Blatt 12

Aufgabe 1:

Was macht der μ-Operator?
Er berechnet einfach das, was auf S. 114 steht:
geg. $F(n, x_1, \dots, x_k)$ ist eine prim. rek. Fkt.
Dann ist

$$\mu F(n, x_1, \dots, x_k) = g(x_1, \dots, x_k) = \min \left\{ n \,\middle|\, \begin{array}{l} F(n, x_1, \dots, x_k) = 0 \\ \forall m < n \quad F(m, x_1, \dots, x_k) \text{ ist def.} \end{array} \right\}$$

Er berechnet also das Minimum für n.
Da n nicht nach oben beschränkt ist, kann das beliebig lange dauern. Deshalb lässt sich der μ-Operator auch nur mithilfe eines While-(= Goto- = Turing-) Programms beschreiben:

$x_0 := 0;\ y := F(0, x_1, \dots, x_n);$
WHILE $y \neq 0$ DO
$\quad x_0 := x_0 + 1;$
$\quad y := F(x_0, x_1, \dots, x_n)$
END

und nicht mithilfe eines LOOP-Programms ($\hat{=}$ prim. rek.).
Deshalb erweitert die μ-Operation auch die primitiv rekursiven Programme:

$$\mu\text{-rekursiv} \supsetneq \text{primitiv rekursiv}$$

Beispielsweise könnten wir mithilfe des μ-Operators das Problem lösen:
Welches ist die kleinste durch 23 teilbare Zahl?
Gerade $\mu F(x)$ mit $F(x) = x \bmod 23$.
(Ja, das Beispiel ist total sinnlos... es soll's nur veranschaulichen...)

Aufgabe 1:

a) $\lceil n^{\frac{3}{2}} \rceil = \min\{ m \leq n^3 \mid m^2 \geq n^3 \}$

$\Rightarrow \mu_{m \leq n^3}(F) = \min_{m \leq n^3}\{ m \leq n^3 \mid F(m,n) = 0 \}$

$f(m,n) \quad n^3 - m^2$

Die Exponentialfunktion ist, wie wir auf Blatt 11 gesehen haben primitiv rekursiv. Eigentlich sollte an der Stelle der n^3 besser $\exp(n,3)$ stehen:

$\lceil n^{\frac{3}{2}} \rceil = g(\exp(n,3))$

$g(y) = \mu_{x \leq y}(F) = \min_{x \leq y}\{ x \leq y \mid f(y, \exp(x,2)) = 0 \}$

$F(y,x) = y - x$

b) Wie der Name schon sagt, ist der beschränkte μ-Operator beschränkt. Das heisst, es reicht, wenn unsere Berechnung bis zur Schranke geht, was sich im Folgenden LOOP-Programm ausdrücken lässt:

$\mu_{n \leq y}(F)$:

```
n := 0;
LOOP y+1 DO
        IF  F(n, x_1, ..., x_k) ≠ 0 THEN n := n+1 END
END
```

Wg. Satz 5.113 LOOP-berechenbar $\Leftrightarrow$ prim. rek. entsprechen die beschränkt μ-rekursiven Funktionen gerade der Klasse der primitiv rekursiven.

Aufgabe 2:

a)
$$
\begin{aligned}
A(1,1) &= A(0, A(1,0)) \\
&= A(0, A(0,1) - 1) \\
&= A(0, 2 - 1) \\
&= A(0, 1) = 1 + 1 = 2
\end{aligned}
$$

$$A(1,2) = A(0, A(1,1)) = A(0,2) = 3$$

$$
\begin{aligned}
A(3,2) &= A(2, A(3,1)) \\
&= A(2, A(2, A(3,0))) \\
&= A(2, A(2, A(2,1) - 1))
\end{aligned}
$$

$$
\begin{aligned}
A(2,1) &= A(1, A(2,0)) \\
&= A(1, A(1,1) - 1) \\
&= A(1,1) \\
&= 2
\end{aligned}
$$

$$
\begin{aligned}
&= A(2, A(2,1)) \\
&= A(2,2) \\
&= A(1, A(2,1)) \\
&= A(1,2) = 3
\end{aligned}
$$

