Automaten und Formale Sprachen Praktikum

Blatt 6

Louis Kuhnt - 263622
Matthias Ngo - 263248
AIN3 - SE

HS Furtwangen

Automaten und Formale Sprachen
Prof. Dr. Bernhard Hollunder

WISE2020 - 14. Dezember 2020

Aufgabe 1

Beweisen Sie folgende Eigenschaften:

1. $\mathbb{Z}$ ist abzählbar. Geben Sie eine geeignete Funktion $f : \mathbb{Z} \to \mathbb{N}$ an, die jedem $z \in \mathbb{Z}$ eineindeutig eine natürliche Zahl $n \in \mathbb{N}$ zuordnet.
2. $\mathbb{N} \times \mathbb{N}$ ist abzählbar. Geben Sie eine geeignete Funktion $f : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ an, die jedem Tupel $(m, n) \in \mathbb{N} \times \mathbb{N}$ eineindeutig eine natürliche Zahl $o \in \mathbb{N}$ zuordnet.
3. Sei Σ ein endliches Alphabet. Geben Sie eine Funktion $f : \Sigma^* \to \mathbb{N}$ an, die jedem $w \in \Sigma^*$ eineindeutig eine Zahl zuweist (d.h. Σ^* ist abzählbar).

Aufgabe 1.1

Ganze Zahlen (Z)	…	-3	-2	-1	0	1	2	3	…
Natürlichen Zahlen (N)	…	5	3	1	0	2	4	6	…

Gesucht: f(z) = n

Funktion mit Fallunterscheidung:

Falls z ist positiv: f(z) = z * 2
Falls z ist negativ: f(z) = z * (-2) -1
Falls z = 0: f(z) = z

Beispiele:

f(1) = 1 * 2 = 2;
f(2) = 2 * 2 = 4;
f(3) = 3 * 2 = 6;
f(0) = 0;
f(-1) = -1 * (-2) -1 = 1;
f(-2) = -2 * (-2) -1 = 3;
f(-3) = -3 * (-2) -1 = 5;

Aufgabe 1.2

(m,n)	0	1	2	3	4	5	6	7
0	0	2	5	9	14	20	..	
1	1	4	8	13	19	..		
2	3	7	12	18	..			
3	6	11	17	..				
4	10	16	..					
5	15	..						
6	..							
7								

Gesucht: f(m,n) = o;

Funktion: f(m,n) = n + ½ * (m + n) * (m + n + 1)

Beispiele:

f(0,0) = 0 + ½ * (0 + 0) * (0 + 0 + 1) = 0
f(0,1) = 1 + ½ * (0 + 1) * (0 + 1 + 1) = 2
f(1,0) = 0 + ½ * (1 + 0) * (1 + 0 + 1) = 1
f(1,1) = 1 + ½ * (1 + 1) * (1 + 1 + 1) = 4
f(0,2) = 2 + ½ * (0 + 2) * (0 + 2 + 1) = 5
f(2,0) = 0 + ½ * (2 + 0) * (2 + 0 + 1) = 3
f(m,n) = n + ½ * (m + n) * (m + n + 1)
…

Aufgabe 1.3

wort	N
e	0
a	1
b	2
c	3
aa	4

ab	5
ac	6
ba	7
bb	8
bc	9
ca	10

cb	11
cc	12
aaa	13
…	…

Funktion:

$$f(w) = \sum_{i=0}^{k} n^i * f(u_i)$$

Variablen Definition:

k: ist die Länge eines Wortes
n: ist die Anzahl der Elemente in der Menge/Alphabet
f(u) ist eine Funktion, die den Index des Elements zurückgibt

Beispiel:
- Alphabet = {a, b, c}
- n = 3
- f(u): z.B. f(a) = 1, f(b) = 2, f(c) = 3

Beispielrechnung:
f(a) = n^0 * f(a) = 3^0 * 1 = 1
…
f(aa) = n^0 * f(a) + n^1 * f(a) = 3^0 * 1 + 3^1 * 1 = 4
f(ab) = n^0 * f(a) + n^1 * f(b) = 3^0 * 2 + 3^1 * 1 = 5
f(ac) = n^0 * f(a) + n^1 * f(c) = 3^0 * 3 + 3^1 * 1 = 6
…
f(aaa) = n^0 * f(a) + n^1 * f(a) + n^2 * f(a) = 3^0 * 1 + 3^1 * 1 + 3^2 * 1 = 13

Aufgabe 2

Schreiben Sie jeweils ein loop-Programm für folgende Aufgaben:

1. Setze den Inhalt des Registers A auf den Wert 5.

```
whilenot iszero(A) do
        A := pred(A);          //Register A wird auf 0 gesetzt
od;

A := succ(A);
A := succ(A);
A := succ(A);                  //Register A wird den Wert 5 zugewiesen
A := succ(A);
A := succ(A);
```

Ergebnis: Register A = 5;

