



Universität Karlsruhe

Informatik 1 WS 00/01

Institut für Telematik, Forschungsgruppe C&M
Prof. Dr. S. Abeck, R. Scholderer

Musterlösungen zum Übungsblatt 3

Lösung

1. Semi Thue System: Kaffeedosenenspiel (3 Punkte)

- a. Zeichenvorrat = {weiss, schwarz}

Semi-Thue-Regeln:

schwarz schwarz schwarz → schwarz

schwarz schwarz weiss → schwarz

schwarz weiss schwarz → schwarz

weiss schwarz schwarz → schwarz

weiss weiss schwarz → schwarz

weiss schwarz weiss → schwarz

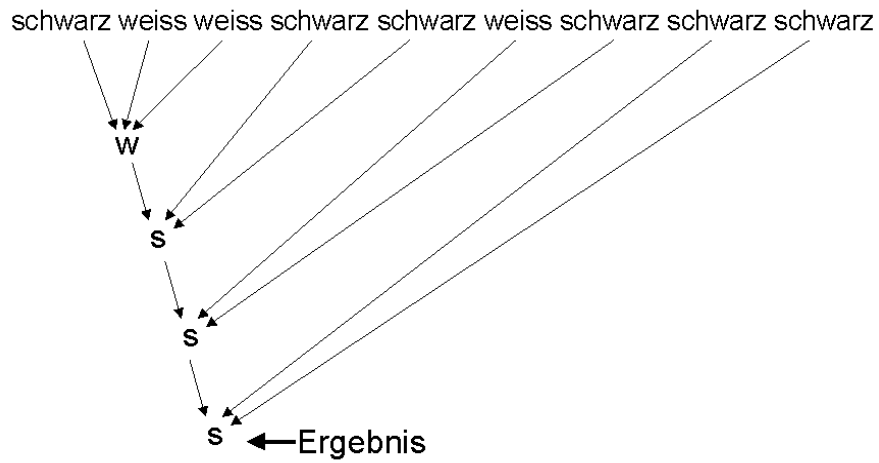
schwarz weiss weiss → schwarz

weiss weiss weiss → weiss

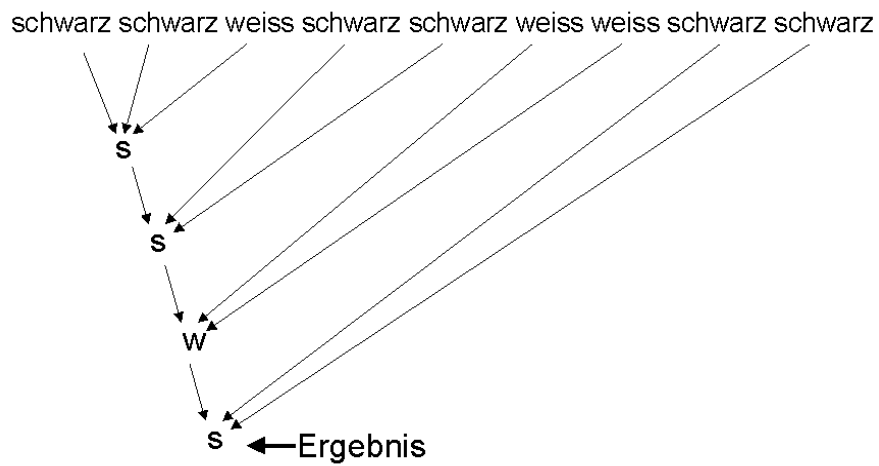
- b. Doseninhalte

1. [schwarz weiss weiss schwarz schwarz weiss schwarz schwarz schwarz]

