Künstliche Intelligenz - Blatt 8

Lukas Härtel
Mat. Nr.: 210101282
Mail: lukashaertel@uni-koblenz.de

Aufgabe 8.1

Logisches Programm P:

$$
\begin{aligned}
&p \leftarrow \neg a, x.\\
&a \leftarrow b, not\ q, not\ s.\\
&j \leftarrow not\ y, r, s.\\
&k \leftarrow not\ l.\\
&t \leftarrow r, s, y.\\
&r.\\
&s.\\
&y.
\end{aligned}
$$

Aufgabenteil a)

Für $S = \{r, s, t\}$. ist das Redukt P^S:

$$P^S = \{p \leftarrow \neg a, x., j \leftarrow r, s., k., t \leftarrow r, s, y., r., s., y.\}$$

Minimales Modell von P^S ist $\{k, r, s, y, t, j\} \neq S$, damit ist S keine Antwortmenge.

Aufgabenteil b)

Setze $S := \{k, r, s, y, t, j\}$, dann ist das Redukt P^S:

$$P^S = \{p \leftarrow \neg a, x., j \leftarrow r, s., k., t \leftarrow r, s, y., r., s., y.\}$$

Minimales Modell von P^S ist $\{k, r, s, y, t, j\} = S$, damit Antwortmenge.

Aufgabe 8.2

Gegeben folgendes erweiterte logische Programm P:

$$
\begin{aligned}
a &\leftarrow not\ c \\
c &\leftarrow b, not\ q \\
c &\leftarrow not\ d, not\ e \\
e &\leftarrow a, not\ d \\
b.&
\end{aligned}
$$

Nach Umformung in Default-Logik:

$$
\begin{aligned}
\delta_1 &= \frac{\top : \neg c}{a} \\
\delta_2 &= \frac{b : \neg q}{c} \\
\delta_3 &= \frac{\top : \neg d, \neg e}{c} \\
\delta_4 &= \frac{b : \neg q}{c} \\
def(P) &= (\{b\}, \{\delta_1, \delta_2, \delta_3, \delta_4\})
\end{aligned}
$$

Aufgabe 8.3

Allgemeines Wissen Aus dem Übungstext übernommen. Erst werden Tageszeiten, Aufgaben und ihre Dauer, Voraussetzungen und bekannte Startzeiten angegeben.

Listing 1: problem.lp

```
% The usable hours in the day
day(9, 20).

% Known events, with their durations
event(geldautomat, 1).
event((einkaufen; ki_vorlesung; ki_uebung), 2).
event(grillfeier, 3).

% Dependencies between events
dependency(grillfeier, einkaufen).
dependency(einkaufen, geldautomat).
dependency(ki_uebung, ki_vorlesung).

% Known start times
starttime(ki_vorlesung, 10; ki_uebung, 17).
```

Abgeleitetes Wissen Berechnung der Endzeit einer Aufgabe, Constraints für Aufgabenvoraussetzung, Überprüfung auf Tageszeit, Vermeidung von Überlappung, sowie Generator für alle möglichen Startzeiten.

Überlappungen werden durch die belegten Stunden abgefangen, wie durch occupied/2 vorbereitet und das letzte Constraint angegeben.

Listing 2: problem.lp (contd.)

```
16 % The endtime is the start time plus the duration
17 endtime(X, S + D) :-
18     starttime(X, S), event(X, D).
19
20 % The hours of a day that are occupied by an event
21 occupied(X, S .. (E - 1)) :-
22     starttime(X, S), endtime(X, E).
23
24 % For a dependency from X to Y, X needs to happen after Y has ended
25 :- dependency(X, Y), starttime(X, S), endtime(Y, E), S < E.
26
27 % An event may not happen outside of the hours of a day
28 :- starttime(_, X), day(Y, _), X < Y.
29 :- endtime(_, X), day(_, Y), X > Y.
30
31 % Two different events may not overlap
32 :- occupied(X, T), occupied(Y, T), X != Y.
33
34 % Starttime of an event can be any time of the day, but only one assignment
35 { starttime(X, Y) : Y = S .. (E - 1) } == 1 :-
36     event(X, _),
37     day(S, E).
38
39 #show starttime/2.
```

Resultat Nach Berechnung über gringo problem.lp | clasp > result.txt ergeben sich die folgenden Belegungen.

9 Uhr Geldautomat

10 Uhr KI-Vorlesung

12 Uhr Einkaufen

14 Uhr Grillfeier

17 Uhr KI-Übung

Listing 3: result.txt

```
clasp version 3.1.4
Reading from stdin
Solving...
Answer: 1
starttime(ki_vorlesung,10) starttime(ki_uebung,17) starttime(grillfeier,14) starttime(einkaufen,12) starttime(geldautomat,9)
SATISFIABLE

Models       : 1+
Calls        : 1
Time         : 0.007s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.016s
```