12. Übung Programmierung

Dominic Deckert

17. Juli 2017

Previously on ...

- H_0 (Tailrekursion)
- Umwandlungen $H_0 \leftrightarrow C_0 \leftrightarrow AM_0$

a)

```
int main()
  int x1, x2;   scanf("%d", &x1);
  x2 = 1;
  while(x1 > 0)
    x2 = x2 * 2;
    x1 = x1 - 1;
  printf("%d"; x2);
  return 0;
```

b)

A
```
function = 2;
flag = 1;
scanf("%d", &x1);
x2 = x1;
```
B
```
x1 > 0
```
C
```
x1 = x1 - x2;
x2 = x2 + 1;
```
D
```
x1 = x2;
x2 = x2 + 1;
```

b)

E
```
if(x1 == x2){
  result = x1;
  function = 0;}
else{
  result = x2;
  function = 0;}
```

Logik-Programmierung

Logik-Programmierung: “Programme” in Prädikatenlogik
beschreiben Objekte, mögliche Relationen zwischen Objekten
Programm stellt bekanntes Wissen dar
Form: bekannte Fakten / Regeln

Aufgabe 1

siehe “*ancestry*”-Datei

SLD-Refutation

Logische Anfrage soll geprüft werden (v.A. erfüllende Variablenbelegungen)
SLD-Refutation: Suche Erklärung für Aussage in Programm
-Unifikation mit Kopf einer Regel: ersetze mit Regelkörper & wende Unifikator an
-Unifikation mit Fakt: wende Unifikator an & entferne verwendete Aussage

Aufgaben 2, 3

siehe "*sld*"-Datei
Format: Zeile = Menge zu erklärender Aussagen
Angewandter Unifikator notiert (falls vorher bekannte Variablen ersetzt)
Immer neue Variablen verwenden!