

## BLATT 7

DANIEL SCHMIDT & PAMELA FLEISCHMANN

**Aufgabe 1.** Sei das folgende Datalog-Programm gegeben:

$$\begin{aligned} gn(X, Y) &: -gl(X, Y). \\ gn(X, Y) &: -kp(X, X1), gn(Y1, X1), kp(Y, Y1). \end{aligned}$$

und die Zielklausel  $? - gn(c, Y)$ .

Schritt 1:

$$\begin{aligned} r_0 &= query^f(Y) : -gn^{bf}(c, Y). \\ r_1 &= gn^{bb}(X, Y) : -gl(X, Y). \\ r_2 &= gn^{bf}(X, Y) : -kp(X, X1), gn^{bb}(Y1, X1), kp(Y, Y1). \\ r_3 &= gn^{fb}(X, Y) : -kp(X, X1), gn^{bb}(Y1, X1), kp(Y, Y1). \end{aligned}$$

Schritt 2:

$$\begin{aligned} magic_{r_0-gn^{bf}}(c) &: -. \\ magic_{r_2-gn^{fb}}(Y) &: -gn^{bb}(Y1, X1), kp(Y, Y1). \\ magic_{r_3-gn^{bf}}(X) &: -kp(X, X1), gn^{bb}(Y1, X1). \end{aligned}$$

Schritt 3:

$$\begin{aligned} query^f(Y) &: -magic_{r_0-gn^{bf}}(c), gn^{bf}(c, Y). \\ gn^{bf}(X, Y) &: -gl^{bf}(X, Y). \\ gn^{bf}(X, Y) &: -magic_{r_2-gn^{fb}}(Y), gn^{bf}(Y1, X1), kp(Y, Y1). \\ gn^{fb}(X, Y) &: -magic_{r_3-gn^{bf}}(X), kp(X, X1), gn^{fb}(Y1, X1). \end{aligned}$$

Schritt 4:

$$\begin{aligned} magic_{gn^{bf}}(c) &: -magic_{r_0-gn^{bf}}(c). \\ magic_{gn^{bf}}(X) &: -magic_{r_3-gn^{bf}}(X). \\ magic_{gn^{fb}}(Y) &: -magic_{r_2-gn^{fb}}(Y). \end{aligned}$$

Resultat:

$$\begin{aligned}
query^f(Y) &: -magic\_gn^{bf}(c), gn^{bf}(c, Y). \\
magic\_gn^{bf}(c) &: -magic\_r_0\_gn^{bf}(c). \\
magic\_gn^{bf}(X) &: -magic\_r_3\_gn^{bf}(X). \\
magic\_gn^{fb}(Y) &: -magic\_r_2\_gn^{fb}(Y). \\
gn^{bf}(X, Y) &: -gl^{bf}(X, Y). \\
gn^{bf}(X, Y) &: -magic\_gn^{fb}(Y), gn^{bf}(Y1, X1), kp(Y, Y1). \\
gn^{fb}(X, Y) &: -magic\_gn^{bf}(X), kp(X, X1), gn^{fb}(Y1, X1).
\end{aligned}$$

**Aufgabe 2.** a. Betrachte das Datalog-Programm

$$p(a). q(X) : -\neg p(X).$$

Herbrand-Modelle für dieses Programm sind  $\{p(a)\}$ ,  $\{p(a), q(b)\}$  und  $\{p(a), q(a), q(b)\}$ .  $r(x, y)$  kann außerdem immer mit dazugenommen werden, da es in der Regel nicht vorkommt. Die Variable  $X$  ist unbeschränkt, da sie nicht-negiert nur im Kopf vorkommt. Desweiteren ist  $p$  geschichtet, da es im Abhängigkeitsgraphen keinen Zyklus gibt.

b. Betrachte das Datalog-Programm

$$p(X) : -\neg q(X). q(X) : -\neg p(X).$$

Herbrand-Modelle sind  $\{q(a), q(b)\}$ ,  $\{p(a), p(b)\}$ ,  $\{q(a), p(b)\}$  und  $\{q(b), p(a)\}$ . Analog zu a. kann  $r$  in allen Varianten wieder dazugenommen werden.  $X$  ist in beiden Regeln unbeschränkt. Das Programm ist nicht geschichtet, da der Abhängigkeitsgraph einen Zyklus mit einer mit  $\neg$  beschrifteten Kante enthält.

c. Betrachte das Datalog-Programm

$$p(X) : -\neg q(X). q(X) : -p(X).$$

Herbrand-Modell ist  $\{p(a), q(a), p(b), q(b)\}$  und wieder kann  $r$  in allen Varianten dazugenommen werden.  $X$  ist in Regel 1 unbeschränkt, in Regel 2 dagegen beschränkt, da sie im Kopf und im Rumpf nicht-negiert vorkommt. Analog zu b. ist  $P$  nicht geschichtet.