2. Addiere den Inhalt von Register B auf den Inhalt von Register A.

Register B = 3; Register A = 2;

```
loop B do A := succ(A) od;     //Mithilfe von Loop wird Inhalt addiert
```

Ergebnis: Register A = 5;

3. Kopiere den Inhalt von Register A nach B. Beachten Sie, dass der Wert im Register A erhalten bleiben soll.

Register A = 5; Register C = A;

```
whilenot iszero(B) do
        B := pred(B);          //Register B wird auf 0 gesetzt
od;

whilenot iszero(A) do
        B := succ(B);          // Inhalt wird  Kopiert
        A := pred(A);
od;

whilenot iszero(C) do
        A := succ(A);          // Inhalt A wird wiederhergestellt
        C := pred(C);
od;
```

Ergebnis: Register B = 5; Register A = 5;

4. Signum-(Vorzeichen-)Test:

$$B := \text{sgn}(A) := \begin{cases} 1 & \text{falls } A > 0 \\ 0 & \text{sonst} \end{cases}$$

Beispiel(1): A = 10, B = 1

```
loop A do A := pred(A) od;      // Dann ist A = 0 durch 10 Iterationen
loop A do B := 0 od;            // Da A = 0 ist wird die Schleife übersprungen
                                // Register B liefert somit 1 bei A>0
```

Ergebnis: Register B = 1;

Bespiel(2): A = -10, B = 1

```
loop A do A := pred(A) od;      // Dann ist A = -20 durch 10 Iterationen
loop A do B := 0 od;            // Register B wird durch 20 Iterationen auf 0
                                gesetzt
                                // Register B liefert somit 0 bei A<0
```

Ergbenis: Register B = 0;

5. Vergleichstest:

$$C := A > B := \begin{cases} 1 & \text{falls } A > B \\ 0 & \text{sonst} \end{cases}$$

Register A = 10; Register B = 5; Register C = 1;

```
loop B do A := pred(A) od;      // A = 10 – 5 = 5
loop A do A := pred(A) od;      // A = 5 – 5 = 0
loop A do C := 0 od;            // schleife wird übersprungen da A = 0
```

Ergebnis: Register C = 1; // Register C bleibt 1;

Register A = 5; Register B = 10; Register C = 1;

```
loop B do A := pred(A) od;      // A = 5 – 10 = -5
loop A do A := pred(A) od;      // A = -5 -5 = -10
loop A do C := 0 od;            // C = 0 gesetzt
```

Ergebnis: Register C = 0;

Aufgabe 3

Schreiben Sie jeweils ein while-Programm für folgende Aufgaben:

1. Subtrahiere den Inhalt von Register B vom Inhalt von Register A: A := A - B. Ist B ≥ A, soll das Ergebnis 0 sein.

Bsp: Register A = 4; Register B = 2;
Register A = 2; Register B = 3;
Gesucht: A = A – B;

```
whilenot iszero(B) do
        if iszero(A) then                    // Fals B>= A dann ist A = 0

        else
                A := pred(A);                // Ansonsten A und B verringern
                B := pred(B);
        fi;
od;
```

2. Multipliziere den Inhalt von Register B mit dem Inhalt von Register A: A := A * B.

Bsp: Register A = 5; Register B = 3;
Register A = 3; Register B = 2;

```
loop B do
        loop A do C := succ(C) od;           // Mehre Loops verschachtelt
od;
```

Ergebnis: Register C = 15; Register C = 6;

3. Test auf Gleichheit:

$$C := A = B := \begin{cases} 1 & \text{falls } A = B \\ 0 & \text{sonst} \end{cases}$$

Bsp: Register A = 5; Register B = 5; Register C = 0;

```
whilenot iszero(A) do
        A := pred(A);           // dann ist A = 0
        B := pred(B);           // dann ist B = 0
od;

if iszero(B) then
        C := succ(C);           // C = 1
fi;
```

Ergebnis: Register C = 1;

Bsp: Register A = 4; Register B = 5; Register C = 0;

```
whilenot iszero(A) do
        A := pred(A);           // dann ist A = 0
        B := pred(B);           // dann ist B = 1;
od;

if iszero(B) then
        C := succ(C);           // wird übersprungen, somit ist C = 0;
fi;
```

Ergebnis: Register C = 0;

4. Abstand:

$$C := |A - B| := \begin{cases} A - B & \text{falls } A \geq B \\ B - A & \text{falls } A < B \end{cases}$$

Register A = 10; Register B = 5;

Register C := A; Register D := B;

```
whilenot iszero(B) do
        A := pred(A);                   // A = 5
        B := pred(B);                   // B = 0
od;

if iszero(A) then                       // Prüfen ob A = B wenn A = 0
        loop D do C := pred(C);
else
        loop A do A := pred(A) od;      // A = 5-5
        if iszero(A) then               // Prüfen ob A > 0 oder  A < 0;
                loop D do C := pred(C);
        else
                loop C do D := pred(D);
                C := D;
        fi;
fi;
```

Ergebnis: Register C = 5;