungen gibt es keine Punkte.

5 *Punkte*