d. Betrachte das Datalog-Programm

$$p(a). q(b). r(X, Y) : -\neg p(X), \neg q(Y).$$

Herbrand-Modelle sind  $\{p(a), q(b), r(b, a)\}$ ,  $\{p(a), q(b), p(b)\}$ ,  $\{p(a), q(b), q(a)\}$ . Beide Variablen sind unbeschränkt, da sie im Rumpf nur negiert vorkommen. Da es keinen Zyklus im Abhängigkeitsgraphen gibt, ist das Programm geschichtet.

e. Betrachte das Datalog-Programm

$$p(a). q(b). r(X, Y) : -p(X), q(Y), \neg p(X).$$

Ein Herbrand Modell ist  $\{p(a), q(b)\}$ . Da der Rumpf nie erfüllt ist, können alle Varianten von  $r$  in das Modell mit aufgenommen werden. Die Variable  $X$  ist unbeschränkt, die Variable  $Y$  ist beschränkt. Das Programm ist geschichtet, da es keinen Zyklus gibt.

**Aufgabe 3.** Betrachte das folgende Datalog-Programm mit Negation

$$\begin{aligned} & p(a, b). p(b, c). p(b, a). p(c, d). p(c, c). \\ & q(X, Y) : \neg p(X, Y), \neg p(Y, X). \\ & q(X, X) : \neg \neg q(X, Y). \\ & r(X, Y) : \neg \neg q(Y, X), p(X, Y), \neg p(X, X). \\ & r(Y, Y) : \neg r(X, Y). \\ & s(X, X) : \neg \neg r(X, Y), p(X, Z), p(W, Y). \end{aligned}$$

Für ein perfektes Modell wird die Schichtung betrachtet und somit zuerst das Prädikat  $q$ . Mit der ersten Regel kommen  $q(c, b)$  und  $q(d, c)$  hinzu. Mit der zweiten dann  $q(c, c), q(b, b), q(d, d)$ . Da  $r$  von  $p$  und  $q$  abhängt, ist  $r$  in der folgenden Schicht und es werden  $r(a, b)$  und  $r(b, a)$  durch Regel 3, sowie  $r(b, b)$  und  $r(a, a)$  durch Regel 4 hinzugenommen. Da  $s$  nicht von  $r$  abhängt, ist  $s$  in derselben Schicht und es kommen die Fakten  $s(a, a), s(b, b), s(c, c)$  hinzu. Da in keiner Regel, die weiter unten steht, Elemente der oberen Schichten verändert werden, ist dies das perfekte Modell.

**Aufgabe 4.** ad a.

Die Ausgaben des Original Programms sind wie folgt:

```
ready>>?notintoys(susan).
CORAL::Warning : Using underbound method 0 for notintoys 1!
(Number of Answers = 0)
ready>>?notintoys(john).
CORAL::Warning : Using underbound method 0 for notintoys 1!
yes.
... next answer ? (y/n/all)[y] all
(Number of Answers = 1)
```

Folgende Anpassungen an das Programm waren notwendig:

```
module flounder_example.
export notintoys(f).

person(john).
person(susan).

employed(susan, marketing).

notintoys(X) :- person(X), Y = toys, not employed(X,Y).
```

end\_module.

Die resultierende Ausgabe ist die folgende:

```
ready>>consult(flounder.P).
ready>>?notintoys(susan).
CORAL::Warning : Using underbound method 0 for notintoys 1!
yes.
```

```
... next answer ? (y/n/all)[y] all
(Number of Answers = 1)
```

ad b.

Das Programm sieht wie folgt aus:

```
module serie.
export zshg(bf,bb).
```

```
zshg(G, true) :- allNodes(G, LA), allConnectedNodes(G, LC), sameLength(LA,
```

```
sameLength(LA, LC) :- length(LA, LLA), length(LC, LLC), LLA=LLC.
```

```
connected(K1, K2) :- K1=K2.
connected(K1, K2) :- kante(K1, K2).
connected(K1, K2) :- kante(K2, K1).
```

```
path(K1, K2) :- connected(K1, K2).
path(K1, K3) :- connected(K1, K2), connected(K2, K3), not K1=K3, not K1=K
```

```
allNodes(G, []).
allNodes(G, [H|T]) :- node(G, H), not member(H, T).
```

```
reachableNodes(G, X, []).
reachableNodes(G, X, [H|T]) :- node(G, H), connected(X, H), not member(H,
```

```
allConnectedNodes(G, []).
allConnectedNodes(G, [H|T]) :- node(G, H), not member(H, T), reachableNod
```

end\_module.

Die folgende Ausgabe wird erzeugt:

```
ready>>consult(graphen.T).
ready>>consult(zshg.P).
ready>>?zshg(g1, X).
```

```
CORAL :: Warning — unable to find relation: node
Warning/error flagged in function Find_External_Relation()
Arity of relation = 2
```

```
Allocating new relation.
X=true.
    ... next answer ? (y/n/all)[y]y
(Number of Answers = 1)
ready>>?zshg(g2, X).
X=true.
    ... next answer ? (y/n/all)[y]y
(Number of Answers = 1)
ready>>?zshg(g6, X).
X=true.
    ... next answer ? (y/n/all)[y]y
(Number of Answers = 1)
```

Leider ist dies nicht die erwartete Ausgabe, g6 müsste eigentlich False ergeben.