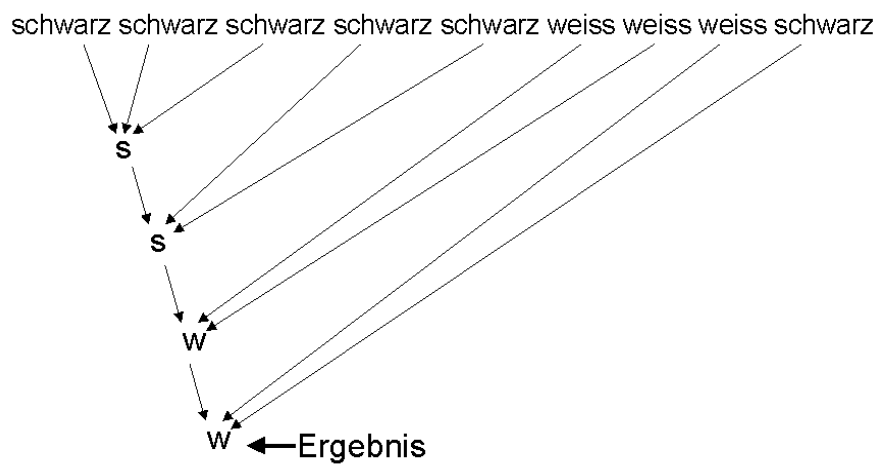
Ergebnis: schwarz



2. [schwarz schwarz weiss schwarz schwarz weiss weiss schwarz schwarz]
Ergebnis: schwarz



3. [schwarz schwarz schwarz schwarz schwarz weiss weiss weiss schwarz]
Ergebnis: weiss



Sind zu Beginn weniger als 3 Bohnen in der Dose, so terminiert das Kaffeedosenspiel sofort, da keine Regel anwendbar ist.

Sind 3 oder mehr Bohnen in der Dose, so läßt sich immer wenigstens eine Regel anwenden, die die Anzahl der Bohnen der Dose um 2 verringert. Nach endlich vielen Regelanwendungen verbleiben noch 3 bzw. 4 Bohnen in der Dose. Eine weitere Regelanwendung ergibt, daß nur noch eine bzw. zwei Bohnen in der Dose zurückbleiben. Dann ist keine Regel mehr anwendbar und das Kaffeedosenspiel terminiert.

- d. *Behauptung:* Ist die Anzahl der Bohnen zu Beginn gerade, so verbleiben 2 Bohnen in der Dose, ist sie ungerade, so verbleibt eine Bohne.

Beweis: Sei b die Anzahl der Bohnen die sich zu Beginn in der Kaffeedose befinden.

1. Fall: b gerade $\rightarrow b = 2n$ ($n \hat{=} N$)

Falls $n = 1$, so ist $b = 2$ und das Kaffeedosenspiel terminiert sofort mit 2 Bohnen in der Dose.

Falls $n > 1$, so ist $b = 2n$ eine gerade Zahl mit $b \geq 4$. Da jede Regelanwendung, die Anzahl der Bohnen um 2 vermindert, ist die Anzahl der Bohnen zu jeder Zeit während des, Spiels $2k$ ($k \hat{=} N, k \geq 2$), also gerade. So wird die Anzahl der Bohnen nach endlich vielen Ziehungen 4. Durch eine weitere Ziehung, terminiert das Spiel nach c) mit zwei Bohnen in der Dose.

2. Fall: b ungerade $\rightarrow b = 2n-1$ ($n \hat{=} N$)

Falls $n = 1$, so ist $b = 1$ und das Kaffeedosenspiel terminiert sofort mit 1 Bohne in der Dose.

Falls $n > 1$, so ist $b = 2n-1$ eine ungerade Zahl mit $b \geq 3$. Da jede Regelanwendung, die Anzahl der Bohnen um 2 vermindert, ist die Anzahl der Bohnen zu jeder Zeit während des, Spiels $2k-1$ ($k \hat{=} N, k \geq 2$), also ungerade. So wird die Anzahl der Bohnen nach endlich vielen Ziehungen 3. Durch eine weitere Ziehung, terminiert das Spiel nach c) mit einer Bohne in der Dose.

