8. Übung (7.–11. Juni 2021)

Programmierung

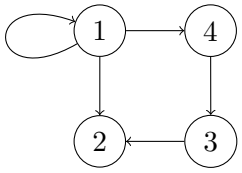
Hinweise. In zwei der folgenden Aufgaben sollen Prolog$^-$-Programme implementiert werden. Da Prolog$^-$ eine Teilsprache von Prolog ist, können Sie Ihre Programme mit einem Prolog-Interpreter (z.B. `swipl`) überprüfen. Speichern Sie dazu die definierten Klauseln im Klartext und laden diese Definitionen mit `swipl`, z.B. auf der Kommandozeile mit `swipl (Pfad zur Datei)`. Danach können Sie die Gültigkeit bzw. gültige Variablenbelegungen für eingegebene Goals untersuchen.

Achtung. Die Evaluationsstrategie von Prolog sorgt unter gewissen Umständen dafür, dass `swipl` für formal korrekte Programme und erfüllbare Goals keine SLD-Refutation findet. Beispielsweise kann für das Goal `p(0).` im folgenden Programm keine SLD-Refutation gefunden werden, obwohl es offensichtlich gültig ist:

```
p(X) :- p(X).
p(X).
```

Die Programme lassen sich üblicherweise aber umformulieren, sodass das nicht der Fall ist, z.B. durch Änderung der Reihenfolge der Klauseln im Programm, oder der Literale auf den rechten Seiten der Klauseln.

Aufgabe 1 (AGS 13.6 ⋆)



Gegeben sei der links abgebildete Graph.

(a) Bilden Sie die Kantenrelation von G durch ein zweistelliges Prädikat `edge` in Prolog ab.

(b) Seien u und w Knoten in G. Es gibt einen *Pfad von u nach w*, wenn (i) $u = w$ gilt, oder (ii) es einen Knoten v gibt, sodass eine Kante von u nach v und ein Pfad von v nach w existieren. Bilden Sie das Konzept Pfad von u nach w mit einem zweistelligen Prädikat `path` in Prolog ab. Nutzen Sie dazu das Prädikat `edge` aus Aufgabe (a).

(c) Geben Sie alle SLD-Refutationen für `?- path(4, X).` an. Geben Sie dabei die Belegungen für alle Variablen an. Geben Sie die Menge der möglichen Belegungen für `X` an!

Aufgabe 2 (AGS 13.8)

Natürliche Zahlen stellen wir in Prolog$^-$ als Terme über dem einstelligen Funktionssymbol `s` und dem nullstelligen Funktionssymbol `0` dar:

```
1 nat(0).
2 nat(s(X)) :- nat(X).
```

Dabei kürzen wir wie in der Vorlesung den Term für die natürliche Zahl `n` mit `<n>` ab, z.B. `s(s(s(0)))` = `<3>`. Weiterhin wurde das Prädikat `sum` besprochen:

```
3 sum(0, Y, Y) :- nat(Y).
4 sum(s(X), Y, s(S)) :- sum(X, Y, S).
```

(a) Geben Sie ein Prädikat `even` an, das für alle geraden natürlichen Zahlen gilt und für alle ungeraden natürlichen Zahlen nicht.

(b) Geben Sie eine zweistellige Relation `div2` an, die für jede natürliche Zahl `n` das Paar (`<n>`, $\texttt{<}\lfloor\frac{\texttt{n}}{2}\rfloor\texttt{>}$) enthält und sonst nichts.

(c) Geben Sie eine SDL-Refutation für `?- div2(<3>, <1>).` an.

(d) *(Zusatzaufgabe)* Geben Sie eine dreistellige Relation `div` an, die für jedes Paar von natürlichen Zahlen `n` und `m`, wobei $\texttt{m} \neq 0$, das Tripel (`<n>`, `<m>`, $\texttt{<}\lfloor\frac{\texttt{n}}{\texttt{m}}\rfloor\texttt{>}$) enthält und sonst nichts. Nutzen Sie dafür die dreistellige Relation `sum` aus der Vorlesung.
Hinweis: Nutzen Sie die Relation `lt` aus Zusatzaufgabe 1.

(e) *(Zusatzaufgabe)* Geben Sie eine SDL-Refutation für `?- div(<3>, <2>, <1>).` an.

Zusatzaufgabe 1 (AGS 13.16 a,b ⋆)
Natürliche Zahlen stellen wir in Prolog$^-$ als Terme über dem einstelligen Funktionssymbol `s` und dem nullstelligen Funktionssymbol `0` dar:

```
1 nat(0).
2 nat(s(X)) :- nat(X).
```

Dabei kürzen wir wie in der Vorlesung den Term für die natürliche Zahl `n` mit `<n>` ab, z.B. `s(s(s(0)))` = `<3>`.

(a) Gegeben seien die dreistellige Relation `sum`, die alle Tripel `(<X>, <Y>, <Z>)` von natürlichen Zahlen enthält, sodass $\texttt{X} + \texttt{Y} = \texttt{Z}$ gilt, und die zweistellige Relation `lt`, die alle Paare `(<X>, <Y>)` von natürlichen Zahlen enthält, sodass $\texttt{X} < \texttt{Y}$ gilt:

```
3 sum(0, X, X) :- nat(X).
4 sum(s(X), Y, s(Z)) :- sum(X, Y, Z).
5
6 lt(0, s(X)) :- nat(X).
7 lt(s(X), s(Y)) :- lt(X, Y).
```

Programmieren Sie in Prolog$^-$ eine dreistellige Relation `mod`, die alle Tripel `(<X>, <Y>, <Z>)` von natürlichen Zahlen enthält, sodass `Z` der ganzzahlige Rest der Division von `X` durch `Y` ist, d.h. $\texttt{X} \equiv \texttt{Z} \mod \texttt{Y}$ gilt. *Hinweis:* verwenden Sie die oben definierten Relationen `sum` und `lt`.

(b) Programmieren Sie in Prolog$^-$ eine einstellige Relation `noprime`, die alle natürlichen Zahlen enthält, die nicht prim sind. Eine natürliche Zahl `X` ist nicht prim, wenn $\texttt{X} \leq 1$ oder, wenn es eine natürliche Zahl `Y` mit $1 < \texttt{Y} < \texttt{X}$ gibt, sodass `X` restlos durch `Y` teilbar ist. *Hinweis:* verwenden Sie die Relationen aus Teilaufgabe (a) (einschließlich `mod`).

(c) Bestimmen Sie für das Goal `?- mod(<3>, Y, <1>).` eine Belegung der Variablen `Y` mittels SLD-Refutation. Notieren Sie in jedem Schritt die verwendete Programmzeile. Mehrere Resolutionsschritte unter Anwendung derselben Programmzeile können Sie mit `?-*` zusammenfassen.

Zusatzaufgabe 2 (AGS 13.2 $\star$)

Gegeben ist folgender Prolog$^-$-Code.

```
1   nat(0).
2   nat(s(X)) :- nat(X).
3
4   sum(0, X, X) :- nat(X).
5   sum(s(X), Y, s(Z)) :- sum(X, Y, Z).
6
7   prod(0, X, 0) :- nat(X).
8   prod(s(X), Y, Z) :- prod(X, Y, W), sum(Y, W, Z).
```

Geben Sie eine SLD-Refutation für `?- prod(s(s(0)), s(0), X).` an.