Fachhochschule Wedel
Prof. Dr. Uwe Schmidt

20. August 1999

Aufgaben zur Klausur **Expertensysteme** im SS 99 (II h763)

Zeit: 60 Minuten erlaubte Hilfsmittel: keine

Bitte tragen Sie Ihre Antworten und fertigen Lösungen ausschließlich an den gekennzeichneten Stellen in das Aufgabenblatt ein. Ist ihre Lösung wesentlich umfangreicher, so überprüfen Sie bitte nochmals Ihren Lösungsweg.

Sollten Unklarheiten oder Mehrdeutigkeiten bei der Aufgabenstellung auftreten, so notieren Sie bitte, wie Sie die Aufgabe interpretiert haben.

Viel Erfolg !

Diese Klausur besteht einschließlich dieses Deckblattes aus 8 Seiten

---

Aufgabe 1:

Transformieren Sie die folgende Formel des Prädikatenkalküls in Klauselform:

$$\neg\ (\exists\, x\ \bullet\ (q(x) \wedge \forall\, y\ \bullet\ (r(x) \Rightarrow p(x, y))))$$

Lösung:

. . . . . .

. . . . . .

. . . . . .

. . . . . .

Aufgabe 2:

Die folgende Abbildung zeigt die Anfangsbelegung eines Schiebepuzzles

Auf diesem Spielfeld gibt es 7 Felder, die mit 3 schwarzen Steinen (S) und 3 weißen Steinen (W) belegt sind, ein Feld ist frei. Man kann 3 Arten von Zügen machen:

1. ein Stein kann auf das freie Feld geschoben werden
2. ein Stein kann genau einen Stein überspringen, um in das freie Feld zu gelangen
3. ein Stein kann genau zwei Steine überspringen, um in das freie Feld zu gelangen

Ziel des Spiels ist es, alle weißen Steine links von allen schwarzen zu positionieren. Wo der freie Platz sich befindet ist dabei unwesentlich

Einen Zustand eines Spiels kann man mit folgender Variablen beschreiben

```
var
    state : array [1..7] of (schwarz, weiß, frei);
```

Beschreiben Sie die Menge der Endzustände mit Hilfe eine Prädikats über der Variablen *state* (einer Formel aus dem Prädikatenkalkül), d.h. welche Eigenschaft muß erfüllt sein, damit *state* einen Endzustand dieses Spiels enthält?

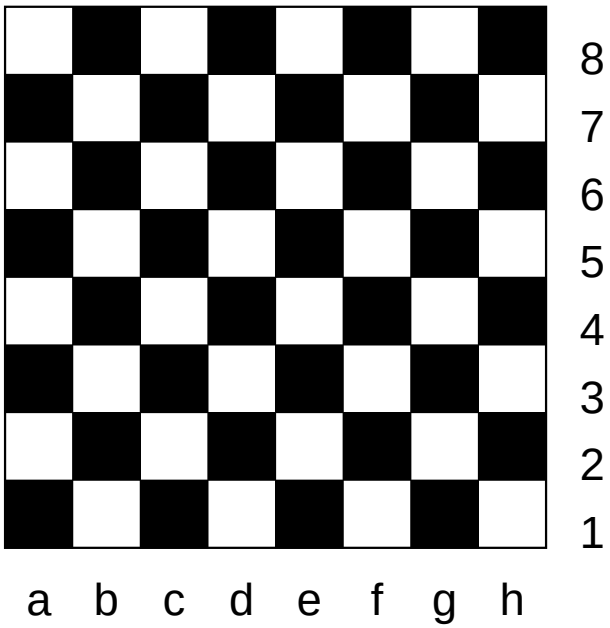
.....

.....

.....

Aufgabe 3:

Gegeben sei ein Schachbrett



Die einzelnen Felder werden beim Schach über ein Buchstaben–Zahl–Paar indiziert. Wir werden in dieser Aufgabe die Buchstaben durch die Zahlen 1 bis 8 ersetzen und ein Feld durch ein Zahlenpaar bestimmen, bei dem der 1. Index die Rolle des Buchstabens übernimmt.

Ziel der Aufgabe soll es sein, die Bewegungen von Figuren auf dem Brett mit Prolog-Regeln zu beschreiben.

Ein Prädikat zur Berechnung aller erreichbaren Felder mit einem Turm

$turmzug(+AltesFeld, -NeuesFeld)$

kann also mit den folgenden Regeln beschrieben werden, Feldpositionen werden hier durch eine 2–elementige Listen dargestellt:

```
                                                    /* waagerecht ziehen */
turmzug([X,Y],[Xn,Yn]) :-
        member(D,[1,-1,2,-2,3,-3,4,-4,5,-5,6,-6,7,-7]),    /* generieren */
        Xn is X + D,
        member(Xn,[1,2,3,4,5,6,7,8]),                      /* testen */
        Yn = Y.

                                                    /* senkrecht ziehen */
turmzug([X,Y],[Xn,Yn]) :-
        member(D,[1,-1,2,-2,3,-3,4,-4,5,-5,6,-6,7,-7]),    /* generieren */
        Yn is Y + D,
        member(Yn,[1,2,3,4,5,6,7,8]),                      /* testen */
        Xn = X.
```

Entwerfen sie für ein analog aufgebautes Prädikat
$laeuferzug(+AltesFeld, -NeuesFeld)$
die zugehörigen Regeln:

$laeuferzug([X,Y],[X_n,Y_n]) :-$

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

$laeuferzug([X,Y],[X_n,Y_n]) :-$

. . . . . .