Das Ergebnis ist anhand des Eingabewortes nicht eindeutig bestimmt. Für b) 3. existiert beispielsweise noch die Lösung:

sssswwws \rightarrow sssswws \rightarrow sssws \rightarrow sss \rightarrow s

Da beim Semi-Thue-System, die Stelle der Regelanwendungen nicht eindeutig bestimmt ist, kann das System bei gleichem Eingabewort zu unterschiedlichen Ergebnissen führen. Das System ist also nicht deterministisch.

Lösung

2. Markov-Algorithmus: Klammerkorrektheitstest (1,5 Punkte)

1. $() \rightarrow \varepsilon$ (Ein korrektes Klammerpaar wird eliminiert)
2. $) \rightarrow \cdot$ Fehler (Der Ausdruck beginnt mit $)$ und enthält kein korrektes Klammerpaar)
3. $(\rightarrow \cdot$ Fehler (Der Ausdruck enthält nur noch öffnende Klammern)
4. $\varepsilon \rightarrow \cdot$ korrekt (Der Ausdruck enthält keine Klammern mehr und ist korrekt)

Ablauf des Tests eines korrekten Klammerausdrucks

$((())((())) \rightarrow ((())(())) \rightarrow ((())) \rightarrow (()) \rightarrow () \rightarrow \varepsilon \rightarrow \text{.korrekt}$

Ablauf des Tests eines nicht korrekten Klammerausdrucks

$(() \rightarrow (\rightarrow \text{.Fehler ($

Lösung

3. Markov-Algorithmus: Strichfolgen-Subtraktion (1,5 Punkte)

Die Regeln lauten:

1. $| - | \rightarrow -$
2. $| - \rightarrow \cdot |$
3. $- | \rightarrow \cdot - |$
4. $- \rightarrow \cdot \varepsilon$

Ablauf für das Beispiel: $||| | - | | |$

- 1) $||| | - | | | \rightarrow ||| | - |$ [Anwendung von $| - | \rightarrow -$]
- 2) $||| | - | \rightarrow || - |$ [Anwendung von $| - | \rightarrow -$]
- 3) $|| - | \rightarrow | -$ [Anwendung von $| - | \rightarrow -$]
- 4) $| - \rightarrow \cdot |$ [Anwendung von $| - \rightarrow \cdot |$]

Ablauf für das Beispiel: $| - | |$

- 1) $| - | | | \rightarrow | - |$ [Anwendung von $| - | \rightarrow -$]
 - 2) $| - | | \rightarrow - |$ [Anwendung von $| - | \rightarrow -$]
 - 3) $- | \rightarrow \cdot - |$ [Anwendung von $- | \rightarrow \cdot - |$]
-

Lösung

4. Chomsky-Grammatiken (3 Punkte)

- a. Typ CH-2, zum Beispiel wegen $X \rightarrow AXB$. CH-0 oder CH-1 Produktionen sind nicht enthalten.
Es werden die Wörter $ab, aabb, aaabbb, \text{ usw.}$ produziert.
Sprache: $L(G_1) = \{a^n b^n, n \geq 1\}$
- b. Typ CH-2, zum Beispiel wegen $B \rightarrow bcB$. (Bei CH-3 dürfen Terminale nicht aus Σ^* sein.) CH-0 oder CH-1 Produktionen sind nicht enthalten.
Es werden Wörter wie aaa
Sprache: $L(G_2) = \{a\} \cup \{a^n (bc)^m, n \geq 0, m \geq 1\}$
- c. Typ CH-1, zum Beispiel wegen $dR \rightarrow Rd$. CH-0 Produktionen kommen nicht vor.
Es werden Wörter $abcd, aabbccdd, \text{ usw.}$ erzeugt.

Sprache: $L(G_3) = \{a^n b^n c^n d^n, n > 1\}$

Lösung

5. Optimierung des JAVA-Programms ggT (1 Punkt)

```
import infol.*
public class GGT {
    public static void main (String args[]){
        int q;
        int p;
        q = Console.in.readInt();
        p = Console.in.readInt();
        int r;
        r = p%q;
        while (r!=0) {
            p = q;
            q = r;
            r = p%q;
        }
        System.out.println("Der größte gemeinsame Teiler lautet"+q+".");
    }
}
```