b)
$$
\begin{aligned}
A(1,y) &= A(0, A(1, y-1)) \\
&= A(0, A(0, A(1, y-2))) \\
&= \underbrace{A(0, \dots A(0,}_{y\text{-mal}} \underbrace{A(1,0))}_{1} \dots) \\
&= 1 + y \cdot 1
\end{aligned}
$$

$$
\begin{aligned}
&= A(0, A(1, y-1)) \\
&= A(1, y-1) + 1 \\
&= A(1, y-2) + 1 + 1 \\
&= A(1,0) + y \cdot 1 \\
&= 1 + y
\end{aligned}
$$

c) $A(2,y) = A(1, A(2, y-1))$

$= \underbrace{A(1, \dots A(1,}_{y\text{-mal}} \underbrace{A(2,0)}_{1})\dots)$

$= y+1$

$A(3,y) = \underbrace{A(2, \dots A(2,}_{y\text{-mal}} \underbrace{A(3,0)}_{1})\dots)$

$= y+1$

Wenn man die Regeln dieser „modifizierten „Ackermann Funktion" anschaut, sieht man, dass gilt:

$A(x,y) = y+1$

und das ist natürlich LOOP-berechenbar und somit primitiv rekursiv...

Aufgabe 3:

$A = \{F(0), F(1), F(2), \dots\}$

$F: \mathbb{N} \to \Sigma^*$ ist berechenbar und monoton wachsend

1) A endlich ⇒ entscheidbar, weil einfach Wort, das entschieden werden soll, mit den endlich vielen aus A vergleichen

2) A unendlich ⇒ entscheidbar, weil:

$w \in A$?

```
n := 0; drin := false;
WHILE F(n) ≤ w DO
    IF F(n)=w THEN drin:=true;
    n := n+1
END
```

Es reicht, solange die F(n) zu berechnen, bis $F(n) > w$. Dann war das w entweder „drin" oder nicht..

Aufgabe 4:

$F: \mathbb{N} \to \mathbb{N}$ mit $2^{F(2^n)} > a(n,n)$ $\forall$ $n > 0$

Kann F primitiv rekursiv sein?

Betrachten wir dazu den Beweis des LEMMA E zur Ackermann-Funktion auf S. 118:

LEMMA E: Für jedes LOOP-Programm P gibt es eine Konstante k, so dass für alle n gilt: $f_P(n) < a(k,n)$

Das heisst, ab einem bestimmten Punkt wächst die Ackermann-Funktion stärker, als jede LOOP-berechenbare - also primitiv rekursive - Funktion.

$\Rightarrow$ F kann nicht prim. rek. sein, denn sobald n die Konstante k* überschreitet wäre sonst $2^{F(2^n)} < a(n,n)$.

*durch 2^F gölte: die Konstante muss größer als $2^{\text{konst. von } F}$ sein.

Da das Argument aber ebenso 2^n ist gilt „<" ab $n > k_F$ ...

Aufgabe 5:

markierte Vereinigung: $A \oplus B := \{0w \mid w \in A\} \cup \{1w \mid w \in B\}$

a) Wenn das Wort mit 0 beginnt, „verfüttere" es an den Entscheidungsalgorithmus für A, ansonsten an den für B ...

Sei $h(w) = 1.$ Symbol von w
$t(w) =$ „der Rest"

$$\Rightarrow \chi_{A \oplus B} = \begin{cases} \chi_A(t(w)), & \text{für } h(w) = 0 \\ \chi_B(t(w)), & \text{sonst} \end{cases}$$

b) $\chi_A(w) = \chi_{A \oplus B}(0w)$

$\chi_B(w) = \chi_{A \oplus B}(1w)$

c) Nein, weil es keinerlei Möglichkeit mehr gibt, festzustellen, ob w aus A oder B kam ...
χ_A und χ_B sind somit nicht mehr „angebbar".
Gegenbsp.: Σ^* ist entscheidbar ($\chi = 1$).

Sei A unentscheidbar.

$A \cup \Sigma^* = \Sigma^*$

$\Rightarrow A$ immer noch unentscheidbar...