. . . . . .

. . . . . .

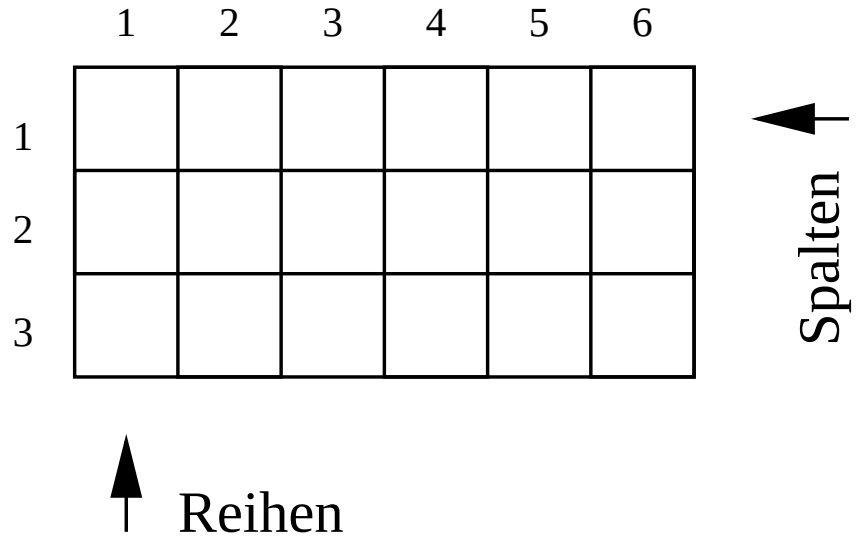
. . . . . .

. . . . . .

. . . . . .

Aufgabe 4:

Es soll ein Lastwagen mit Paletten beladen werden. Auf den Wagen passen genau 3 Paletten nebeneinander, 6 Paletten hintereinander. Die zur Verfügung stehenden Plätze sind in der folgenden Skizze veranschaulicht:



Auf den Wagen sollen die folgenden 8 Palletten geladen werden:

p1 soll als erste beladen werden, da sie als letzte ausgeliefert werden soll. Sie ist so schwer, daß keine andere Palette daneben paßt.

p2 soll als letzte beladen werden, da sie als erste ausgeliefert werden soll, sie ist ebenfalls so schwer, daß keine andere daneben paßt.

p3,p4,p5 von diesen Paletten paßt jeweils nur eine in eine Reihe, da zwei von diesen wieder zu schwer für eine Reihe sind.

p6 soll weiter hinten im Wagen plaziert sein als p3.

p7 soll weiter hinten im Wagen plaziert sein als p4.

p8 soll genau hinter p5 im Wagen plaziert sein.

Es soll die gesamte Länge der Ladefläche ausgenutzt werden.

Die Zuordnung der Paletten zu Stellplätzen soll mit Hilfe eines CLP Programms berechnet werden. In einem CLP Programm sind neben den üblichen Prolog–Prädikaten noch zusätzliche Prädikate für Einschränkungen eingebaut:

- die Operatoren $\#=$, $\#\backslash=$, $\#>$, $\#>=$, $\#<$, $\#=<$ für lineare Einschränkungen
- $L :: min..max$ zur Einschränkung der Wertemenge einer Liste von Variable auf ein Intervall $min..max$ (Beispiel: $[S, E, N, D] :: 0..9$)
- $alldistinct(L)$ zur Einschränkung, daß alle Variablen der Liste L mit unterschiedlichen Werten belegt sein müssen (Beispiel: $alldistinct([S, E, N, D])$)

Die Einschränkungen werden mit folgendem hier nicht vollständig angegebenen Prolog-Prädikat beschrieben, S_i sind die Spalten, R_j die Reihen, dieses sind die Constraint–Variablen:

$mkvariable(PL) :-$
$PL = [$
$p(p1, S1, R1),$
$p(p2, S2, R2),$
$p(p3, S3, R3),$
$p(p4, S4, R4),$
$p(p5, S5, R5),$
$p(p6, S6, R6),$
$p(p7, S7, R7),$
$p(p8, S8, R8)$
$],$
$nicht2PallettenAmGleichenPlatz(PL),$
$.../*zuentwickelndeEinschränkungen*/$
.

Entwickeln Sie die zusätzlich notwendigen Einschränkungen, oben mit ... gekennzeichnet. Das Prädikat dafür, daß 2 Palletten nicht am selben Platz stehen können, soll hier nicht entwickelt werden. Die Einschränkungen werden gruppiert:

Die Einschränkungen an die Wertebereiche der Constraint–Variablen:

.....

.....

.....

Die Einschränkungen an die Paletten p1 und p2:

.....

.....

.....

.....

Die Einschränkungen, daß Paletten p3, p4 und p5 nicht in eine Reihe passen:

.....

.....

.....

.....

Die Einschränkungen an die Paletten p6, p7 und p8.

.....

.....

.....

.....

Weitere notwendige Einschränkungen:

.....

.....

.....

.....
